Я.Т. Гринчишин
В.И.Ефимов
А.Н.Ломакович

АЛГОРИТМЫ И ПРОГРАММЫ НА БЕЙСИКЕ

Допущено Государственным комитетом СССР
по народному образованию
в качестве учебного пособия
для студентов педагогических институтов
по физико-математическим специальностям

МОСКВА «ПРОСВЕЩЕНИЕ» 1988

ББК 32.973-01
Г85

Рецензенты:
кафедра информатики и вычислительной техники Московского педагогического института им. Н. К. Крупской (зав. кафедрой, профессор, доктор технических наук *Я. А. Ваграменко*);
доцент, кандидат физико-математических наук *А. А. Богуславский* (Коломенский педагогический институт)

Гринчишин Я. Т. и др.

Г85 Алгоритмы и программы на Бейсике: Учеб. пособие для студентов пед. ин-тов по физ.-мат. спец./Я. Т. Гринчишин, В. И. Ефимов, А. Н. Ломакович.— М.: Просвещение, 1988.— 160 с.: ил.

ISBN 5-09-000406-4

Настоящее пособие содержит стандартные программы по некоторым разделам алгебры, теории чисел и математического анализа, написанные на языке программирования Бейсик и ориентированные на вычислительные процессы. Книга предназначена для студентов физико-математических специальностей педагогических институтов.

Г $\frac{4309020000-806}{103(03)-88}$ КБ—51—47—1987 **ББК 32.973-01**

ISBN 5-09-000406-4

ВВЕДЕНИЕ

В последние годы все более широкое распространение получают персональные электронно-вычислительные машины (ПЭВМ). Их популярность возрастает среди различных категорий пользователей: ученых, инженеров, студентов и учащихся. Внедрение ПЭВМ в учебный процесс высшей и средней школы дает возможность овладеть компьютерной грамотностью, использовать вычислительную технику в практической работе. При этом на первый план выдвигается проблема внедрения в среду пользователей хорошо отработанных, надежных алгоритмов. Практически все персональные компьютеры снабжены трансляторами языка программирования Бейсик. В настоящее время имеется лишь книга [22]*, содержащая программы на Бейсике, однако в ней отсутствуют программы, реализующие стандартные математические методы. Книги же, выпускаемые различными фирмами, производящими ПЭВМ, содержат наиболее простые и не всегда удобные пользователю программы. В то же время существует достаточно много источников с программами на других языках программирования. В первую очередь надо отметить сборники [4—8], а также [20], [21].

Нами предпринята попытка перевести на Бейсик некоторые стандартные программы из [4—8], [20]. В пособии содержатся также и другие написанные нами программы. Всего их около 120. Подбор программ ориентирован на внедрение их в педагогическую практику обучения студентов физико-математических факультетов педвузов, проведение инженерных и научных расчетов.

Несколько замечаний об используемом в настоящем пособии диалекте языка Бейсик. Известно много вариантов Бейсика: MSX, Бейсик-80, Бейсик-Агат, Atari, Apple II, Бейсик / *F* и др. Все они отличаются как один от другого, так и от стандартного варианта, описанного в [1]. Однако во всех версиях языка имеется общее ядро. Программы, написанные на его основании, могут быть использованы в любой версии. Опишем особенности такого сокращенного ядра.

* Здесь и далее число в квадратных скобках означает ссылку на источник, указанный под этим номером из списка литературы в конце книги.

1. Во всех операторах присваивания опускается служебное слово LET (пусть). Например, вместо

```
10 LETX=8
```

мы пишем:

```
10X=8
```

2. Идентификатор переменной или массива состоит не более чем из двух символов — латинской буквы и, возможно, второй цифры. Например, A, C, B2, C3 (10).

Практически мы нигде не пользуемся переменной целого типа (A%, B%) и текстовой переменной (A$, B3$). То же самое относится и к массивам.

3. В операторе PRINT (печать) разделитель ; задает печать вплотную, разделитель , задает печать по зонам. Наличие точки с запятой ; в конце списка оператора PRINT задает печать следующего оператора PRINT в ту же строку, иначе вывод начинается с новой строки. При наличии в операторе PRINT строковой константы "..." точка с запятой может опускаться. Например, действия операторов

```
10 PRINT "X="; X; "Y="; Y
```

и

```
10 PRINT "X="X"Y="Y
```

совпадают.

4. В операторе INPUT (ввод) строковая константа заменяет знак вопроса (?) при вводе. Например:

```
20 INPUT"X, Y="; X, Y
```

В MSX-системе такая замена не происходит.

Ввод нескольких переменных осуществляется через разделитель ', ' либо новым оператором INPUT.

5. В одной строке может быть записано несколько операторов через разделитель ': '. Например:

```
100 X=5:Y=2*X-4: GOTO 200
```

6. Оператор GOTO (перейти на) передает управление из текущей строки на указанный номер строки:

```
100 GOTO 1000
. . . . . .
1000 PRINT X
```

7. Оператор IF A THEN... полностью вычисляет логическое выражение A, и, если оно истинно, выполняются последовательно операторы, стоящие после THEN в данной строке, если оно ложно, то выполняются операторы следующей строки. Например:

```
120 IF X<0 THEN X=2*X:Y=Z:GOTO 1000
130 PRINT "X>=0":END
. . . . . .
1000 PRINT X
```

Если $x<0$, то выполняются все операторы после THEN: X=2*X:Y=Z:GOTO 1000. Если $x \geqslant 0$, то выполняется оператор в строке 130.

Служебное слово ELSE (иначе) не используется ввиду его отсутствия в некоторых системах.

8. Мы нигде не используем матричные операции (в большинстве версий языка они отсутствуют).

9. Оператор цикла FOR I=1 TO N STEP 2...NEXT I выполняется следующим образом: присваивается начальное значение переменной цикла ($i=1$), затем выполняется тело цикла и к значению переменной цикла прибавляется шаг (в примере шаг равен 2). И только после этого выполняется сравнение с конечным значением переменной (n). Например, следующие два оператора цикла будут выполнены одинаково:

```
10 FOR I=2 TO 1 STEP 3: PRINT I: NEXT
20 FOR I=2 TO 2 STEP 3: PRINT I:NEXT
```

Если служебное слово STEP (шаг) опущено, то шаг равняется единице по умолчанию. Если после служебного слова NEXT (следующий) не указано имя переменной, то NEXT относится к последнему заголовку цикла FOR (для)... TO (до)... Если необходимо обеспечить невыполнение тела цикла FOR I=10 TO I1... NEXT при $i_1<i_0$, перед заголовком цикла надо поставить условный оператор.

IF I1<I0 THEN...

или заменить оператор цикла, например, следующей конструкцией:

```
100 I=10
110 IF I>I1 THEN 210
. . . тело цикла
200 I=I+1: GOTO 110
210 . . . . .
```

Служебным словом NEXT можно закрыть несколько операторов цикла: NEXT K, J

10. Оператор GOSUB (перейти к подпрограмме) 200 передает управление в подпрограмму, которая начинается в строке 200 и заканчивается оператором RETURN (возврат), который возвращает управление на следующий после GOSUB 200 оператор (необязательно находящийся в новой строке). Например, при выполнении фрагмента программы

```
10 X=10: GOSUB 200: PRINT F
20 END
. . . . . .
200 F=X*X:RETURN
```

будет напечатано число 100.

11. Мы стараемся избегать употребления функций пользователя DEFFN, так как во многих версиях эти функции могут быть только однооператорными и зависеть только от одной переменной. Такие функции используются только в главе 6.

12. При каждом новом выполнении программы во многих вариантах языка происходит обнуление значений всех переменных.

Мы, однако, это явление не используем и специально вводим нули в ходе выполнения программы.

13. При вводе исходной информации по оператору INPUT, как правило, не производится ее анализ. Предполагается, что ввод должен быть правильным. Отметим, что в некоторых системах неправильный ввод может привести к сбою в программе. Так, в MSX версии на ПЭВМ YAMAXA, если по INPUT X ввести символ $, это приведет к синтаксической ошибке.

14. Мы нигде не используем букву *O*, учитывая, что при печати на принтере не всегда различается символ *O* (буква) и число (нуль).

15. Из логических операций используются AND (конъюнкция), OR (дизъюнкция), = (равно).

16. В обозначениях переменных и массивов можно использовать одинаковые символы, например A и A (10).

17. Отношения ($\geqslant$), ($\leqslant$), ($\neq$) записываются в виде "$>=$", "$<=$", "$<>$" соответственно.

18. Из стандартных функций используются следующие:

ABS (X) — абсолютное значение x;
ATH (X) — арктангенс (в радианах);
COS (X) — косинус (x в радианах);
SIN (X) — синус (x в радианах);
EXP (X) — e^x;
INT (X) — целая часть x;
SGN (X) — знаковая функция; принимает значение $+1$ для $x>0$, -1 для $x<0$, 0, если $x=0$;
LOG (X) — $\ln x$.

19. Предполагается, что нижняя граница индексов массива равна нулю. Например, оператор DIM A(2) определяет одномерный массив с элементами A(0), A(1), A(2).

20. Программы заканчиваются оператором END, хотя его использование и не является обязательным.

21. В программы нигде не включены операторы комментариев (REM).

Тексты всех Бейсик-программ приведены после директивной команды LIST (чтение) и представляют собой распечатку («листинг») разработанных программ. Контрольные примеры напечатаны после директивной команды RUN (выполнить или пуск) и представляют собой распечатку результата выполнения программы. Практически все программы снабжены детальными пояснениями алгоритма и структуры программы, приводится список и значение используемых идентификаторов. Там, где это возможно для данного алгоритма, в строках 10—90 производится ввод исходных данных, определение размерностей массивов, выдается сообщение о содержании программы, задается точность, пределы вычислений. Если использовать данную программу как подпрограмму, то эта часть в большинстве случаев должна быть опущена.

В строках 100—490 производится реализация основного алго-

ритма. В строках 500—990 размещаются подпрограммы, используемые в данной программе. Например, здесь производятся вычисления значений функций, которые интегрируются или для которых надо найти корень. Эта часть, как правило, должна изменяться пользователем. В строках с номером 1000 и более производится вывод полученных в результате вычислений значений.

22. Отметим, также, что в силу особенностей набора текста программ формат операторов печати PRINT не всегда соответствует печати после оператора RUN.

С момента написания данной книги в печати появился целый ряд работ, авторы которых преследуют сходные с нами цели. Отметим прежде всего отличную книгу В. П. Дьяконова.

Дьяконов В. П. Справочник по алгоритмам и программам на языке Бейсик для персональных ЭВМ.— М.: Наука, 1987.

В ней приведены сведения о характеристиках современных отечественных и зарубежных ПЭВМ, описаны некоторые версии Бейсика. Книга содержит более 300 прикладных программ.

Из других книг обратим внимание на следующие:

Моррил. Бейсик для ПК ИБМ.— М.: Финансы и статистика, 1987.

Фокс А., Фокс Д. Бейсик для всех.— М.: Энергоатомиздат, 1987.

Банди Б. Методы оптимизации: Вводный курс.— М.: Радио и связь, 1988.

Баласанян В. Э., Богдюкевич С. В., Шахвердов В. А. Программирование на микроЭВМ «Искра-226».— М.: Финансы и статистика, 1987.

Понимая, что в приведенных здесь программах могут содержаться описки и ошибки, мы обращаемся ко всем читателям и пользователям с просьбой присылать свои замечания и исправления в адрес Тернопольского пединститута: 282009, Тернополь, ул. Максименко, 2. Физико-математический факультет. Авторам.

Авторы благодарят за участие в работе Я. Г. Господарского и П. М. Маланюка по подготовке рукописи к изданию.

Глава I

ТЕОРИЯ ЧИСЕЛ И ЛИНЕЙНАЯ АЛГЕБРА

ТЕОРИЯ ЧИСЕЛ

1.1. Алгоритм Евклида для нахождения наибольшего общего делителя двух целых чисел

Наибольшим общим делителем (НОД) двух целых чисел называется такое наибольшее по модулю целое число, которое делит эти два числа. По определению НОД (0,0)=0.

Первый алгоритм вычисления НОД двух целых чисел a и b состоит в следующем: если хотя бы одно из чисел равно нулю, то НОД $(a, b)=|a+b|$, в других случаях последовательно находятся остатки r_i от делений:

$$\begin{aligned} &a \text{ на } b-a=bq_1+r_1, \\ &b \text{ на } r_1-b=r_1q_2+r_2, \\ &r_1 \text{ на } r_2-r_1=r_2q_3+r_3, \\ &\ldots\ldots\ldots\ldots \end{aligned} \qquad (1)$$

и т. д., пока r_{n+1} не станет равно нулю. Тогда
НОД $(a, b)=$НОД $(b, r_1)=$НОД $(r_1, r_2)=\ldots=$НОД $(r_{n-1}, r_n)=|r_n|$.
НОД нескольких чисел находят с учетом соотношения

$$\text{НОД}(a, b, c)=\text{НОД}(a, \text{НОД}(b, c)). \qquad (2)$$

В программе используются переменные: A, B — начальные числа; R2, R1, R — последовательные остатки от делений (1): r_i, r_{i+1}, r_{i+2}.

Структура программы:

10—30 — ввод чисел a и b;

100—150 — реализация алгоритма (1);

1000—1020 — вывод НОД.

Программа получена переводом на язык Бейсик Алгол-программы 7б [4].

В контрольном примере показано, что НОД (40902, 24140)=34.

```
JLIST
10 PRINT "ПРОГРАММА ВЫЧИСЛЯЕТ НАИБОЛЬШИЙ ОБЩИЙ ДЕЛИТЕЛЬ
   ДВУХ ЧИСЕЛ А И В."
20 INPUT "ВВЕДИТЕ ПЕРВОЕ ЧИСЛО A=";A
30 INPUT "ВВЕДИТЕ ВТОРОЕ ЧИСЛО B=";B
100 A=INT(A):B=INT(B)
```

```
110 IF A= 0 OR B=0 THEN R1=ABS (A+B):GOTO 1000
120 R2=A:R1=B
130 R=R2-R1 * INT (R2 / R1)
140 IF R< > 0 THEN R2=R1:R1=R: GOTO 130
150 R1=ABS (R1)
1000 PRINT "НАИБОЛЬШИЙ ОБЩИЙ ДЕЛИТЕЛЬ"
1010 PRINT "НОД ("A", "B") ="R1
1020 END
```

```
JRUN
ПРОГРАММА ВЫЧИСЛЯЕТ НАИБОЛЬШИЙ ОБЩИЙ ДЕЛИТЕЛЬ ДВУХ ЧИ-
СЕЛ А И В.
ВВЕДИТЕ ПЕРВОЕ ЧИСЛО A=40902
ВВЕДИТЕ ВТОРОЕ ЧИСЛО B=24140
НАИБОЛЬШИЙ ОБЩИЙ ДЕЛИТЕЛЬ
НОД (40902, 24140) =34
```

Второй алгоритм вычисления НОД (a, b) основан на бинарном алгоритме Евклида [15]. Вычисления проводятся на основании следующих свойств НОД:

$$\begin{aligned} &\text{НОД}\ (2m,\ 2n) = 2\ \text{НОД}\ (m,\ n),\\ &\text{НОД}\ (2m,\ 2n+1) = \text{НОД}\ (m,\ 2n+1),\\ &\text{НОД}\ (-m,\ n) = \text{НОД}\ (m,\ n). \end{aligned} \qquad (3)$$

Программа работает следующим образом:

1. Число A1 (B1) делится на такую степень числа два $R=2^{k_1}$ ($S=2^{k_2}$), чтобы получилось нечетное число A (B).

2. Если результаты делений совпадают (A=B), то НОД (A1, B1)=KA, где K=min {R, S}.

3. В противном случае (A≠B) проводим вычитание A—B (или B—A), результат снова делим на 2 столько раз, чтобы получить нечетное число, и возвращаемся к шагу 2.

В программе используются переменные: A1, B1 — исходные целые числа; A, B — текущие значения чисел, для которых определяем НОД; R, S — числа 2^{k_1} и 2^{k_2}; K=min {R, S}; $R1$ — НОД (a, b).

Структура программы:

10—20 — ввод a, b (A1, B1);

100 — задание исходных значений A, B, R, S;

110 — если a или $b=0$, то НОД (a, b)=$|a+b|$;

120—150 — деление чисел A и B на 2 требуемое число раз (первый шаг алгоритма);

160—170, 210 — реализация второго шага алгоритма;

180—210 — реализация третьего шага алгоритма;

1000 — вывод НОД (a, b).

```
JLIST
10 PRINT "БИНАРНЫЙ АЛГОРИТМ ЕВКЛИДА ДЛЯ НОД (A, B)"
20 INPUT "A, B="; A1, B1
100 A=ABS (A1):B=ABS (B1):R=1:S=1
110 IF A=0 OR B=0 THEN R1=ABS (A+B):GOTO 1000
```

```
120 IF 2 * INT (A / 2) < > A THEN 140
130 A=A / 2:R=R * 2: GOTO 120
140 IF 2 * INT (B / 2) < > B THEN 160
150 B=B / 2:S=S * 2: GOTO 140
160 K=S: IF K > R THEN K=R
170 IF A=B THEN 220
180 IF A > B THEN A=A-B: GOTO 200
190 B=B-A: R1=B:B=A:A=R1
200 A=A / 2: IF 2 * INT (A / 2)=A THEN 200
210 GOTO 170
220 R1=A * K
1000 PRINT "НОД(" A1"," B1")="R1
1010 END

JRUN
БИНАРНЫЙ АЛГОРИТМ ЕВКЛИДА ДЛЯ НОД (A, B)
A, B=34, 85
НОД (34,85)=17
```

1.2. Наименьшее общее кратное двух целых чисел

Наименьшим общим кратным двух целых чисел $a \neq 0$ и $b \neq 0$ называется наименьшее целое положительное число, которое делится на a и b.

Известно, что

$$\text{НОК}\ (a,\ b) = ab\ /\ \text{НОД}\ (a,\ b). \qquad (1)$$

Сначала по программе вычисляется НОД (a, b), затем НОК по формуле (1). Программа почти совпадает с первой программой для НОД в п. 1.1; введено обозначение — $R2 = \text{НОК}\ (a, b)$. Отличие программ состоит в том, что при $a=0$ или $b=0$ НОД не вычисляется.

```
JLIST
10 PRINT "ПРОГРАММА ВЫЧИСЛЯЕТ НАИМЕНЬШЕЕ ОБЩЕЕ КРАТНОЕ
   ДВУХ ЧИСЕЛ А И В."
20 INPUT "ВВЕДИТЕ ПЕРВОЕ ЧИСЛО A=";A
30 INPUT "ВВЕДИТЕ ВТОРОЕ ЧИСЛО B=";B
100 A=INT (A):B=INT (B)
110 IF A=0 OR B=0 THEN PRINT "ЧИСЛО РАВНО НУЛЮ!": GOTO 20
120 R2=A: R1=B
130 R=R2-R1 * INT (R2 / R1)
140 IF R < > 0 THEN R2=R1:R1=R: GOTO 130
150 R1=ABS (R1)
160 R2=A * B / R1
1000 PRINT "НАИМЕНЬШЕЕ ОБЩЕЕ КРАТНОЕ"
1010 PRINT "НОК("A","B")="R2
1020 END
```

```
JRUN
ПРОГРАММА ВЫЧИСЛЯЕТ НАИМЕНЬШЕЕ ОБЩЕЕ КРАТНОЕ ДВУХ ЧИ-
СЕЛ А И В.
ВВЕДИТЕ ПЕРВОЕ ЧИСЛО А=0
ВВЕДИТЕ ВТОРОЕ ЧИСЛО В=68
ЧИСЛО РАВНО НУЛЮ!
ВВЕДИТЕ ПЕРВОЕ ЧИСЛО А=34
ВВЕДИТЕ ВТОРОЕ ЧИСЛО В=85
НАИМЕНЬШЕЕ ОБЩЕЕ КРАТНОЕ
НОК (34,85) =170
```

1.3. Решето Эратосфена для нахождения простых чисел

Известно, что если целое положительное число $n \neq 1$ не делится ни на одно положительное простое число, не большее $\sqrt{n}$, то оно простое.

В последовательности чисел 2, 3, ..., n последовательно вычеркиваем каждое второе число после 2. Первое незачеркнутое число простое (3). Далее вычеркиваем каждое третье число после 3. Первое незачеркнутое число простое (5). Затем вычеркиваем каждое пятое число после 5 и т. д. до тех пор, пока не дойдем до числа, большего $\sqrt{n}$. Все числа, которые остаются, простые. Такой метод нахождения простых чисел называется решетом Эратосфена.

Количество простых чисел r не превышает:

$$\begin{aligned} &\text{int}\,(1{,}6n \,/\, \ln n + 1), \quad n \leqslant 200, \\ &\text{int}\left(\frac{n}{\ln n - 2} + 1\right), \quad n > 200. \end{aligned} \tag{1}$$

В программе используются переменные: N — число n; R — оценка количества простых чисел r по (1); $P(R)$ — массив простых целых чисел p_i, где $i = \overline{1, r}$; I, J, K — переменные циклов; $S = \sqrt{K}$.

Структура программы:

10—20 — ввод n;

100—120 — определение r;

130 — значения первых трех простых чисел 1, 2, 3;

140—190 — для всех нечетных чисел от 3 до n по решету Эратосфена находятся простые числа;

1000—1040 — вывод массива p_i.

Программа получена переводом на язык Бейсик программы 35б [4].

```
JLIST
  10 PRINT "ПРОГРАММА НАХОДИТ ВСЕ ПРОСТЫЕ ЧИСЛА ОТ 1 ДО N."
  20 INPUT "N=";N
 100 IF N > 200 THEN R=INT (N / (LOG (N)-2)+1):GOTO 120
 110 R=INT (1.6*N / LOG (N)+1)
```

```
 120 DIM P(R)
 130 P (1) =1:P (2) =2:P (3) =3:J=3
 140 FOR K=3 TO N STEP 2
 150 I=2:S=SQR (K)
 160 I=I+1
 170 IF P (I) > S THEN P (J) =K:J=J+1: GOTO 190
 180 IF INT (K / P (I)) * P (I) < >K THEN 160
 190 NEXT
1000 PRINT "ПРОСТЫЕ ЧИСЛА МЕНЬШЕ "N" РАВНЫ:"
1010 FOR I=1 TO R
1020 IF P (I) =0 THEN END
1030 PRINT" P ("I") ="P (I),
1040 NEXT
1050 END
```

```
JRUN
ПРОГРАММА НАХОДИТ ВСЕ ПРОСТЫЕ ЧИСЛА ОТ 1 ДО N.
N=50
ПРОСТЫЕ ЧИСЛА МЕНЬШЕ 50 РАВНЫ:
 P (1) =1          P (2) =2
 P (3) =3          P (4) =5
 P (5) =7          P (6) =11
 P (7) =13         P (8) =17
 P (9) =19         P (10) =23
 P (11) =29        P (12) =31
 P (13) =37        P (14) =41
 P (15) =43        P (16) =47
```

1.4. Разложение числа на простые множители

Всякое целое число $a \neq 1$ представим в виде произведения простых чисел

$$a = 2^{k_1} 3^{k_2} \ldots P_s^{k_s}. \qquad (1)$$

По программе сначала определяется степень k_1 числа 2, затем степени k_2, k_3 всех чисел до a. При этом автоматически остаются лишь степени простых чисел 2, 3, 5, ...

В программе используются переменные: A — разлагаемое число a; A_1 — знак числа a; J — переменная цикла, множитель числа a; k — показатель степени числа J.

Структура программы:

10—40 — ввод числа a;

100—160 — определение степеней простых множителей.

Вывод происходит в процессе выполнения программы.

В контрольном примере показано:

$$3080 = 2^3 \cdot 5^1 \cdot 7^1 \cdot 11^1.$$

```
JLIST
10 PRINT "ПРОГРАММА РАЗЛАГАЕТ НА ПРОСТЫЕ МНОЖИТЕЛИ ЦЕ-
   ЛЫЕ ЧИСЛА!"
```

```
 20 INPUT "ВВЕДИТЕ ЧИСЛО A=";A
 30 IF A=0 OR A < > INT (A) THEN PRINT "МНОЖИТЕЛЕЙ НЕТ!": END
 40 A1=SGN (A):A=ABS (A): PRINT "ЧИСЛО "A * A1 "РАЗЛАГАЕТСЯ НА
    МНОЖИТЕЛИ:"
100 FOR J=2 TO A
110 K=0
120 IF A / J < > INT (A / J) THEN 140
130 A=A / J:K=K+1: GOTO 120
140 IF K=O THEN 160
150 PRINT J"^"K
160 IF A=1 THEN END
170 NEXT
180 END
```

```
JRUN
ПРОГРАММА РАЗЛАГАЕТ НА ПРОСТЫЕ МНОЖИТЕЛИ ЦЕЛЫЕ ЧИСЛА!
ВВЕДИТЕ ЧИСЛО A=3080
ЧИСЛО 3080 РАЗЛАГАЕТСЯ НА МНОЖИТЕЛИ:
2^3
5^1
7^1
11^1
```

1.5. Нахождение числа сочетаний C_n^m

По программе находится число сочетаний из n элементов по m:

$$C_n^m = C_n^{n-m} = \frac{n(n-1)\ldots(n-m+1)}{m!} \quad (n \geqslant m). \tag{1}$$

Вычисление производится по рекуррентной формуле

$$C_n^m = C_n^{n-m} = \frac{n-m+1}{m} C_n^{m-1}, \quad C_n^0 = C_n^n = 1. \tag{2}$$

В программе используются переменные: N — число n; M — m; $C = C_n^m$; I — переменная цикла.

Структура программы:

10—30 — ввод чисел n и m;

100—130 — алгоритм вычисления по формуле (2);

1000—1010 — вывод C_n^m.

```
JLIST
  10 PRINT "ЧИСЛО СОЧЕТАНИЙ ИЗ N ПО M"
  20 INPUT "N=";N: INPUT "M=";M
  30 IF N<M THEN PRINT "ОШИБКА!": GOTO 20
 100 IF M > N / 2 THEN M=N-M
 110 IF M=0 THEN C=1: GOTO 1000
 120 N=N+1:C=1
 130 FOR I=1 TO M:C=C * (N-I) / I: NEXT
1000 PRINT "C="C
1010 END
```

```
JRUN
ЧИСЛО СОЧЕТАНИЙ ИЗ N ПО M
N=8
M=3
C=56
JRUN
ЧИСЛО СОЧЕТАНИЙ ИЗ N ПО M
N=3
M=8
ОШИБКА!
```

1.6. Коэффициенты полинома $(1+x)^n$

По программе вычисляются биномиальные коэффициенты C_n^i, где $i=\overline{0,n}$. Вычисление производится по формуле (2) п. 1.5. Коэффициенты C_n^i содержатся в массиве $C(N)$.

```
JLIST
10 PRINT "КОЭФФИЦИЕНТЫ ПОЛИНОМА (1+X)^N"
20 INPUT "N=";N: DIM C (N)
100 C (0) =1
110 FOR I=1 TO N:C (1) = (N-I+1) / I * C (I-1) :NEXT
1000 FOR I=0 TO N: PRINT "C ("I") ="C (I): NEXT
1010 END
```

```
JRUN
КОЭФФИЦИЕНТЫ ПОЛИНОМА (1+X)^N
N=6
C (0) =1
C (1) =6
C (2) =15
C (3) =20
C (4) =15
C (5) =6
C (6) =1
```

1.7. Диофантово уравнение $ax+by=c$

Пусть задано диофантово уравнение

$$ax+by=c, \text{ где } a,\ b,\ c\in\mathbf{Z}. \qquad (1)$$

Его частное решение можно найти следующим образом [16]. Находим наибольший общий делитель чисел a и b — НОД (a, b) и запомним последовательные частные q_i (см. п. 1.1) деления чисел $|a|$ и $|b|$ по алгоритму Евклида. Если НОД $(a, b)=$ $=c=0$, то уравнению (1) удовлетворяют любые целые x и y. Если НОД $(a, b)\neq 0$ и c на НОД (a, b) не делится, то уравнение (1) решения не имеет. В противном случае производим сокращение

коэффициентов a, b, c и получаем уравнение вида (1) со взаимно простыми a и b. Далее находим разложение 1 через $|a|$ и $|b|$. Последнее всегда возможно и требует знания частных q_i [16]. Таким образом, мы находим u и v, такие, что $|a|\,v+|b|\,v=1$. Умножив u и v на c sgn (a) и c sgn (b) соответственно, находим частное решение (1) — x_0, y_0. Общее решение тогда имеет вид $x=x_0+kb$, $y=y_0-ka$.

В программе используются переменные: $A1$, $B1$, $C1$, (A, B, C) — исходные (текущие) значения коэффициентов a, b, c; D = НОД (a, b); M — параметр в 5 раз больший, чем количество цифр в $\max\{|a|, |b|\}$; $X0$, $Y0$ — частное решение (1); N — число делений в алгориме Евклида; $Q(M)$ — частные q_i; U, V — рабочие переменные, решения уравнения $|a|\,u+|b|\,v=1$; R, S, D — рабочие переменные; I — переменная цикла.

Структура программы:

10—20 — ввод a, b, c;

100—170 — определение M;

180—240 — вычисление НОД (a, b), частных q_i, количества делений n, сравнение НОД и c;

250—260 — приведение уравнения к виду (1) с взаимно простыми a, b;

270—310 — решение уравнения $|a|\,u+|b|\,v=1$;

320—330 — нахождение частного решения и «малого» значения частного решения;

1000—1070 — вывод общего решения.

Программа получена переводом на язык Бейсик Алгол-программы 139б [6].

В контрольном примере найдено общее решение уравнения

$294x-50y=28$; $x=-13-25k$, $y=-77-147k$, $k=0, \pm1, \pm2, \ldots$.

```
JLIST
10 PRINT "ОБЩЕЕ РЕШЕНИЕ ДИОФАНТОВОГО УРАВНЕНИЯ AX+BY=C."
20 INPUT "A, B, C="; A1, B1, C1:A=A1:B=B1:C=C1
100 D=ABS (A):R=ABS (B):V=ABS (C)
110 S=D
120 M=D: IF M < R THEN M=R
130 IF M < V THEN M=V
140 IF M=0 THEN 180
150 I=1:U=1
160 U=U*10: IF U > M THEN M=5*I: GOTO 180
170 I=I+1: GOTO 160
180 N=0:I=0: DIM Q (M)
190 IF R=0 THEN 220
200 I=I+1
210 N=I:Q (I)=INT (S / R):D=R:R=S-R*Q (I):S=D: GOTO 190
220 IF D < > 0 THEN 250
230 IF C=0 THEN PRINT "X И Y — ПРОИЗВОЛЬНЫ!": END
```

```
240 GOTO 1000
250 IF INT (C / D) * D < > C THEN 1000
260 A=A / D:B=B / D:C=C / D
270 IF N=0 THEN V=0:U=1:GOTO 320
280 V=1:U=0
290 IF N=1 THEN 320
300 FOR I=N-1 TO 1 STEP-1
310 S=V:V=U-V * Q (I):U=S: NEXT I
320 X0=C * U * SGN (A): Y0=C * V * SGN (B)
330 IF B < > 0 THEN C=INT (X0 / B):X0=X0-C * B: Y0=Y0+C * A
340 GOTO 1010
1000 PRINT "УРАВНЕНИЕ РЕШЕНИЯ НЕ ИМЕЕТ!": END
1010 PRINT "УРАВНЕНИЕ "A1" * X";
1020 IF B1>0 THEN PRINT "+";
1030 PRINT B1" * Y=" C1
1040 PRINT "ИМЕЕТ ОБЩЕЕ РЕШЕНИЕ:"
1050 PRINT "X=" X0"+I * ("B");": PRINT "Y=" Y0"-I * ("A"),"
1060 PRINT "ГДЕ I — ЦЕЛОЕ ЧИСЛО."
1070 END
```

```
JRUN
ОБЩЕЕ РЕШЕНИЕ ДИОФАНТОВОГО УРАВНЕНИЯ AX+BY=C.
A, B, C=294, -50, 28
УРАВНЕНИЕ 294 * X-50 * Y=28
ИМЕЕТ ОБЩЕЕ РЕШЕНИЕ:
X=-13+I * (-25);
Y=-77-I * (147),
ГДЕ I — ЦЕЛОЕ ЧИСЛО.
```

ЛИНЕЙНАЯ АЛГЕБРА

1.8. Обращение матрицы с помощью расширенной матрицы

Программа обращает квадратную матрицу M размером $n \times n$ с помощью элементарных операций, которые приводят матрицу M к единичной. Обозначим расширенную матрицу A:

$$A = \left(\begin{array}{c|c} m_{11}m_{12}\ldots m_{1n} & 10\ldots0 \\ m_{21}m_{22}\ldots m_{2n} & 01\ldots0 \\ \ldots\ldots & \\ m_{n1}m_{n2}\ldots m_{nn} & 00\ldots1 \end{array} \right). \qquad (1)$$

К числу элементарных операций относятся:

1. Перестановка двух столбцов (строк) матрицы.
2. Умножение строки (столбца) на $\lambda \neq 0$.
3. Сложение двух строк (столбцов).

В программе используются переменные: $M(N, N)$ — обращаемая матрица; $A(N, 2N)$ — расширенная мат-

рица; $S=1$ — матрица вырождена; T — рабочая переменная; I, J, K — переменные циклов; $M=2N$.

Структура программы:

10—50 — ввод размерности матрицы (n) и элементов матрицы m_{ik};

100—150 — определение элементов расширенной матрицы A; (1).

160—350 — выполнение элементарных операций над элементами расширенной матрицы с целью приведения ее левой части к диагональному виду;

1000—1030 — вывод обращенной матрицы.

Программа получена переводом на язык Бейсик программы 42б [4].

В контрольном примере находится матрица, обратная матрице M:

$$M=\begin{pmatrix} 2 & 1 & 3 \\ 3 & 6 & 1 \\ 8 & -4 & 2 \end{pmatrix}.$$

```
JLIST
  10 PRINT "ОБРАЩЕНИЕ МАТРИЦЫ N * N С ПОМОЩЬЮ РАСШИРЕННОЙ
     МАТРИЦЫ"
  20 INPUT "N=";N:DIM M (N, N), A (N, 2 * N)
  30 FOR I=1 TO N:FOR K=1 TO N
  40 PRINT "M ("I", "K ;:INPUT") ="; M (I, K)
  50 NEXT K: NEXT I
 100 M=2 * N:S=0
 110 FOR I=1 TO N: FOR J=1 TO M
 120 IF J<=N THEN A (I, J) =M (I, J): GOTO 150
 130 IF J=N+I THEN A (I, J) =1: GOTO 150
 140 A (I, J) =0
 150 NEXT J: NEXT I
 160 FOR I=1 TO N
 170 K=I
 180 IF A (K, I) <>0 THEN 220
 190 S=1: IF K<N THEN K=K+1: GOTO 210
 200 GOTO 1000
 210 GOTO 180
 220 IF S<>1 THEN 270
 230 FOR J=1 TO M
 240 T=A (K, J):A (K, J) =A (I, J):A (I, J) =T
 250 NEXT J
 270 FOR J=M TO I STEP-1:A (I, J) =A (I, J) / A (I, I): NEXT J
 280 FOR K=1 TO N
 290 IF K=I THEN 320
 310 FOR J=M TO I STEP-1:A (K, J) =A (K, J) -A (I, J) * A (K, I): NEXT J
 320 NEXT K: NEXT I
 330 FOR I=1 TO N: FOR J=1 TO N
 340 M (I, J) =A (I, J+N):NEXT J: NEXT I
```

```
 350 S=0
1000 IF S=1 THEN PRINT "МАТРИЦА ВЫРОЖДЕНА!": END
1010 FOR I=1 TO N: FOR J=1 TO N
1020 PRINT "M("I", "J")^(-1)="M (I, J)
1030 NEXT:NEXT
1040 END

JRUN
ОБРАЩЕНИЕ МАТРИЦЫ N * N С ПОМОЩЬЮ РАСШИРЕННОЙ МАТРИЦЫ.
N=3
M (1, 1)=2
M (1, 2)=1
M (1, 3)=3
M (2, 1)=3
M (2, 2)=6
M (2, 3)=1
M (3, 1)=8
M (3, 2)=-4
M (3, 3)=2
M (1, 1)^(-1)=-.109589041
M (1, 2)^(-1)=.0958904109
M (1, 3)^(-1)=.116438356
M (2, 1)^(-1)=-.0136986301
M (2, 2)^(-1)=.136986301
M (2, 3)^(-1)=-.0479452055
M (3, 1)^(-1)=.410958904
M (3, 2)^(-1)=-.109589041
M (3, 3)^(-1)=-.0616438356
```

1.9. Вычисление определителя методом триангуляции

По программе вычисляется определитель матрицы $\det(a_{ij})$. Матрица A имеет размер $n \times n$. Триангуляция производится с выбором максимального элемента.

Алгоритм вычисления: а) в первом столбце определяется максимальный элемент M1. Если он находится в j-й строке, то первая и j-я строки переставляются. При $M1=0$ определитель равен нулю; б) затем из элементов a_{ij} $(j \geqslant 2)$ вычитаются соответствующие элементы первого столбца, умноженные на a_{1j}/a_{11}. В результате получим:

$$\det A = \Delta_n = a_{11} \begin{vmatrix} 1 & 0 & \dots & 0 \\ \frac{a_{21}}{a_{11}} & a_{22}^{(1)} & \dots & a_{2n}^{(1)} \\ \cdot & \cdot & \cdot & \cdot \\ \frac{a_{n1}}{a_{11}} & a_{n2}^{(1)} & \dots & a_{nn}^{(1)} \end{vmatrix} = a_{11}\Delta_{n-1}, \tag{1}$$

где $a_{ij}^{(1)} = a_{ij} - a_{i1}a_{1j} / a_{11}$. (2)

К определителю матрицы $a_{ij}^{(1)}$ применяем тот же прием еще $(n-2)$ раза. Окончательно получим:

$$\Delta_n = a_{11} a_{22}^{(1)} \ldots a_{nn}^{n-1}. \qquad (3)$$

При перестановке столбцов учитывается изменение знака определителя.

В программе используются переменные: $A(N, N)$ — матрица a_{ij}; N — размер матрицы (n); $M1$ — значение максимального элемента; T — рабочая переменная; D — значение определителя; K, L, I, J, F, Z — переменные циклов.

Структура программы:

10—50 — ввод n, a_{ij};

120—160 — выбор максимального элемента $M1$;

170 — если $M1=0$, то определитель равен нулю;

180—230 — перестановка J- и K-й строк;

240—290 — вычисления по формуле (2);

300 — вычисления по формуле (3);

110—310— повторение алгоритма перехода от Δ_i к Δ_{i-1};

1000 — вывод $\det(a_{ij})$.

Программа получена переводом на язык Бейсик программы 41б [4].

В контрольном примере найдено:

$$\begin{vmatrix} 3 & 2 & 1 \\ 5 & 3 & 6 \\ 4 & 3 & 6 \end{vmatrix} = -9.$$

```
JLIST
  10 PRINT "ВЫЧИСЛЕНИЕ ДЕТЕРМИНАНТА."
  20 INPUT "N=";N: DIM A (N, N)
  30 FOR I=1 TO N: FOR K=1TO N
  40 PRINT "A("I","K ;: INPUT ") =";A (I, K)
  50 NEXT:NEXT
 100 D=1
 110 FOR K=1 TO N: M1=0:1=K
 120 IF I>N THEN 170
 130 T=A (I, K)
 140 IF ABS (T)<=ABS (M1) THEN 160
 150 M1=T:J=I
 160 I=I+1:GOTO 120
 170 IF M1=0 THEN D=0: GOTO 1000
 180 IF J=K THEN 230
 190 D=-D:L=K
 200 IF L>N THEN 230
 210 T=A (J, L):A (J, L)=A (K, L):A (K, L)=T
 220 L=L+1: GOTO 200
 230 F=K+1
 240 IF F>N THEN 300
 250 T=A (F, K) / M1
 260 Z=K+1
```

```
 270 IF Z>N THEN 290
 280 A (F, Z)=A (F, Z)-T*A (K, Z):Z=Z+1: GOTO 270
 290 F=F+1:GOTO 240
 300 D=D*A (K, K)
 310 NEXT
1000 PRINT "DET="; D: END

JRUN
ВЫЧИСЛЕНИЕ ДЕТЕРМИНАНТА.
N=3
A (1, 1)=3
A (1, 2)=2
A (1, 3)=1
A (2, 1)=5
A (2, 2)=3
A (2, 3)=6
A (3, 1)=4
A (3, 2)=3
A (3, 3)=6
DET=-9
```

1.10. Решение системы линейных алгебраических уравнений методом Гаусса

Программа решает неоднородную систему n линейных алгебраических уравнений с n неизвестными:

$$\begin{cases} a_{11}x_1+a_{12}x_2+\ldots+a_{1n}x_n=a_{1n+1}, \\ a_{21}x_1+a_{22}x_2+\ldots+a_{2n}x_n=a_{2n+1}, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{n1}x_1+a_{n2}x_2+\ldots+a_{nn}x_n=a_{nn+1}. \end{cases} \tag{1}$$

Вначале находим отличный от нуля коэффициент при x_1. Соответствующее уравнение переставляем с первым (если это необходимо). Получаем систему (1) с $a_{11}\neq 0$. Разделив коэффициенты этого уравнения на a_{11}, получим:

$$x_1+b_{12}x_2+\ldots+b_{1n}x_n=b_{1n+1}. \tag{2}$$

С помощью (2) исключаем x_1 из системы (1):

$$\begin{cases} a^{(1)}_{22}x_2+a^{(1)}_{23}x_3+\ldots+a^{(1)}_{2n}x_n=a^{(1)}_{2n+1}, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a^{(1)}_{n2}x_2+a^{(1)}_{n3}x_3+\ldots+a^{(1)}_{nn}x_n=a^{(1)}_{nn+1}, \end{cases} \tag{3}$$

где

$$a^{(1)}_{ij}=a_{ij}-a_{i1}b_{1j},\ i,\ j=\overline{2,\ n}. \tag{4}$$

Система (3) содержит $n-1$ уравнение. Применяем описанную выше процедуру к системе (3). Операции повторяем требуемое число раз, пока не приведем систему к виду

$$\begin{aligned} x_1+c_{12}x_2+c_{13}x_3+\ldots+c_{1n}x_n&=c_{1n+1}, \\ x_2+c_{23}x_3+\ldots+c_{2n}x_n&=c_{2n+1}, \end{aligned} \tag{5}$$

. . . . .

$x_n = c_{nn+1}$

Если $\det(A)=0$, то система (1) не имеет решений и программа завершается (см. 140).

Из (5) легко определить x_n, x_{n-1}, . . ., x_1.

В программе используются переменные: N — количество уравнений и неизвестных (n); $A(N, N+1)$ — коэффициенты линейных уравнений; $X(N)$ — решение системы (x_i); U, K; M, I, J — переменные циклов, T — рабочая переменная.

Структура программы:

10—70 — ввод исходных данных;

120—130 — поиск отличного от нуля коэффициента при x_u;

170—190 — перестановка U-го и K-го уравнений;

200 — определение коэффициентов уравнения типа (2);

210—250 — определение коэффициентов уравнения с исключенной переменной;

270—280 — решение системы уравнений (5);

1000—1010 — вывод решения.

Программа получена переводом на язык Бейсик программы 126б [6].

```
JLIST
  10 PRINT "РЕШЕНИЕ N ЛИНЕЙНЫХ УРАВНЕНИЙ ПО ГАУССУ"
  20 INPUT "N=";N: DIM A (N, N+1), X (N)
  30 PRINT "ВВОД КОЭФФИЦИЕНТОВ УРАВНЕНИЙ:"
  40 FOR I=1 TO N: FOR K=1 TO N+1
  50 IF K=N+1 THEN PRINT "ПРАВАЯ ЧАСТЬ:"
  60 PRINT "A ("I","K;: INPUT ") =";A (I, K)
  70 NEXT K: NEXT I
 100 U=0
 110 U=U+1:K=U
 120 IF A (K, U)<>0 THEN 150
 130 K=K+1: IF K<=N THEN 120
 140 PRINT "СИСТЕМА НЕ ОПРЕДЕЛЕНА.": END
 150 IF K=U THEN 200
 170 FOR M=U TO N+1
 180 T=A (U, M):A (U, M) =A (K, M):A (K, M) =T
 190 NEXT
 200 FOR J=N+1 TO U STEP-1:A (U, J) =A (U, J) / A (U, U):NEXT
 210 M=N+1: IF K+1>N THEN 260
 220 FOR I=K+1 TO N
 230 FOR J=U+1 TO M
 240 A (I, J) =A (I, J) -A (I, U) * A (U, J)
 250 NEXT J: NEXT I
 260 IF U<>N THEN 110
 270 FOR I=N TO 1 STEP-1:X (I) =A (I, M):IF I=1 THEN 290
 280 FOR K=I-1 TO 1 STEP-1:A (K, M) =A (K, M) -A (K, I) * X (I): NEXT
 290 NEXT I
1000 PRINT "РЕШЕНИЕ СИСТЕМЫ РАВНО:"
```

```
1010 FOR I=1 TO N: PRINT "X ("I") ="X (I): NEXT I
1020 END

JRUN
РЕШЕНИЕ N ЛИНЕЙНЫХ УРАВНЕНИЙ ПО ГАУССУ
N=3
ВВОД КОЭФФИЦИЕНТОВ УРАВНЕНИЙ:
A (1, 1) = -2
A (1, 2) =3
A (1, 3) =4
ПРАВАЯ ЧАСТЬ:
A (1, 4) =21
A (2, 1) =1
A (2, 2) =5
A (2, 3) =7
ПРАВАЯ ЧАСТЬ:
A (2, 4) = -3
A (3, 1) =4
A (3, 2) =2
A (3, 3) =1
ПРАВАЯ ЧАСТЬ:
A (3, 4) = -5
РЕШЕНИЕ СИСТЕМЫ РАВНО:
X (1) = -7.74074076
X (2) = 19.6666667
X (3) = -13.3703704
```

1.11. Обращение матрицы методом Гаусса

Программа находит матрицу, обратную к квадратной матрице A размером $n \times n$, по методу Гаусса. Для неособенной матрицы $A=(a_{ij})$ находится матрица $A^{-1}=(x_{ij})$, такая, что

$$AA^{-1}=E, \tag{1}$$

где E — единичная матрица.

Уравнение (1) представляет собой n систем n линейных уравнений для n^2 неизвестных x_{ij}. Каждая из систем имеет одну и ту же основную матрицу A и различные свободные члены. Все системы решаются одновременно методом Гаусса (см. п. 1.10).

Если матрица A близка к вырожденной, то программа завершается сообщением об этом (см. 190).

В программе используются переменные: N — порядок матрицы (n); $A(N, N)$ — массив, матрица $A=(a_{ij})$; $E1$ — значения главных элементов должно быть больше чем $E1$, в противном случае вычисления не выполняются; W — значение максимального элемента в строках матрицы A; $C(N)$, $B(N)$ — массивы, в которых запоминаются элементы главного столбца и строки соответственно; $Z(N)$ — массив, в котором запоминается

порядок перестановки столбцов; I, J, K — переменные циклов; Y — рабочая переменная.

Программа получена переводом на язык Бейсик программы 120б [6].

В контрольном примере найдена обратная матрица к следующей матрице:

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 0 \\ 3 & 1 & 2 \end{pmatrix}^{-1} = \begin{pmatrix} -2/9 & 1/9 & 1/3 \\ 4/9 & 7/9 & -2/3 \\ 1/9 & -5/9 & 1/3 \end{pmatrix}.$$

```
JLIST
  10 PRINT "ОБРАЩЕНИЕ МАТРИЦЫ N * N"
  20 INPUT "N=";N: DIM A (N, N), B (N), C (N), Z (N)
  30 FOR I=1 TO N: FOR K=1 TO N
  40 PRINT "A("I","K;: INPUT ")=";A (I, K)
  50 NEXT K: NEXT I
  60 E1=1E-6
 100 D=1
 110 FOR I=1 TO N:Z (I)=I: NEXT
 120 FOR I=1 TO N
 130 K=I:Y=A (I, I)
 140 IF I+1>N THEN 180
 150 FOR J=1+I TO N
 160 W=A (I, J): IF ABS (W)>ABS (Y) THEN K=J:Y=W
 170 NEXT
 180 D=D * Y
 190 IF ABS (Y)<E1 THEN PRINT"МАТРИЦА БЛИЗКА К ВЫРОЖДЕН-
     НОЙ": END
 200 Y=1 / Y
 210 FOR J=1 TO N
 220 C (J)=A (J, K):A (J, K)=A (J, I):A (J, I)=-C (J) * Y
 230 B (J)=A (I, J) * Y:A (I, J)=B (J)
 240 NEXT
 250 J=Z (I):Z (I)=Z (K):Z (K)=J:A (I, I)=Y
 260 FOR K=1 TO N: IF K=I THEN 300
 270 FOR J=1 TO N: IF J=I THEN 290
 280 A (K, J)=A (K, J)-B (J) * C (K)
 290 NEXT J
 300 NEXT K
 310 NEXT I
 320 FOR I=1 TO N
 330 K=Z (I): IF K=I THEN 390
 340 FOR J=1 TO N
 350 W=A (I, J):A (I, J)=A (K, J):A (K, J)=W
 360 NEXT J
 370 P=Z (I):Z (I)=Z (K):Z (K)=P:D=-D
 380 GOTO 330
 390 NEXT I
```

```
1000 FOR I=1 TO N: FOR K=1 TO N
1010 PRINT "A("I","K")^(-1)="A (I, K)
1020 NEXT K: NEXT I
1030 END
```

```
JRUN
ОБРАЩЕНИЕ МАТРИЦЫ N * N
N=3
A (1, 1)=1
A (1, 2)=2
A (1, 3)=3
A (2, 1)=2
A (2, 2)=1
A (2, 3)=0
A (3, 1)=3
A (3, 2)=1
A (3, 3)=2
A (1, 1)^(-1)=-.222222222
A (1, 2)^(-1)=.111111111
A (1, 3)^(-1)=.333333333
A (2, 1)^(-1)=.444444444
A (2, 2)^(-1)=.777777778
A (2, 3)^(-1)=-.666666667
A (3,1)^(-1)=.111111111
A (3, 2)^(-1)=-.555555556
A (3, 3)^(-1)=.333333333
```

Глава II

УРАВНЕНИЯ. ИНТЕРПОЛЯЦИЯ. РЯДЫ. ЭКСТРЕМУМЫ

ПОЛИНОМИАЛЬНЫЕ И ТРАНСЦЕНДЕНТНЫЕ УРАВНЕНИЯ

2.1. Корень уравнения $x=F(x)$

Программа находит корень уравнения $x=F(x)$ методом простой итерации с относительной погрешностью ε_1. По i-му приближению корня x_i находится следующее приближение по формуле

$$x_{i+1}=F(x_i),\ i=0,\ 1,\ 2,\ \ldots . \qquad (1)$$

Процесс продолжается до тех пор, пока относительная точность для двух последовательных приближений не станет меньше ε_1:

$$\left|\frac{x_{i+1}-x_i}{x_i}\right|<\varepsilon_1. \qquad (2)$$

Процесс итерации сходится на $[a,\ b]$, если

$$|F'(x)|<1 \qquad (3)$$

при всех $x\in(a,\ b)$.

В программе используются переменные: X — начальное приближение корня, на выходе значения корня; N — максимально допустимое число итераций; F — значение функции $F(x)$; $E1$ — относительная точность корня (ε_1).

Структура программы:

10—40 — ввод и задание исходных данных x_0, n, ε_1;

100—140 — реализация формулы (1);

120 — проверка условия (2);

500 — подпрограмма вычисления функции $F(x)$;

1000 — вывод корня x.

В контрольном примере найден один из двух корней уравнения $x=e^x-2$, а именно тот корень, где $|F'(x)|=e^x<1$: $x_1\approx-1.84140566\ldots$. Второй корень этого уравнения $x_2=$ $=1.146193\ldots$. Это же уравнение решается в п. 2.2 и 2.3.

```
JLIST
 10 PRINT "КОРЕНЬ УРАВНЕНИЯ X=F (X)!"
 20 INPUT "НАЧАЛЬНОЕ ПРИБЛИЖЕНИЕ КОРНЯ X="; X
 30 E1=1E-8
 40 N=100
100 FOR I=1 TO N: GOSUB 500
110 IF ABS (F-X)<E1*ABS (X) THEN 1000
```

```
 120 X=F
 130 NEXT I
 140 PRINT "ТОЧНОСТЬ НЕ ДОСТИГНУТА!": END
 500 F=EXP (X)-2: RETURN
1000 PRINT "КОРЕНЬ X=";X: END

JRUN
КОРЕНЬ УРАВНЕНИЯ X=F (X)!
НАЧАЛЬНОЕ ПРИБЛИЖЕНИЕ КОРНЯ X=-2
КОРЕНЬ X=-1.84140567
```

2.2. Корень уравнения $x=F(x)$ — модифицированный метод итераций

Программа находит корень уравнения модифицированным методом итерации. Условие сходимости в этом методе отличается от (3) п. 2.1. Процесс сходится к значению $x=\xi$ при условии, что в некоторый момент

$$|F(x)-\xi|\cdot|F''(\xi)|<|F'(\xi)-1|\cdot 2. \qquad (1)$$

Следующее приближение определяется по формулам:

$$x_0=a;\ x_{i+1}=x_i+d_i;$$
$$d_0=F(x_0)-x_0;\ d_i=d_{i-1}/(h_i-1); \qquad (2)$$
$$h_i=\frac{F(x_{i-1})-x_{i-1}}{F(x_i)-x_i}.$$

В программе используются переменные: A — началь ое приближение корня ($x_0=a$); $E1$ — относительная погрешность корня; N — максимально допустимое число итераций; X — значение корня, аргумент функции $F(x)$; F — значение функции; C — разность $F(x)-x$; H, G, B, D — рабочие переменные, используемые в формулах (2); M — параметр, который показывает, сколько раз поправка превосходила предыдущую. Если разность $F(x_i)-x_i$ имела одно и то же значение дважды подряд, то печатается сообщение об этом и вычисление прерывается.

Структура программы:

10—40 — ввод данных: a, ε_1, n;

100 — проверка того, является ли x нулем;

110—130 — то же для точки $x=a$;

140—210 — реализация формул (2);

500 — подпрограмма вычисления $F(x)$;

1000 — вывод значения корня.

Программа получена переводом на язык Бейсик программы 26б [4].

В контрольном примере найдены два корня уравнения $x=e^x-2$ (см. п. 2.1).

```
JLIST
 10 PRINT "КОРЕНЬ УРАВНЕНИЯ X=F (X)!"
 20 INPUT "НАЧАЛЬНОЕ ПРИБЛИЖЕНИЕ КОРНЯ A=";A
```

```
 30 E1=1E-8
 40 N=500
100 M=0:G=0:C=0:X=0: GOSUB 500: IF F=0 THEN 1000
110 X=A: GOSUB 500:
120 G=F:B=G-A:D=B:C=B
130 IF C=0 THEN 1000
140 FOR J=1 TO N
150 X=G: GOSUB 500:C=F-G:
160 IF ABS (C)<ABS (G) * E1 THEN 1000
170 H=B / C:B=C
180 IF H=1 THEN PRINT "ДВА РАВНЫХ ЗНАЧЕНИЯ!": END
190 IF H>0 AND H<2 THEN M=M+1
200 D=D / (H-1):G=G+D
210 NEXT J
220 PRINT "ТОЧНОСТЬ НЕ ДОСТИГНУТА!": END
500 F=EXP (X)-2: RETURN
1000 PRINT "КОРЕНЬ X="X: END

JRUN
КОРЕНЬ УРАВНЕНИЯ X=F (X)!
НАЧАЛЬНОЕ ПРИБЛИЖЕНИЕ КОРНЯ A=2
КОРЕНЬ X= 1.14619322
JRUN
КОРЕНЬ УРАВНЕНИЯ X=F (X)!
НАЧАЛЬНОЕ ПРИБЛИЖЕНИЕ КОРНЯ A= -2
КОРЕНЬ X= -1.84140566
```

2.3. Корень уравнения — метод половинного деления

По программе находится корень уравнения

$$F(x)=0, \qquad (1)$$

где $F(x)$ — непрерывная на отрезке $[a, b]$ функция, удовлетворяющая условию

$$F(a)\cdot F(b)<0. \qquad (2)$$

Для нахождения корня отрезок $[a, b]$ делится пополам: $\xi=(a+b)/2$ — и выбирается тот полуинтервал, на концах которого знаки $F(x)$ разные. Затем процесс деления повторяется до тех пор, пока длина интервала не станет меньше ε_1.

В программе используются переменные: A, B — границы интервала; $E1$ — погрешность в определении корня; X — аргумент функции, значение корня; F — значение функции в точке x; $F1$ — рабочая переменная, значение функции в одной из точек интервала.

Структура программы:

10—30 — ввод исходных данных: A, B;

100—170 — деление интервала $[A, B]$ пополам;

500 — вычисление $F(x)$;

1000 — вывод корня.

```
JLIST
  10 PRINT "ВВЕДИТЕ ИНТЕРВАЛ ЛОКАЛИЗАЦИИ КОРНЯ:"
  20 INPUT "(A, B) ="; A, B
  30 E1 = 1E - 8
 100 X = B: GOSUB 500: F1 = F: IF F = 0 THEN 1000
 110 X = A: GOSUB 500: IF F = 0 THEN 1000
 120 IF SGN (F1) = SGN (F) THEN PRINT "НАДО ИЗМЕНИТЬ ИНТЕРВАЛ
     (AB)!": GOTO 20
 130 X = (A + B) / 2: GOSUB 500
 140 IF SGN (F1) = SGN (F) THEN B = X: F1 = F: GOTO 160
 150 A = X
 160 IF ABS (A - B) > E1 THEN 130
 170 X = (A + B) / 2: GOSUB 500: GOTO 1000
 500 F = EXP (X) - X - 2: RETURN
1000 PRINT "КОРЕНЬ X = "X: PRINT "ФУНКЦИЯ F = "F
1010 END

JRUN
ВВЕДИТЕ ИНТЕРВАЛ ЛОКАЛИЗАЦИИ КОРНЯ:
(A, B) = 0,2
КОРЕНЬ X = 1.14619322
ФУНКЦИЯ F = -6.51925802E - 09
```

2.4. Решение уравнения методом Ньютона (метод касательных)

Действительный корень ξ уравнения

$$F(x)=0 \tag{1}$$

вычисляется методом Ньютона по итерационному уравнению

$$x_{k+1}=x_k-F(x_k)\,/\,F'(x_k). \tag{2}$$

Процесс (2) сходится к точному значению корня, если начальное приближение x_1 выбрано так, что

$$|F(x_1)\cdot F''(x_1)|<|F'(x_1)|^2$$

Оценка погрешности k-го приближения [9]

$$|x_k-\xi|\leqslant|F(x_k)\,/\,m_1|, \tag{3}$$

где

$$m_1=\min\{|F'(x)|\ x\in[a,\ b]\},$$

в программе производится по приближенной формуле

$$|F(x_n)\,/\,F'(x_n)|<\varepsilon_1. \tag{4}$$

В программе используются переменные: $X1$ — начальное приближение корня; $E1$ — погрешность определения корня ε_1 (4); X — аргумент функции; F, $F1$ — значения функции и ее производной.

Структура программы:

10—30 — ввод исходных данных: x_1, ε_1;

100—130 — реализация формулы (2);

500—520 — подпрограмма вычисления значения функции F и ее производной F';

1000—1010 — вывод корня и значения функции в этой точке x.

В контрольном примере найдены корни уравнения $x^3-2x^2-x+2=0$: $x_1=-1$, $x_2=1$, $x_3=2$.

```
JLIST
  10 PRINT "МЕТОД НЬЮТОНА ДЛЯ НЕЛИНЕЙНОГО УРАВНЕНИЯ"
  20 INPUT "НАЧАЛЬНОЕ ПРИБЛИЖЕНИЕ X1="; X1
  30 E1=1E-8
 100 X=X1
 110 GOSUB 500:F=F / F1:X=X-F
 120 IF ABS (F)>E1 THEN X1=X: GOTO 110
 130 GOSUB 500: GOTO 1000
 500 F=((X-2) * X-1)*X+2: F1=(3 * X-4) * X-1
 510 IF F1=0 THEN PRINT "ПРОИЗВОДНАЯ F' ("X")=0"
 520 RETURN
1000 PRINT "КОРЕНЬ X="X
1010 PRINT "ФУНКЦИЯ F="F
1020 END

JRUN
МЕТОД НЬЮТОНА ДЛЯ НЕЛИНЕЙНОГО УРАВНЕНИЯ
НАЧАЛЬНОЕ ПРИБЛИЖЕНИЕ X1=-2
КОРЕНЬ X=-1
ФУНКЦИЯ F=0
JRUN
МЕТОД НЬЮТОНА ДЛЯ НЕЛИНЕЙНОГО УРАВНЕНИЯ
НАЧАЛЬНОЕ ПРИБЛИЖЕНИЕ X1=1
КОРЕНЬ X=1
ФУНКЦИЯ F=0
JRUN
МЕТОД НЬЮТОНА ДЛЯ НЕЛИНЕЙНОГО УРАВНЕНИЯ
НАЧАЛЬНОЕ ПРИБЛИЖЕНИЕ X1=3
КОРЕНЬ X=2
ФУНКЦИЯ F=0
```

2.5. Нахождение корней уравнения модифицированным методом Ньютона (метод хорд)

Если вычисление производной в методе Ньютона затруднено, можно заменить ее вычисление оценкой:

$$F'(x)\approx\frac{F(x+h)-F(x)}{h}. \qquad (1)$$

Структура программы такая же, как в п. 2.4.

Y — значение функции в точке $x+h$, значение поправки к корню.

В контрольном примере найдены корни уравнения $x^3-2x^2-x+2=0$.

```
JLIST
  10 PRINT "МЕТОД НЬЮТОНА ДЛЯ НЕЛИНЕЙНОГО УРАВНЕНИЯ."
  20 INPUT "НАЧАЛЬНОЕ ПРИБЛИЖЕНИЕ X1="; X1
  30 E1=1E-8:H=1E-4
 100 X=X1: GOSUB 500:Y=F
 110 X=X1+H: GOSUB 500
 120 Y=Y*H/(F-Y):X1=X1-Y
 130 IF ABS (Y)>E1 THEN 100
 140 X=X1: GOSUB 500
 150 GOTO 1000
 500 F=((X-2)*X-1)*X+2
 510 RETURN
1000 PRINT "КОРЕНЬ X="; X1
1010 PRINT "ПРИ ЭТОМ F="F: END

JRUN
МЕТОД НЬЮТОНА ДЛЯ НЕЛИНЕЙНОГО УРАВНЕНИЯ.
НАЧАЛЬНОЕ ПРИБЛИЖЕНИЕ X1=-2
КОРЕНЬ X=-1
ПРИ ЭТОМ F=0

JRUN
МЕТОД НЬЮТОНА ДЛЯ НЕЛИНЕЙНОГО УРАВНЕНИЯ.
НАЧАЛЬНОЕ ПРИБЛИЖЕНИЕ X1=.5
КОРЕНЬ X=1
ПРИ ЭТОМ F=0

JRUN
МЕТОД НЬЮТОНА ДЛЯ НЕЛИНЕЙНОГО УРАВНЕНИЯ.
НАЧАЛЬНОЕ ПРИБЛИЖЕНИЕ X1=3
КОРЕНЬ X=2
ПРИ ЭТОМ F=0
```

2.6. Решение уравнения методом секущих-хорд

Если на интервале $[a, b]$ непрерывная функция $F(x)$ удовлетворяет условию $F(a)\cdot F(b)<0$, то корень уравнения $F(x)=0$ может быть найден методом хорд:

$$x_n=x_{n-1}-F(x_{n-1})\cdot(x_{n-1}-x_{n-2})/(F(x_{n-1})-F(x_{n-2})). \quad (1)$$

Погрешность значения корня оценивается по формуле

$$|x_n-x_{n-1}|<\varepsilon_1. \quad (2)$$

В программе используются переменные: A, B — интервал локализации корня; $F1$, $F2$ — значения функции $F(x)$ в точках a и b; X, F аргумент и значение функции $F(x)$; $E1$ — точность значения корня.

Структура программы:

10—40 — ввод a, b, ε_1;

100—120 — вычисление функций в точках a и b, проверка условия локализации корня и равенства нулю функции F;

130—150 — реализация итераций (1);

500 — вычисление $F(x)$;

1000 — вывод значения корня и функции $F(x)$.

```
JLIST
  10 PRINT "МЕТОД СЕКУЩИХ — ХОРД"
  20 PRINT "ВВЕДИТЕ ИНТЕРВАЛ ЛОКАЛИЗАЦИИ КОРНЯ:"
  30 INPUT "(A, B) ="; A, B
  40 E1 = 1E - 8
 100 X = B: GOSUB 500: F2 = F: IF F = 0 THEN 1000
 110 X = A: GOSUB 500: F1 = F: IF F = 0 THEN 1000
 120 IF F * F2 > 0 THEN PRINT "НАДО ИЗМЕНИТЬ ИНТЕРВАЛ!": GOTO 20
 130 X = A - F1 * (A - B) / (F1 - F2): GOSUB 500
 140 IF ABS (X - A) < E1 THEN 1000
 150 B = A: A = X: F2 = F1: F1 = F: GOTO 130
 500 F = ((X - 2) * X - 1) * X + 2: RETURN
1000 PRINT "КОРЕНЬ X="; X: PRINT "ФУНКЦИЯ F="F
1010 END

JRUN
МЕТОД СЕКУЩИХ — ХОРД
ВВЕДИТЕ ИНТЕРВАЛ ЛОКАЛИЗАЦИИ КОРНЯ:
(A, B) = -2, 0
КОРЕНЬ X = -1
ФУНКЦИЯ F = 1.39698386E - 09
```

2.7. Решение системы уравнений

Решение системы трансцендентных уравнений

$$\begin{cases} x_1 = F_1(x_1, x_2, \ldots, x_n), \\ x_2 = F_2(x_1, x_2, \ldots, x_n), \\ \ldots\ldots\ldots\ldots \\ x_n = F_n(x_1, x_2, \ldots, x_n) \end{cases} \tag{1}$$

производится по итерационным формулам модифицированного метода Зейделя [9]:

$$\begin{aligned} x_1^{(k+1)} &= F_1(x_1^{(k)}, x_2^{(k)}, \ldots, x_n^{(k)}), \\ x_2^{(k+1)} &= F_2(x_1^{(k+1)}, x_2^{(k)}, \ldots, x_n^{(k)}), \\ &\ldots\ldots\ldots\ldots\ldots\ldots \\ x_n^{(k+1)} &= F_n(x_1^{(k+1)}, \ldots, x_{n-1}^{(k+1)}, x_n^{(k)}). \end{aligned} \tag{2}$$

Оценка точности имеет вид:

$$\max |x_i^{(k+1)} - x_i^{(k)}| < \varepsilon_1, \quad i = \overline{1, n}. \tag{3}$$

Итерационный процесс (2) сходится при условии

$$\sum_{i,\ k=1}^{n} |\partial F_i / \partial x_k| < 1.$$

В программе используются переменные: N — количество переменных; $X(1), \ldots, X(N)$ — начальное приближение при вводе и значении корня на k-м шаге; $E1$ — точность значения корней; $Y(1), \ldots, Y(N)$ — значения корня на $(k+1)$ — м шаге; K — количество переменных, удовлетворяющих условию $|x_i^{(k+1)} - x_i^{(k)}| < \varepsilon_1$; I — рабочая переменная цикла.

Обращение к подпрограммам вычисления $F_1, F_2, \ldots, F_n$ производится в строке 120. Подпрограммы для $F_1, F_2, \ldots$ размещены в строках 500, 520, ...

В контрольном примере решается система

$$\begin{cases} x_1 = e^{x_1} - x_2 - 1, \\ x_2 = x_1 / 2. \end{cases}$$

```
JLIST
  10 N=2: DIM X (N), Y (N):E1=1E-8
  20 PRINT "НАЧАЛЬНОЕ ПРИБЛИЖЕНИЕ КОРНЕЙ:"
  30 FOR I=1 TO N: PRINT "X("I;: INPUT ")=";X (I): NEXT
 100 K=0
 110 FOR I=1 TO N
 120 ON I GOSUB 500, 520
 130 IF ABS (Y (I)-X (I))<E1 THEN K=K+1
 140 X (I)=Y (I): NEXT I
 150 IF K<N THEN 100
 160 GOTO 1000
 500 Y (1)=EXP (X (1))-X (2)-1: RETURN
 520 Y (2)=X (1) / 2: RETURN
1000 PRINT "СИСТЕМА ИМЕЕТ СЛЕДУЮЩЕЕ РЕШЕНИЕ:"
1010 FOR I=1 TO N: PRINT "X ("I")="X (I): NEXT : END
JRUN
НАЧАЛЬНОЕ ПРИБЛИЖЕНИЕ КОРНЕЙ:
X (1)=.5
X (2)=0.5
СИСТЕМА ИМЕЕТ СЛЕДУЮЩЕЕ РЕШЕНИЕ:
X (1)=6.51925802E-09
X (2)=3.25962901E-09
```

2.8. Разложение полинома на рациональные линейные множители

По программе находятся все рациональные линейные множители $(U_i x + V_i)$ многочлена

$$p(x) = a_0 x^n + a_1 x^{n-1} + \ldots + a_n = (U_1 x + V_1) \ldots (U_r x + V_r) \times \times (d_0 x^{n-r} + \ldots + d_{n-r}), \qquad (1)$$

где коэффициенты a_k — целые, а полином с коэффициентами d_k линейных рациональных множителей не содержит.

В программе находятся все делители p и q чисел a_0 и a_n соответственно. Затем при условии, что $px-q$ является множителем многочлена (1), $p(x)$ делится по алгоритму Евклида на $px-q$. Получается новый полином, для которого повторяется вся процедура до тех пор, пока не будет получен полином, не имеющий рациональных линейных множителей.

В программе используются переменные: $N1$ — степень исходного полинома; N — степень исходного, затем конечного полинома; $B(N)$ — коэффициенты исходного полинома; $A(N)$ — коэффициенты вначале исходного полинома, затем полинома после выделения множителей; $A0=|a_0|$; R — количество линейных множителей (r); $U(R)$, $V(R)$ — коэффициенты u_i, v_i $(i=\overline{1, r})$; P, $Q-p$, q; $C=$НОД $(a_0 \ldots a_n)$; F — значение $a_0p^n+a_1p^{n-1}q+\ldots+a_nq^n$; L — переменная цикла $(L=Q)$; $N2=N1-R$.

Структура программы:

10—30 — ввод степени полинома n и коэффициентов a_i;

100 — исходные значения количества линейных множителей ($R=0$) и НОД коэффициентов a_i ($c=1$);

110—120 — отыскание множителей вида $(x-0)$. Если такие есть, то степень полинома понижается ($N=N-1$), а количество множителей увеличивается ($R=R+1$);

130 — открытие цикла по p;

140 — проверка того, является ли p множителем a_n;

150 — открытие цикла по L $(L=q)$;

160—170 — отыскание $q\neq 1$, являющегося множителем a_n;

180—210 — определение того, является ли q общим множителем для всех a_i;

220 — если q — общий множитель, то сокращение на него всех коэффициентов a_i;

230 — C — общий множитель всех a_i, повторная проверка того, является ли q множителем числа a_n;

240—250 — вычисление $F=a_0p^n+a_1p^{n-1}q+\ldots+a_nq^n$;

260 — если $F=0$, т. е. $(px-q)$ — множитель, то деление на него исходного полинома;

270—340 — деление полинома на $px-q$ по алгоритму Евклида;

360—370 — закрытие циклов по L, p;

1000—1190 — вывод начального полинома и множителей, на которые он разлагается.

Программа получена переводом на язык Бейсик программы 75б [5].

В контрольном примере показано:

$$3x^6-25x^5+52x^4+9x^3+38x^2-40x=$$
$$=x(x-4)(x-5)(3x-2)(x^2+x+1).$$

```
JLIST
  10 PRINT "МНОЖИТЕЛИ ПОЛИНОМА СТЕПЕНИ N."
  20 INPUT "N=";N: DIM A (N), B (N), U (N), V (N)
  30 FOR I=0 TO N: PRINT "A ("I;: INPUT ") =";A (I):B (I) =A (I) : NEXT
```

```
100 A0=ABS (A (0)):N1=N:R=0:C=1
110 IF A (N)<>0 THEN 130
120 N=N-1:R=R+1:U (R)=1:V (R)=0: GOTO 110
130 FOR P=1 TO A0
140 IF INT (A (0) / P) * P<>A (0) THEN 370
150 FOR L=1 TO ABS (A (N)):Q=L
160 IF Q=1 THEN 240
170 IF INT (A (N) / Q) * Q<>A (N) THEN 240
180 I=0
190 IF I>N THEN 220
200 IF INT (A (I) / Q) * Q<>A (I) THEN 240
210 I=I+1:GOTO 190
220 FOR I=0 TO N:A (I)=A (I) / Q: NEXT
230 C=C * Q: GOTO 170
240 F=A (0):G=1
250 FOR I=1 TO N:G=G * P:F=F * Q+A (I) * G: NEXT
260 IF F<>0 THEN 340
270 R=R+1:U (R)=P:V (R)=Q
280 FOR I=0 TO N
290 F= (A (I)+F) / P:A (I)=F:F=F * Q
300 NEXT
310 N=N-1
320 IF N=0 THEN 380
330 GOTO 240
340 Q=-Q
350 IF Q<0 THEN 240
360 NEXT L
370 NEXT P
380 IF N=0 THEN C=C * A (0)
1000 PRINT "ПОЛИНОМ-"
1010 FOR I=0 TO N1
1020 IF B (I)=0 THEN 1070
1030 IF B (I)>=0 THEN PRINT "+";
1040 PRINT B (I);: IF I=N1 THEN 1070
1050 IF I=N1-1 THEN PRINT "X";
1060 IF I<>N1-1 THEN PRINT "X^" N1-I;
1070 NEXT : PRINT "РАЗЛАГАЕТСЯ НА:"
1075 IF R=0 THEN PRINT "МНОЖИТЕЛЕЙ НЕТ.": END
1080 FOR I=1 TO R: PRINT "("U (I)"X";: IF V (I)<=0 THEN PRINT "+";
1090 PRINT-V (I)")";: NEXT
1100 N=N1-R: IF N=0 THEN END
1105 PRINT "(";
1110 FOR I=0 TO N; IF I=0 AND A (0)<>1 THEN PRINT A (0);
1120 IF I=0 THEN PRINT "X^"N;
1130 IF I=0 THEN 1190
1140 IF A (I)=0 THEN 1190
1150 IF A (I)>=0 THEN PRINT "+";
1160 PRINT A (I);: IF N-I=1 THEN PRINT "X";
1170 IF I=N THEN PRINT ")"
```

```
1180 IF I<>N AND I<>N-1 THEN PRINT "X^" N-I;
1190 NEXT : END

JRUN
МНОЖИТЕЛИ ПОЛИНОМА СТЕПЕНИ N.
N=6
A (0) =3
A (1) = -26
A (2) =52
A (3) =9
A (4) =38
A (5) = -40
A (6) =0
ПОЛИНОМ-
+3X^6-26X^5+52X^4+9X^3+38X^2-40X РАЗЛАГАЕТСЯ НА:
(1X+0) (1X-4) (1X-5) (3X-2) (X^2+1X+1)
```

ИНТЕРПОЛЯЦИЯ ФУНКЦИЙ

2.9. Полином Лагранжа по Эйтекену

Пусть на отрезке $[a, b]$ дано $n+1$ значение функции $F_i=F(x_i)$ в $n+1$ различных точках: $x_0, x_1, \ldots, x_n$. Тогда полином степени не выше n, имеющий в заданных узлах значения F_i, есть полином Лагранжа [9]:

$$L_n(x)=\sum_{i=0}^{n} F_i \frac{(x-x_0)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_n)}{(x_i-x_0)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_n)}. \qquad (1)$$

Вычисление значения $L_n(x)$ по (1) осуществляется по итерационной схеме Эйтекена. Начальные значения F_i теряются. Интервалы $[x_i, x_{i+1}]$ могут быть неравными.

В программе используются переменные: N — количество интервалов; $X(N)$, $F(N)$ — массивы значений аргумента и функции; x, $F1$ — аргумент и значение полинома $L_n(x)$; J, I — переменные циклов.

Структура программы:

10—60 — ввод n, x_i, F_i, x;

100—130 — вычисление $L_n(x)$;

1000 — вывод значения $L_n(x)$.

В контрольном примере при $n=3$, $x_0=1$, $x_1=1.25$, $x_2=1.6$, $x_3=2$, $y(x_i)=1/x_i$ найдено значение $L_3(1.5)=0.665625$.

Программа получена переводом на язык Бейсик Алгол-программы 70б [5].

```
JLIST
10 PRINT "ПО ЗНАЧЕНИЮ F В X0, ... XN ВЫЧИСЛЯЕТСЯ ПОЛИНОМ
   ЛАГРАНЖА L (X)."
20 INPUT "N=";N: DIM X (N), F (N)
30 FOR I=0 TO N
```

```
  40 PRINT "[X("I"), F ("I;: INPUT ")]=";X (I), F (I)
  50 NEXT
  60 INPUT "АРГУМЕНТ X=";X
 100 FOR J=0 TO N-1: FOR I=J+1 TO N
 110 F (I) = ((X-X (J)) * F (I) - (X-X (I)) * F (J)) / (X (I)-X (J))
 120 NEXT I: NEXT J
 130 F1=F (N)
1000 PRINT "L"N" ("X") ="F1
1010 END

JRUN
ПО ЗНАЧЕНИЮ F В X0, ... XN ВЫЧИСЛЯЕТСЯ ПОЛИНОМ ЛАГРАНЖА
     L (X).
N=3
[X (0), F (0)]=1,1
[X (1), F (1)]=1.25,.8
[X (2), F (2)]=1.6,.625
[X (3), F (3)]=2,0.5
АРГУМЕНТ X=1.5
L3 (1.5) =.665625
```

2.10. Рациональная интерполяция с помощью непрерывных дробей

Если задано m значений функции $F_i=F(x_i)$ в m точках $x_1, \ldots, x_m$, то значение $F(x)$ может быть определено по формуле (см. [10])

$$F(x)=a_1+\cfrac{x-x_1}{a_2+\cfrac{x-x_2}{a_3+\ldots}}\cdot$$

$$\ldots+\cfrac{x-x_{m-1}}{a_m}\,. \qquad (1)$$

Непрерывная дробь (1) может быть представлена в виде рациональной функции:

$$\frac{b_0+b_1x+\ldots+b_lx^l}{d_0+d_1x+\ldots+d_lx^l}\,, \qquad (2)$$

где $l\leqslant m_1=\text{int}\,(m/2)$.

В программе вычисляются коэффициенты a_i, b_j, d_j $(i=\overline{1,\ m}$, $j=\overline{0,\ m_1})$.

В программе используются переменные: M — количество точек; $X(M)$, $Y(M)$ — координаты точек x_i, y_i; $A(M)$, $B(M1)$, $D(M1)$ соответственно a_i, b_j, d_j; $M1=\text{int}\,(M/2)$; K, R, S, $P(M1)$, $Q(M1)$ — рабочие переменные и массивы; I, J — переменные циклов.

Структура программы:

10—40 — ввод m, x_i, y_i $(i=\overline{1,\ m})$; $m_1=\text{int}\,(m/2)$;

100—250 — вычисление a_i, b_j, d_j;

1000—1020 — вывод a_i, b_j, d_j.

Программа получена переводом на язык Бейсик Алгол-программы 18б [4].

В контрольном примере для пяти точек найдены коэффициенты a_i, b_j, d_j: $x_1=0$, $y_1=3$; $x_2=1$, $y_2=2{,}5$; $x_3=2$, $y_3=1{,}6666...$; $x_4=3$, $y_4=2.25$; $x_5=7$, $y_5=2.125$. Эти коэффициенты соответствуют коэффициентам функции

$$y_i=\frac{-6-x_i+2x_i^2}{-2-x_i+x_i^2}.$$

```
JLIST
  10 PRINT "РАЦИОНАЛЬНАЯ ИНТЕРПОЛЯЦИЯ"
  20 INPUT "КОЛ. ТОЧЕК M=";  M: DIM X(M), Y (M), A (M)
  30 M1=INT (M/2): DIM P (M1), Q (M1), B (M1), D (M1)
  40 FOR I=1 TO M: PRINT "X ("I"), Y ("I;: INPUT")=";  X (I), Y (I): NEXT
 100 FOR J=1 TO M
 110 R=Y (J):S=X (J):IF J=1 THEN 130
 120 FOR I=1 TO J-1:R=(S-X (I))/(R-A (I)):NEXT
 130 A (J)=R: NEXT J
 140 P (0)=1:Q (0)=A (1):K=1
 150 FOR I=1TO M1:P (I)=0:Q (I)=0:NEXT
 160 FOR I=2 TO M
 170 FOR J=INT (I/2) TO 1 STEP-1
 180 R=A (I) * Q (J)-X (I-1) * P (J)+P (J-1):P (J)=Q (J):Q (J)=R
 190 NEXT
 200 R=A (I) * Q (0)-X (I-1) * P (0):P (0)=Q (0):Q (0)=R
 210 NEXT I
 220 IF K=2 THEN GOTO 250
 230 FOR I=0 TO M1:B (I)=Q (I):NEXT
 240 K=2:P (0)=0:Q (0)=1:GOTO 150
 250 FOR I=0 TO M1:D (I)=Q (I):NEXT
1000 PRINT "КОЭФФИЦИЕНТЫ:"
1010 FOR I=1 TO M: PRINT "A ("I")="A (I): NEXT
1020 PRINT "КОЭФФИЦИЕНТЫ B (I):"
1030 FOR I=0 TO M1: PRINT "B ("I")="B (I): NEXT
1040 PRINT "КОЭФФИЦИЕНТЫ D (I):"
1050 FOR I=0 TO M1: PRINT "D ("I")="D (I): NEXT
1060 END

IRUN
РАЦИОНАЛЬНАЯ ИНТЕРПОЛЯЦИЯ
КОЛ. ТОЧЕК M=5
X (1), Y (1)=0,3
X (2), Y (2)=1,2.5
X (3), Y (3)=2,1.666666666
X (4), Y (4)=3,2.25
X (5), Y (5)=7,2.125
КОЭФФИЦИЕНТЫ:
A (1)=3
A (2)=-2
```

```
A (3) = 2
A (4) = -.333333333
A (5) = -3
КОЭФФИЦИЕНТЫ B (I):
B (0) = -6.00000002
B (1) = -.999999985
B (2) = 2
КОЭФФИЦИЕНТЫ D (I):
D (0) = -2.00000001
D (1) = -.999999998
D (2) = 1
```

2.11. Интерполяция по Ньютону

Пусть на отрезке $[a, b]$ задано $n+1$ значение аргумента: $x_0=a$, $x_1, \ldots, x_n=b$ и $n+1$ значение функции $y_i=y(x_i)$. Тогда интерполяционный многочлен степени n в форме Ньютона может быть построен по следующим формулам [17]:

$$L_n(x)=\sum_{i=0}^{n} C_i N_i(x), \tag{1}$$

где

$$\begin{gathered} N_0(x)=1, \\ N_i(x)=(x-x_0)(x-x_1)\ldots(x-x_{i-1}) \ (i=\overline{1,\ n}), \\ C_i=[x_0, x_1, \ldots, x_i], \\ [x_i]=y_i \ (i=\overline{0,\ n}), \\ [x_0, x_1, \ldots, x_i]=\frac{1}{x_i-x_0}([x_1, \ldots, x_i]-[x_0, \ldots, x_{i-1}]). \end{gathered} \tag{2}$$

Выражение $[x_0, x_1, \ldots, x_i]$ называется разделенной разностью.

В программе используются переменные: X — аргумент; N — степень полинома; $X(N)$, $Y(N)$ — массивы x_i, y_i соответственно $(i=\overline{0,n})$ (разделенные разности затем помещаются в массив Y); I, K — переменные циклов; L — значение полинома в точке x; S, $I1$ — рабочие переменные.

Структура программы:

10 — 40 — ввод n, x, x_i, y_i, $(i=\overline{0,\ n})$;

100 — 180 — вычисление $L_n(x)$;

1000 — 1010 — вывод значения L.

Контрольный пример тот же, что в п. 2.9.

```
LIST
 10 PRINT "ИНТЕРПОЛЯЦИЯ ПО НЬЮТОНУ."
 20 INPUT "СТЕПЕНЬ ПОЛИНОМА N=": N
 30 INPUT "АРГУМЕНТ X=": X
 40 FOR I=0 TO N: PRINT "X ("I"), Y ("I";: INPUT") =": X (I), Y (I): NEXT
100 L=Y (0):S=1
110 FOR I=N TO 1 STEP-1
```

```
 120 I1=N-I
 130 FOR K=0 TO I-1
 140 Y (K)=(Y (K+1)-Y (K))/(X (K+1+I1)-X (K))
 150 NEXT
 160 S=S * (X-X (I1)):L1=L
 170 L=L+Y (0) * S
 180 NEXT I
1000 PRINT "L ("X") ="L
1010 END

JRUN
ИНТЕРПОЛЯЦИЯ ПО НЬЮТОНУ.
СТЕПЕНЬ ПОЛИНОМА N=3
АРГУМЕНТ X=1.5
X (0), Y (0) =1,1
X (1), Y (1) =1.25,.8
X (2), Y (2) =1.6,.625
X (3), Y (3) =2,.5
L (1.5) =.665625
```

ОПЕРАЦИИ НАД ПОЛИНОМАМИ И СТЕПЕННЫМИ РЯДАМИ

2.12. Умножение рядов

Программа находит $n+1$ коэффициент произведения двух степенных рядов:

$$\begin{aligned} g(x) &= g_0 + g_1 x + \ldots + g_n x^n + \ldots, \\ h(x) &= h_0 + h_1 x + \ldots + h_n x^n + \ldots, \\ C(x) &= g(x) \cdot h(x) = C_0 + C_1 x + \ldots + C_n x^n + \ldots . \end{aligned} \tag{1}$$

причем (см. [10])

$$C_j = \sum_{i=0}^{j} g_i h_{j-i}. \tag{2}$$

Программа по известным значениям g_i и h_i вычисляет по формуле (2) значения C_i $(i=\overline{0, n})$.

В программе используются переменные: N — номер последнего нужного коэффициента C_n; $G(N)$, $H(N)$, $C(N)$ — массивы, содержащие коэффициенты g_i, h_i, c_i соответственно; I, J — переменные циклов в (2).

Структура программы:

10—80 — ввод n и коэффициентов g_i, h_i $(i=\overline{0, n})$;

100—140 — реализация вычислений по формуле (2);

1000—1010 — вывод коэффициентов C_i $(i=\overline{0, n})$.

В контрольном примере найдено произведение рядов e^x и e^{-2x} при $n=4$:

$$e^x = 1 + x + \frac{1}{2}x^2 + \frac{1}{6}x^3 + \frac{1}{24}x^4 + \ldots,$$

$$e^{-2x}=1-2x+2x^2-\frac{4}{3}x^3+\frac{2}{3}x^4+\ldots,$$

$$e^{-x}=1-x+\frac{1}{2}x^2-\frac{1}{6}x^3+\frac{1}{24}x^4+\ldots\ .$$

```
JLIST
  10 PRINT "ПЕРВЫЕ N+1 КОЭФФИЦИЕНТА ПРОИЗВЕДЕНИЯ ДВУХ
     СТЕПЕННЫХ РЯДОВ:"
  20 PRINT "C (X) =H (X) G (X), ГДЕ"
  30 PRINT "G (X) =G (0) +G (1) * X+...G (N) * X^N"
  40 PRINT "H (X) =H (0) +H (1) * X+...H (N) * X^N"
  50 INPUT "N="; N: DIM G (N), H (N), C (N)
  60 FOR I=0 TO N: PRINT "G ("I;: INPUT") ="; G (I): NEXT
  70 PRINT "ВТОРОЙ РЯД:"
  80 FOR I=0 TO N: PRINT "H ("I;: INPUT") ="; H (I): NEXT
 100 FOR J=0 TO N
 110 C (J) =0
 120 FOR I=0 TO J
 130 C (J) =C (J) +G (I) * H (J-I)
 140 NEXT I: NEXT J
1000 PRINT "ПРОИЗВЕДЕНИЕ РАВНО:"
1010 FOR I=0 TO N: PRINT "C ("I") ="; C (I): NEXT
1020 END

JRUN
ПЕРВЫЕ N+1 КОЭФФИЦИЕНТА ПРОИЗВЕДЕНИЯ ДВУХ СТЕПЕННЫХ
РЯДОВ:
C (X) =H (X) G (X), ГДЕ
G (X) =G (0) +G (1) * X+...G (N) * X^N
H (X) =H (0) +H (1) * X+... H (N) * X^N
N=4
G (0) =1
G (1) =1
G (2) =.5
G (3) =.166666666
G (4) =.041666666
ВТОРОЙ РЯД:
H (0) =1
H (1) =-2
H (2) =2
H (3) =-1.33333333
H (4) =.6666666666
ПРОИЗВЕДЕНИЕ РАВНО:
C (0) =1
C (1) =-1
C (2) =.5
C (3) =-.166666664
C (4) =.0416666696
```

2.13. Деление рядов

Пусть заданы два ряда

$$h(x)=h_0+h_1x+\ldots+h_nx^n+\ldots,$$
$$g(x)=g_0+g_1x+\ldots+g_nx^n+\ldots\ . \qquad (1)$$

Тогда коэффициенты частного от деления этих рядов

$$c(x)=h(x)/g(x)=c_0+c_1x+\ldots+c_nx^n+\ldots \qquad (2)$$

могут быть определены из соотношений [10]:

$$c_i=[h_i-\sum_{j=1}^{i} c_jg_{i-j}]/g_0,$$
$$c_0=h_0/g_0,\ i=1, 2, \ldots\ . \qquad (3)$$

В программе коэффициенты c_i записываются в массив коэффициентов h_i.

В программе используются переменные: N — количество слагаемых в рядах равно $N+1$; $H(N)$, $G(N)$ — массивы коэффициентов h_i, g_i; I, J — переменные циклов в (3).

Структура программы:

10 — 50 — ввод n, h_i, g_i;

100 — 150 — вычисления по формуле (3);

1000 — 1010 — вывод c_i (массив $H(N)$).

Программа получена переводом на язык Бейсик Алгол-программы 131б [6].

В контрольном примере найдены 4 слагаемых от деления ряда e^x на e^{2x} (п. 2.12).

```
JLIST
 10 PRINT "ПРОГРАММА ВЫЧИСЛЯЕТ ПЕРВЫЕ N+1 КОЭФФИЦИЕН-
    ТОВ": PRINT "ЧАСТНОГО ОТ ДЕЛЕНИЯ ДВУХ СТЕПЕННЫХ РЯДОВ:"
 20 PRINT "H (X)/G (X)"
 30 INPUT "N=";N: DIM G (N), H (N)
 40 FOR I=0 TO N: PRINT "H ("I;: INPUT") =";H (I): NEXT
 50 FOR I=0 TO N: PRINT "G ("I;: INPUT") =";G (I): NEXT
100 A=1/G (0)
110 FOR J=0 TO N:B=A*H (J)
120 IF J+1>N THEN 140
130 FOR I=J+1 TO N:H (I)=H (I)-B*G (I-J): NEXT I
140 NEXT J
150 FOR J=0 TO N:H (J)=H (J)*A: NEXT
1000 PRINT "КОЭФФИЦИЕНТЫ ЧАСТНОГО РАВНЫ:"
1010 FOR I=0 TO N: PRINT "H ("I")="H (I): NEXT
1020 END
JRUN
ПРОГРАММА ВЫЧИСЛЯЕТ ПЕРВЫЕ N+1 КОЭФФИЦИЕНТОВ ЧАСТНОГО
ОТ ДЕЛЕНИЯ ДВУХ СТЕПЕННЫХ РЯДОВ:
H (X)/G (X)
N=3
H (0)=1
```

H (1) = 1
H (2) = .5
H (3) = .166666666
G (0) = 1
G (1) = 2
G (2) = 2
G (3) = 1.333333333
КОЭФФИЦИЕНТЫ ЧАСТНОГО РАВНЫ:
H (0) = 1
H (1) = −1
H (2) = .5
H (3) = −.166666667

2.14. Возведение ряда в степень

Пусть задан ряд

$$f(x)=1+a_1x=+\ldots+a_nx^n+\ldots. \qquad (1)$$

Коэффициенты p-й степени ряда $f(x)$ можно найти по рекуррентному соотношению [10]:

$$b_i=pa_i+\frac{1}{i}\sum_{k=1}^{i=1}[p(i-k)-k]\,a_{i-k}b_k, \qquad (2)$$

где

$$i=1, 2, 3, \ldots;\ f^p(x)=1+b_1x+b_2x^2+\ldots.$$

Соотношение (2) справедливо для любого вещественного значения $p\neq 0$. Если $p=0$, то получим коэффициенты ряда $\ln f(x)$. Если в (1) $a_0\neq 1$, то необходимо предварительно разделить все a_i на a_0 и в конце умножить b_i на a_0^p.

В программе используются переменные: N — количество коэффициентов, которые необходимо вычислить; P — показатель степени; $A(N)$ — массив коэффициентов исходного ряда; $B(N)$ — массив коэффициентов ряда $f^p(x)$; Z равно p при $p\neq 0$ и 1 в противном случае; J, K — параметры циклов; S — рабочая переменная.

Структура программы:

10 — 50 — ввод n, p, a_i;

100 — 170 — вычисление по формуле (2);

1000 — 1020 — вывод b_i $(i=\overline{1, n})$.

Программа получена переводом на язык Бейсик Алгол-программы 134б [6].

В контрольных примерах при $n=4$ найдено: $f(x)=1+x$, при $p=0.5$ $f^p(x)=\sqrt{1+x}=1+\frac{1}{2}x-\frac{1}{8}x^2+\frac{1}{16}x^3-\frac{5}{128}x^4+\ldots$, при $p=0$ $f^p(x)=\ln(1+x)=x-\frac{1}{2}x^2+\frac{1}{3}x^3-\frac{1}{4}x^4+\ldots$.

```
JLIST
  10 PRINT "ПРОГРАММА ВЫЧИСЛЯЕТ ПЕРВЫЕ N КОЭФФИЦИЕНТОВ":
     PRINT "P — ТОЙ СТЕПЕНИ РЯДА:": PRINT "F=1+A (1) * X+A (2) *
     * X^2+A (3) * X^3+ ... ": PRINT "ЕСЛИ P=0 ТО ПОЛУЧИМ РЯД
     LN (F)."
  20 INPUT "КОЛИЧЕСТВО СЛАГАЕМЫХ N=";N: INPUT "ПОКАЗАТЕЛЬ
     СТЕПЕНИ P="; P: DIM A (N)
  30 FOR I=1 TO N
  40 PRINT "A ("; I;: INPUT") ="; A (I)
  50 NEXT
 100 Z=P: IF P=0 THEN Z=1
 110 B (1)=Z * A (1): IF N<2 THEN 1000
 120 FOR I=2 TO N:S=0
 130 FOR K=1 TO I-1
 140 S=S+(P * (I-K)-K) * B (K) * A (I-K)
 150 NEXT
 160 B (I)=Z * A (I)+S/I
 170 NEXT
1000 PRINT "КОЭФФИЦИЕНТЫ РАВНЫ:"
1010 FOR I=1 TO N: PRINT B (I)" X^"I: NEXT
1020 END

JRUN
ПРОГРАММА ВЫЧИСЛЯЕТ ПЕРВЫЕ N КОЭФФИЦИЕНТОВ
P—ТОЙ СТЕПЕНИ РЯДА:
F=1+A (1) * X+A (2) * X^2+A (3) * X^3+ ...
ЕСЛИ P=0 ТО ПОЛУЧИМ РЯД LN (F).
КОЛИЧЕСТВО СЛАГАЕМЫХ N=4
ПОКАЗАТЕЛЬ СТЕПЕНИ P=.5
A (1) =1
A (2) =0
A (3) =0
A (4) =0
КОЭФФИЦИЕНТЫ РАВНЫ:
.5X^1
-.125 X^2
.0625 X^3
-.0390625 X^4

JRUN
ПРОГРАММА ВЫЧИСЛЯЕТ ПЕРВЫЕ N КОЭФФИЦИЕНТОВ P—ТОЙ СТЕ-
ПЕНИ РЯДА:
F=1+A (1) * X+A (2) * X^2+A (3) * X^3+ ...
ЕСЛИ P=0 ТО ПОЛУЧИМ РЯД LN (F).
КОЛИЧЕСТВО СЛАГАЕМЫХ N=4
ПОКАЗАТЕЛЬ СТЕПЕНИ P=0
A (1) =1
A (2) =0
A (3) =0
A (4) =0
```

КОЭФФИЦИЕНТЫ РАВНЫ:
1 X^1
-.5 X^2
.333333333 X^3
-.25 X^4

2.15. Обращение ряда

Пусть задан степенной ряд

$$y=x+a_2x^2+\ \ldots\ +a_nx^n+\ \ldots\ . \qquad (1)$$

Программа находит коэффициенты обращенного степенного ряда

$$x=y+b_2x^2+\ \ldots\ +b_nx^n+\ \ldots\ . \qquad (2)$$

В программе используются переменные: N — количество слагаемых ряда (1) и ряда (2); $A(N)$, $B(N)$ — массивы коэффициентов исходного и обращенного степенных рядов соответственно; $Q(N)$, $R(N)$ — рабочие массивы; I, J, K — переменные циклов.

Программа получена переводом на язык Бейсик Алгол-программы 1936 [7].

В контрольном примере найдено 6 коэффициентов ($n=6$) обращенного степенного ряда для функции:

$$y=\ln(1+x)=x-\frac{1}{2}x^2+\frac{1}{3}x^3-\frac{1}{4}x^4+\frac{1}{5}x^5-\frac{1}{6}x^6+\ \ldots,$$

$$x=e^y-1=y+\frac{1}{2}y^2+\frac{1}{6}y^3+\frac{1}{24}y^4+\frac{1}{120}y^5+\frac{1}{720}y^6+\ \ldots\ .$$

```
JLIST
 10 PRINT "ПРОГРАММА ВЫЧИСЛЯЕТ N КОЭФФИЦИЕНТОВ ОБРАЩЕН-
    НОГО СТЕПЕННОГО РЯДА (N>1):"
 20 PRINT "Y=X+A (2) X^2+A (3) X^3+ ... +A (N) X^N"
 30 INPUT "N=";N: DIM A (N), B (N), R (N), Q (N)
 50 FOR I=2 TO N: PRINT "A ("I;: INPUT")=";A (I): NEXT
100 A (1)=0:B (0)=0:B (1)=1
110 FOR K=2 TO N:B (K)=0
120 FOR I=0 TO K:R (I)=0: NEXT
130 FOR J=K TO 1 STEP-1
140 Q (0)=R (0)-A (J)
150 FOR I=1 TO K:Q (I)=R (I): NEXT
180 FOR I=0 TO K
190 S=0
200 FOR M=0 TO I:S=S+B (M) * Q (I-M): NEXT M
210 R (I)=S
220 NEXT I: NEXT J
230 FOR I=2 TO K:B (I)=R (I): NEXT
240 NEXT K
1000 PRINT "КОЭФФИЦИЕНТЫ ОБРАЩЕНИЯ СТЕПЕННОГО РЯДА:"
1010 FOR I=1 TO N: PRINT "B ("I")="B (I): NEXT
1020 END
```

```
JRUN
ПРОГРАММА ВЫЧИСЛЯЕТ N КОЭФФИЦИЕНТОВ ОБРАЩЕННОГО СТЕ-
ПЕННОГО РЯДА (N>1):
Y=X+A (2) X^2+A (3) X^3+...+A (N) X^N
N=6
A (2) = -.5
A (3) = .333333333
A (4) = -.25
A (5) = .2
A (6) = -.166666666
КОЭФФИЦИЕНТЫ ОБРАЩЕНИЯ СТЕПЕННОГО РЯДА:
B (1) = 1
B (2) = .5
B (3) = .166666667
B (4) = .0416666674
B (5) = 8.33333434E-03
B (6) = 1.38888914E-03
```

2.16. Коэффициенты полинома при линейном преобразовании аргумента

Если в полиноме n-й степени

$$P(x)=C_0+C_1x+\ \ldots\ +C_nx^n \tag{1}$$

аргумент x заменить на $x=at+b$, то получим полином

$$P(at+b)=\sum_{k=0}^{n} C_k(at+b)^k=\sum_{i=0}^{n} d_it^i, \tag{2}$$

где

$$d_i=\sum_{k=i}^{n} C_k^i a^i b^{k-i} C_k,$$
$$C_k^i=\frac{k(k-1)\ \ldots\ (k-i+1)}{k!}. \tag{3}$$

В программе вычисляются коэффициенты d_i по формуле (3).

В программе используются переменные: N — степень полинома; A, B — коэффициенты a и b; $C(N)$, $D(N)$ — коэффициенты c_i, d_i; $Z(N)$, $W(N)$ — рабочие массивы; I, J — переменные циклов; $K=I-J$.

Структура программы:

10—40 — ввод a, b, c_i;

100—140 — вычисление d_0;

150—230 — вычисление d_1, d_2, ..., d_n;

1000—1010 — вывод коэффициентов c_i, d_i.

Программа получена переводом на язык Бейсик Алгол-программы 29б [4].

В контрольном примере преобразован полином $2x^2+3x+4$. При $x=3t+2$ получен правильный результат $18t^2+33t+18$.

```
JLIST
  10 PRINT "КОЭФФИЦИЕНТЫ ПОЛИНОМА": PRINT "P=C (0)+
     +C (1) X+ ... +C (N) X^N": PRINT "ПОСЛЕ ЗАМЕНЫ АРГУМЕНТА
     X=A*T+B"
  20 INPUT "ПОКАЗАТЕЛЬ СТЕПЕНИ N="; N: INPUT "A="; A: INPUT
     "B="; B
  30 DIM D (N), Z (N), W (N), C (N)
  40 FOR I=0 TO N: PRINT "C ("I;: INPUT") ="; C (I): NEXT
 100 D (0)=C (0):Z (0)=1:W (0)=1
 110 FOR I=1 TO N
 120 W (I)=1:Z (I)=B*Z (I-1)
 130 D (0)=D (0)+C (I)*Z (I)
 140 NEXT
 150 FOR J=1 TO N
 160 W (0)=A*W (0):D (J)=C (J)*W (0)
 170 IF J+1>N THEN 230
 180 FOR I=J+1 TO N
 190 K=I-J
 200 W (K)=A*W (K)+W (K-1)
 210 D (J)=D (J)+C (I)*W (K)*Z (K)
 220 NEXT
 230 NEXT
1000 PRINT "КОЭФФИЦИЕНТЫ:"
1010 FOR I=0 TO N: PRINT "C ("I")="C (I), "D ("I")="D (I): NEXT: END

RUN
КОЭФФИЦИЕНТЫ ПОЛИНОМА
P=C (0)+C (1) X+ ... +C (N) X^N
ПОСЛЕ ЗАМЕНЫ АРГУМЕНТА X=A*T+B
ПОКАЗАТЕЛЬ СТЕПЕНИ N=2
A=3
B=2
C (0)=4
C (1)=3
C (2)=2
КОЭФФИЦИЕНТЫ:
C (0)=4       D (0)=18
C (1)=3       D (1)=33
C (2)=2       D (2)=18
```

СУММИРОВАНИЕ И ВЫЧИСЛЕНИЕ КОЭФФИЦИЕНТОВ РЯДА

2.17. Сумма ряда Фурье

С помощью программы суммируется n слагаемых рядов Фурье:

$$a=\sum_{r=0}^{n-1} x_r \cos(rw),\quad b=\sum_{r=0}^{n-1} x_r \sin(rw). \qquad (1)$$

Вычисление значений $\cos(rw)$ и $\sin(rw)$ производится на основании рекуррентных соотношений:

$$t_0=0;\ t_1=1;\ t_r=\frac{\sin rw}{\sin w}=2\cos wt_{r-1}-t_{r-2}\ (r\geqslant 2);$$
$$\sin(rw)=t_r\sin w;\ \cos(rw)=t_{r+1}-t_r\cos w. \qquad (2)$$

Из выражений (2) получим:

$$b=\sin w\sum_{r=0}^{n-1}x_rt_r,$$
$$a=\sum_{r=0}^{n-1}x_rt_{r+1}-\cos w\,(\sum_{r=0}^{n-1}x_rt_r). \qquad (3)$$

В программе используются переменные: N — количество слагаемых в рядах (1); $X(N-1)$ — массив коэффициентов $x_0, x_1, x_2, \ldots, x_{n-1}$; W — аргумент; A, B — суммы (1); $C2=2\cos W$; $T1$, $T2$ — рабочие переменные t_{r-1}, t_{r-2} в (2); R — переменная цикла.

Структура программы:

10—70 — ввод n, w, x_i;

100—130 — задание начальных значений t_0, t_1, c_2, нахождение максимального значения r, такого, что $x_r\neq 0$ и $r\leqslant n-1$;

140 — если все $x_r=0$, то $a=b=0$;

150 — $t_2=x_r$;

160 — 190 — вычисление сумм $\sum_{i=0}^{r-1}x_it_i$ и $\sum_{i=0}^{r-1}x_it_{i+1}$;

200 — 210 — вычисление a и b по формулам (3);

1000 — 1010 — вывод коэффициентов a, b.

Программа получена переводом на язык Бейсик Алгол-программы 128б [6].

В контрольном примере найдены суммы рядов Фурье (1) при $n=4$, $w=2$:

$$a=1+2\cos 2+3\cos 4+4\cos 6,$$
$$b=2\sin 2+3\sin 4+4\sin 6.$$

```
JLIST
 10 PRINT "ПРОГРАММА СУММИРУЕТ N СЛАГАЕМЫХ РЯДОВ ФУРЬЕ:"
 20 PRINT "A=X (0)+X (1) COS (W)+X (2) COS (2W)+ ... +X (N-1)
    COS ((N-1) W),": PRINT "B=X (1) SIN (W)+X (2) SIN (2W)+
    + ... X (N-1) SIN ((N-1) W)."
 30 INPUT "N=": N: DIM X (N-1)
 40 INPUT "АРГУМЕНТ W=": W
 50 FOR I=0 TO N-1
 60 PRINT "X ("I;: INPUT") =": X (I)
 70 NEXT
100 T2=0: T1=0: C2=2 * COS (W)
110 FOR R=N-1 TO 0 STEP-1
120 IF X (R) < >0 THEN 150
130 NEXT R
```

```
 140 GOTO 200
 150 T2=X (R)
 160 FOR R=R-1 TO 0 STEP-1
 170 T=T2 * C2+X (R) -T1
 180 T1=T2: T2=T
 190 NEXT R
 200 A=T2-T1 * C2/2
 210 B=T1 * SIN (W)
1000 PRINT "A=";A, "B=";B
1010 END

JRUN
ПРОГРАММА СУММИРУЕТ N СЛАГАЕМЫХ РЯДОВ ФУРЬЕ:
A=X (0) +X (1) COS (W) +X (2) COS (2W) + ... +X (N-1) COS ((N-1 W),
B=X (1) SIN (W) +X (2) SIN (2W) + ... X (N-1) SIN ((N-1) W).
N=4

АРГУМЕНТ W=2
X (0) =1
X (1) =2
X (2) =3
X (3) =4
A=2.04745661        B=-1.56947463
```

2.18. Коэффициенты тригонометрического полинома

Пусть на промежутке $[0, 2\pi]$ задано k значений F_i некоторой функции $F(x)$ в равноотстоящих точках $x_i = 2\pi i/k$, где $i = 0, 1, \ldots, k$. В этом случае функцию $F(x)$ целесообразно интерполировать тригонометрическим полиномом [12], [13]:

$$Q_n(x) = \begin{cases} a_0/2 + \sum\limits_{p=1}^{n} (a_p \cos px + b_p \sin px) \text{ при } k = 2n+1, \\ a_0/2 + \sum\limits_{p=1}^{n} (a_p \cos px + b_p \sin px) + \frac{a_n}{2} \cos nx \text{ при } k = 2n, \end{cases} \quad (1)$$

причем $Q_n(x_i) = F_i = F(x_i)$.

В программе используются переменные: K — количество равноотстоящих точек x_i; $p1 = \pi = 3.141592653 \ldots$; $F(K)$ — массив значений F_i; F — вычисляемое значение функции в точке x_i; $R = 2/K$; $C1 = \cos(\pi R)$; $S1 = \sin(\pi R)$; $N - n$; $A(N)$, $B(N)$ — массивы коэффициентов a_i, b_i, I, P — переменные циклов; $U1$, $U2$, $C2$, $S2$ — рабочие переменные; X — аргумент функции $F(x)$.

Структура программы:

10—40 — ввод k, π, вычисление F_i;

100—190 — вычисление коэффициентов a_i, b_i;

500—510 — подпрограмма вычисления значений функции F;

1000 — 1010 — вывод a_i, b_i (напечатано 17 первых коэффициентов).

Программа получена переводом на язык Бейсик Алгол-программы 157б [7].

В контрольном примере найдены коэффициенты a_i, b_i для функции

$$F(x)=\begin{cases} x^2 & \text{при } x\leqslant\pi, \\ (2\pi-x)^2 & \text{при } x>\pi. \end{cases}$$

Ряд Фурье для этой функции имеет вид:

$$F(x)=\frac{\pi^2}{3}-\frac{4}{1^2}\cos x+\frac{4}{2^2}\cos 2x-\frac{4}{3^2}\cos 3x+\ \ldots,$$

$$a_0=2\pi^2/3=6.57973627\ \ldots\ .$$

```
JLIST
  10 PRINT "КОЭФФИЦИЕНТЫ РЯДА ФУРЬЕ НА ИНТЕРВАЛЕ [0, 6,28 ...]."
  20 INPUT "КОЛИЧЕСТВО СЛАГАЕМЫХ K=";K
  30 P1=3.14159265359: DIM F (K)
  40 FOR I=0 TO K:X=2 * P1/K * I: GOSUB 500:F (I) =F: NEXT
 100 R=2/K:C1=COS (R * P1):S1=SIN (R * P1)
 110 C2=1: S2=0:N=INT (K/2): DIM A (N), B (N)
 120 FOR P=0 TO N:
 130 U1=0: U2=0
 140 FOR I=K-1 TO 1 STEP-1
 150 U0=F (I) +2 * C2 * U1-U2: U2=U1: U1=U0
 160 NEXT I
 170 A (P) =R * (F (0) +U1 * C2-U2) :B (P) =R * U1 * S2
 180 T1=C1 * C2-S1 * S2: S2=C1 * S2+S1 * C2: C2=T1
 190 NEXT P
 200 GOTO 1000
 500 F=X * X: IF X>P1 THEN F= (2 * P1-X) ^2
 510 RETURN
1000 FOR I=0 TO N: PRINT "A ("I") ="A (I), "B ("I") ="B (I): NEXT
1010 END

JRUN
КОЭФФИЦИЕНТЫ РЯДА ФУРЬЕ НА ИНТЕРВАЛЕ [0, 6, 28 ...].
КОЛИЧЕСТВО СЛАГАЕМЫХ K=51
A (0) =6.57720671
A (1) = -3.99746696
A (2) =.997456829
A (3) = -.441884331
A (4) =.24741597
A (5) = -.157384894
A (6) =.108457506
A (7) = -.0789328078
A (8) =.0597457801
A (9) = -.0465655149
A (10) =.0371106532
A (11) = -.0300865392
```

A (12) = .0247139172
A (13) = — .0205007552
A (14) = .0171237481
A (15) = — .0143631238
A (16) = .0120650077
B (0) = 0
B (1) = 9.76628389E — 09
B (2) = 2.28053187E — 09
B (3) = — 4.6440881E — 09
B (4) = 1.79696791E — 09
B (5) = — 1.18169622E — 09
B (6) = 1.52551112E — 09
B (7) = — 4.16029966E — 10
B (8) = — 4.56678071E — 11
B (9) = — 3.26935601E — 11
B (10) = — 5.16694716E — 11
B (11) = — 2.18520998E — 10
B (12) = 2.82978039E — 10
B (13) = — 7.86571517E — 10
B (14) = 8.86466223E — 10
B (15) = — 7.02564654E — 11
B (16) = — 4.162174E — 10

2.19. Сумма ряда по Эйлеру

По программе находится сумма ряда $S=\sum_{i=0}^{\infty} F(i)$ с помощью преобразования Эйлера. Суммирование производится до тех пор, пока t_1 раз подряд абсолютные значения слагаемых преобразованного ряда будут меньше чем ε_1, где ε_1 — погрешность суммирования.

В программе используются переменные: $E1$ — погрешность (ε_1); $T1$ — целый параметр (t_1); F — значение $F(i)$; S — сумма ряда; $M1 — F$; $M(K) — F(i)$; T показывает, сколько раз модули слагаемых преобразованных рядов меньше ε_1; D — сумма слагаемых преобразованного ряда; I, K, J — параметры циклов.

Структура программы:

10 — 30 — ввод ε_1, t_1;

100—230— вычисление S;

500 — 520 — подпрограмма вычисления $F(i)$;

1000 — вывод S.

Программа получена переводом на язык Бейсик Алгол-программы 8б [4].

В контрольном примере найдена сумма ряда ($t_1 = 5$):

$$S=1+\frac{1}{2!}+\frac{1}{3!}+\frac{1}{4!}+\ \ldots\ =e=2.718281828\ \ldots.$$

```
JLIST
  10 PRINT "ПРОГРАММА ВЫЧИСЛЯЕТ СУММУ РЯДА ПО ЭЙЛЕРУ."
  20 E1=1E-6
  30 INPUT "ЦЕЛЫЙ ПАРАМЕТР T1=";T1: DIM M (15)
 100 I=0:N=0:T=0:M (0)=F: GOSUB 500
 110 S=M (0)/2
 120 I=I+1:M1=F: GOSUB 500
 130 FOR K=0 TO N
 140 D=(M1+M (K))/2:M (K)=M1
 150 M1=D
 160 NEXT
 170 IF ABS (M1)<ABS (M (N)) AND N<15 THEN D=M1/2:N=N+1:
     :M (N)=M1: GOTO 190
 180 D=M1
 190 S=S+D
 200 IF ABS (D)<E1 THEN T=T+1: GOTO 220
 210 T=0
 220 IF T<T1 THEN 120
 230 GOTO 1000
 500 F=1: IF I<2 THEN 520
 510 FOR J=2 TO I:F=F/J: NEXT
 520 RETURN
1000 PRINT "S="S
1010 END
JRUN
ПРОГРАММА ВЫЧИСЛЯЕТ СУММУ РЯДА ПО ЭЙЛЕРУ.
ЦЕЛЫЙ ПАРАМЕТР T1=5
S=2.71828183
```

ЭКСТРЕМУМЫ ФУНКЦИЙ

2.20. Метод наискорейшего спуска

Методом наискорейшего спуска может быть найден минимум функции n переменных $F(x_1, ..., x_n)$ или найдены решения системы уравнений [3] вида

$$F_i(x_1, x_2, \ldots, x_n)=0,\ i=\overline{1,\ n}. \qquad (1)$$

Решение системы (1) эквивалентно отысканию равного нулю минимума функции

$$F(x_1, x_2, \ldots, x_n)=\sum_{i=1}^{n} F_i^2(x_1, x_2, \ldots, x_n). \qquad (2)$$

Для нахождения минимума F задаем некоторое начальное приближение $x_i^{(0)}\ (i=\overline{1,\ n})$ и строим последующие приближения по формуле

$$x_i^{(j+1)}=x_i^j+\lambda^{(j)}v_i^{(j)}\ (i=\overline{1,\ n};\ j=0, 1, 2, \ldots), \qquad (3)$$

где направления $V_i^{(j)}$ и величина шага на j-м шаге соответственно равны:

$$V_i^{(j)} = -\frac{\partial F}{\partial x_i},$$

$$\lambda^{(j)} = \sum_{i=1}^{n} \left(\frac{\partial F}{\partial x_i}\right)^2 \left[\sum_{i,\,k} \frac{\partial^2 F}{\partial x_i \partial x_k} \frac{\partial F}{\partial x_i} \frac{\partial F}{\partial x_k}\right]^{-1}. \qquad (4)$$

Все производные в (4) вычисляются при $x_i = x_i^{(j)}$.

Итерационный процесс (3) продолжается до тех пор, пока не будет удовлетворяться условие

$$|x_i^{(j+1)} - x_i^{(j)}| \leqslant \varepsilon_1 \quad (i = \overline{1, n}) \qquad (5)$$

или все производные $\partial F/\partial x_k$ не станут равны нулю (точнее, пока числитель в $\lambda^{(j)}$ не обратится в нуль).

В программе используются переменные: N — количество переменных (n); $E1$ — точность (ε_1); $Y(N)$ — массив, содержащий начальное (затем $j+1$) приближение корня ($x_i^{(j+1)}$); $X(N)$ — j-е приближение корня (x_i^j); $A(N)$ — массив, в котором содержатся производные $\partial F/\partial x_k$; $B(N, N)$ — то же для $\partial^2 F/\partial x_i \partial x_k$; $S = \sum_{i=1}^{n} \left(\frac{\partial F}{\partial x_i}\right)^2$; L — значение $\lambda^{(j)}$; C — знаменатель в (4); F — значение $F(x_1, \ldots, x_n)$; I, K — переменные цикла.

Структура программы:

10—40 — ввод начального приближения $x_i^{(0)}$;
100—160 — вычисления по (3) и проверка условия (5);
500—650 — подпрограмма вычисления по формулам (4);
700 — вычисление значения $F(x_1, x_2, \ldots, x_n)$;
1000—1020 — вывод корня и значения функции.

В контрольном примере найден минимум функции

$$F = (x_1 - 1)^2 + x_2^2 - \frac{1}{2}\cos(2x_3):$$

$$(x_1, x_2, x_3) = (1, 0, k\pi), \quad k = 0, 1, 2, \ldots .$$

```
JLIST
  10 PRINT "МЕТОД НАИСКОРЕЙШЕГО СПУСКА."
  20 E1=1E-8:N=3
  30 DIM X (N), Y (N), A (N), B (N, N)
  40 FOR I=1 TO N: PRINT "X ("I;: INPUT")=";Y (I): NEXT
 100 FOR I=1 TO N:X (I)=Y (I): NEXT
 110 GOSUB 500
 120 FOR I=1 TO N:Y (I)=X (I)-L*A (I): NEXT
 130 FOR I=1 TO N
 140 IF ABS (Y (I)-X (I))>E1 THEN 100
 150 NEXT
 160 GOTO 700
 500 A (1)=2*(X (1)-1)
 510 A (2)=2*X (2)
 520 A (3)=SIN (2*X (3))
 530 FOR I=1 TO N: FOR K=1 TO N:B (I, K)=0: NEXT K, I
```

```
 540 B (1, 1) =2:B (2, 2) =2:B (3, 3) =2 * COS (2 * X (3))
 600 S=0:C=0
 610 FOR I=1 TO N:S=S+A (I) ^ 2
 620 FOR K=1 TO N:C=C+B (I,K) * A (I) * A (K):
 630 NEXT K, I
 640 IF C=0 THEN 700
 650 L=S/C: RETURN
 700 F=(Y (1) -1)^2+Y (2)^2-COS (2 * Y (3))/2
1000 PRINT "КОРЕНЬ:"
1010 FOR I=1 TO N: PRINT "X ("I")="Y (I): NEXT
1020 PRINT "ПРИ ЭТОМ F="F
1030 END

JRUN
МЕТОД НАИСКОРЕЙШЕГО СПУСКА.
X (1) =1
X (2) =2
X (3) =3
КОРЕНЬ:
X (1) =1
X (2) = -1.07268253E-17
X (3) =3.14159266
ПРИ ЭТОМ F= -.5
```

2.21. Минимизация функции многих переменных методом конфигураций

Пусть задана функция n переменных $F(x_1, x_2, \ldots, x_n)$. Поиск минимального значения начинаем с некоторой начальной точки P_i и начального шага $S_{1i}=d$. Вычисляем значение функции в точках $F(P_1, \ldots, P_i \pm d, \ldots, P_n)$ и $F(P_1, \ldots, P_i, \ldots, P_n)$. Если из этих трех значений функция минимальна в крайней точке, то принимаем ее за начальную, если в средней точке $(P_1, \ldots, P_i, \ldots, P_n)$, то она принимается за начальную, а размер шага по x_i уменьшается на коэффициент r и становится равным по i-му аргументу $S_{1i}=S_{1i} \cdot r$. Вычисления прекращаются, если размер шага по всем аргументам становится меньше d_1 или количество вычислений функции F становится больше m_2.

В программе используются переменные: N — количество переменных (n); D, R — начальное значение и коэффициент уменьшения шага (на выходе D — это конечный размер шага); $D1$ — минимальный размер шага; $M2$ — максимально допустимое количество вычислений функций; S — значение функции $F(x_1, \ldots, x_n)$; $P(N)$ — начальная точка поиска на входе и найденная точка на выходе; $M1$ — наименьшая степень 10, представимая в данной машине; $H(N)$-новая начальная точка поиска; E — счетчик количества вычислений функции; I — переменная цикла; $S2$, $S4$ — значения F в точке $H(I)$; $S3$ — значение F в точке $P(I)$.

Структура программы:

10—50 — ввод и задание n, r, d_1, m_1, m_2, P_i;

100—110 — задание начального размера шага по i-му аргументу;

120 — вычисление значения функции в начальной точке, количество вычислений функции $E=1$;

130 — значение функции в новой точке;

140—230 — выбор нового значения функции, направления изменения шага;

240—250 — проверка условий прекращения вычислений;

260—270 — изменение величины шага;

500 — подпрограмма вычисления значений функции в точке p_i;

510 — то же для точки h_i;

600—680 — подпрограмма выбора минимального значения функции $S2$ и точки, в которой это значение достигается;

700—710 — счетчик количества вычислений функции;

800 — подпрограмма вычисления функции;

1000—1020 — вывод значения корня p_i, функции S, количества вычислений функции E.

Программа получена переводом на язык Бейсик Алгол-программы 178б [7].

В контрольном примере найдена точка минимума функции

$$F=x_1^2+(x_2-3)^2.$$

```
JLIST
 10 N=2
 20 R=.2: D1=1E-8: M2=200: M1=1E-34
 30 DIM P (N), S1 (N), H (N): INPUT "НАЧ. РАЗМЕР ШАГА D="; D
 40 PRINT "НАЧАЛЬНАЯ ТОЧКА ПОИСКА:"
 50 FOR I=1 TO N: PRINT "X ("I;: INPUT") ="; P (I): NEXT
100 FOR I=1 TO N: S1 (I)=D * ABS (P (I)): IF S1 (I)<M1 THEN S1 (I)=D
110 NEXT I
120 GOSUB 500: S3=S:E=1
130 S2=S3
140 FOR I=1 TO N:H (I)=P (I): NEXT
150 GOSUB 600: IF S2>=S3 THEN 240
160 FOR I=1 TO N: IF (H (I)>P (I))=(S1 (I)<0) THEN S1 (I)=-S1 (I)
170 Q=P (I):P (I)=H (I):H (I)=2 * H (I)-Q: NEXT
180 S3=S2: GOSUB 700
190 GOSUB 510: S2=S: S4=S: GOSUB 600
200 IF S2>=S3 THEN 130
210 I=1
220 IF ABS (H (I)-P (I))>.5 * ABS (S1 (I)) THEN 160
230 I=I+1: IF I<=N THEN 220
240 IF D<D1 THEN 1000
250 IF E>M2 THEN 1000
260 D=D * R
270 FOR I=1 TO N: S1 (I)=R * S1 (I): NEXT
```

```
 280 GOTO 130
 290 GOTO 1000
 500 FOR L=1 TO N:X (L) =P (L): NEXT: GOSUB 800: RETURN
 510 FOR L=1 TO N:X (L) =H (L): NEXT: GOSUB 800: RETURN
 600 FOR I=1 TO N
 610 H (I) =H (I) +S1 (I) :GOSUB 510
 620 S4=S:E=E+1
 630 IF S4<S2 THEN S2=S4: GOTO 680
 640 S1 (I) = -S1 (I) :H (I) =H (I) +2 * S1 (I)
 650 GOSUB 700: GOSUB 510: S4=S
 660 IF S4<S2 THEN S2=S4: GOTO 680
 670 H (I) =H (I) -S1 (I)
 680 NEXT I: RETURN
 700 IF E<M2 THEN E=E+1: RETURN
 710 GOTO 1000
 800 S=X (1)^2+ (X (2) -3)^2: RETURN
1000  PRINT "КОРНИ РАВНЫ:"
1010 FOR I=1 TO N: PRINT "P ("I") ="P (I): NEXT
1020 PRINT "ПРИ ЭТОМ S="S: PRINT "КОЛ. ВЫЧИСЛЕНИЙ ФУНКЦИИ
     E="E
1030 END
```

```
JRUN
НАЧ. РАЗМЕР ШАГА D=.5
НАЧАЛЬНАЯ ТОЧКА ПОИСКА:
X (1) =1
X (2) =1
КОРНИ РАВНЫ:
P (1) =0
P (2) =3
ПРИ ЭТОМ S=3.46944695E-18
КОЛ. ВЫЧИСЛЕНИЙ ФУНКЦИИ E=70
```

Глава III

КОМПЛЕКСНЫЙ АНАЛИЗ

3.1. Умножение комплексных чисел

По программе вычисляется произведение двух комплексных чисел:

$$e+if=(a+ib)(c+id)=(ac-bd)+i(ad+bc).$$

В программе используются переменные: A, B, C, D, E, F соответственно a, b, c, d, e, f.

```
JLIST
  10 PRINT "ПРОГРАММА ВЫЧИСЛЯЕТ ПРОИЗВЕДЕНИЕ КОМПЛЕКСНЫХ
     ЧИСЕЛ": PRINT "A+B*I И C+D*I".
  20 PRINT "ВВЕДИТЕ ДЕЙСТВИТЕЛЬНУЮ И МНИМУЮ ЧАСТЬ ПЕРВО
     ГО ЧИСЛА": INPUT" (A, B)=";A, B
  30 PRINT "ВВЕДИТЕ ДЕЙСТВИТЕЛЬНУЮ И МНИМУЮ ЧАСТЬ ВТОРО
     ГО ЧИСЛА": INPUT" (C, D)=";C,D
 100 E=A*C-B*D:F=A*D+B*C
1000 PRINT "(E+F*I)="E;: IF F>=0 THEN PRINT "+";
1010 PRINT F"*I"
1020 END

JRUN
ПРОГРАММА ВЫЧИСЛЯЕТ ПРОИЗВЕДЕНИЕ КОМПЛЕКСНЫХ ЧИСЕЛ
A+B*I И C+DI
ВВЕДИТЕ ДЕЙСТВИТЕЛЬНУЮ И МНИМУЮ ЧАСТЬ ПЕРВОГО ЧИСЛА
(A, B)=2,5
ВВЕДИТЕ ДЕЙСТВИТЕЛЬНУЮ И МНИМУЮ ЧАСТЬ ВТОРОГО ЧИСЛА
(C, D)=-2,1
(E+F*I)=-9-8*I
```

3.2. Деление комплексных чисел

По программе находится частное от деления двух комплексных чисел:

$$e+if=\frac{a+ib}{c+id},\ (c,\ d)\neq(0,\ 0).$$

Вычисления проводятся по формулам:

$$\frac{a+ib}{c+id}=\frac{a+bd/c+i(b-ad/c)}{c+d^2/c}\ \text{при}\ |c|\geqslant|d|;$$

$$\frac{a+ib}{c+id}=\frac{b+ac/d+i\,(bc/d-a)}{d+c^2/d} \text{ при } |c|<|d|. \qquad (1)$$

В программе используются переменные: A, B, C, D, E, F соответственно a, b, c, d, e, f; $R=d/c$ или c/d; $D1$ — знаменатели в (1).

В контрольном примере найдено:

$$\frac{-2+6i}{3-4i}=\frac{(-2+6i)\,(3+4i)}{25}=-1.2+0.4i.$$

```
JLIST
  10 PRINT "ПРОГРАММА ВЫЧИСЛЯЕТ КОМПЛЕКСНОЕ ЧАСТНОЕ ОТ
     ДЕЛЕНИЯ": PRINT" A+B*I НА C+D*I"
  20 PRINT "ВВЕДИТЕ ДЕЙСТВИТЕЛЬНУЮ И МНИМУЮ ЧАСТЬ ПЕРВО
     ГО ЧИСЛА:": INPUT"(A, B)=": A, B
  30 INPUT "ВТОРОГО ЧИСЛА (C, D)=": C, D
  50 IF C=0 AND D=0 THEN PRINT "В ЗНАМЕНАТЕЛЕ 0": END
 100 IF ABS (C)<ABS (D) THEN 140
 110 R=D/C: D1=C+R*D
 120 E=(A+B*R) / D1:F=(B-A*R) / D1
 130 GOTO 1000
 140 R=C/D: D1=D+R*C
 150 E=(A*R+B)/D1:F=(B*R-A)/D1
1000 PRINT "ЧАСТНОЕ РАВНО:": PRINT "(E+F*I)=";
1010 PRINT E;: IF F>=0 THEN PRINT "+";
1020 PRINT F"*I"
1030 END

JRUN
ПРОГРАММА ВЫЧИСЛЯЕТ КОМПЛЕКСНОЕ ЧАСТНОЕ ОТ ДЕЛЕНИЯ
A+B*I НА C+D*I
ВВЕДИТЕ ДЕЙСТВИТЕЛЬНУЮ И МНИМУЮ ЧАСТЬ ПЕРВОГО ЧИСЛА:
(A, B)= -2,6
ВТОРОГО ЧИСЛА (C, D)=3, -4
ЧАСТНОЕ РАВНО:
(E+F*I)= -1.2+.4*I
```

3.3. Корни n-й степени комплексного числа

По программе вычисляются все n корней комплексного числа $z=r+iu$:

$$w=\sqrt[n]{r+iu}=\sqrt[n]{|z|}\,[\cos(t+2k\pi)+i\sin(t+2k\pi)]. \qquad (1)$$

Вычисления производятся по формулам:

$$S=(r^2+u^2)^{1/2n},$$

$$t=\begin{cases}\frac{\pi}{2}\operatorname{sgn}(u) \text{ при } r=0,\\ \operatorname{arctg}\frac{u}{r} \text{ при } r>0,\\ \pi+\operatorname{arctg}\frac{u}{r} \text{ при } r<0,\ u\geqslant 0,\\ \operatorname{arctg}\frac{u}{r}-\pi \text{ при } r<0,\ u<0,\end{cases} \qquad (2)$$

$$W_k = S\left[\cos\frac{t+2k\pi}{n} + i\sin\frac{t+2k\pi}{n}\right],\ k=\overline{0,\ n-1}. \qquad (3)$$

В программе используются переменные: N — показатель корня (n); R, U — действительная и мнимая части числа $z=r+iu$; $M=1/N$; $P1=\pi=3.14159265...$; $S=\sqrt[n]{|z|}$; T — аргумент z; $C=2\pi/n$; I — переменная цикла.

Структура программы:

10—40 — ввод n, r, u;

100—150 — вычисление по формулам (2), (3);

1000—1060 — вывод корней w_k.

Программа получена переводом на язык Бейсик Алгол-программы 53б [5].

В контрольном примере найдены корни $\sqrt[6]{-8+6i}$:

$W_0=1.342407746...+i0.593612783...$,

$W_1=0.157120123...+i1.459365601...$,

$W_2=-1.185287622...+i0.865752819...$,

$W_3=-1.342407746...-i593612782...$,

$W_4=-0.157120124...-i1.459365601...$,

$W_5=-1.185287622...-i0.865752819...$.

```
JLIST
 10 PRINT "ПРОГРАММА НАХОДИТ N КОРНЕЙ КОРНЯ N-Й СТЕПЕНИ
    КОМПЛЕКСНОГО ЧИСЛА"
 20 INPUT "ВВЕДИТЕ ДЕЙСТВИТЕЛЬНУЮ ЧАСТЬ R="; R
 30 INPUT "ВВЕДИТЕ МНИМУЮ ЧАСТЬ U="; U
 40 INPUT "ПОКАЗАТЕЛЬ КОРНЯ N="; N: DIM R1 (N), I1 (N)
100 M=1/N: P1=3.14159265: S=(R*R+U*U)^(M/2)
110 IF R=0 THEN T=SGN (U)*P1/2: GOTO 140
120 IF R>0 THEN T=ATN (U/R): GOTO 140
130 T=P1+ATN (U/R): IF U<0 THEN T=T-2*P1
140 T=T*M:C=2*P1*M
150 FOR I=1 TO N: R1 (I)=S*COS (T): I1 (I)=S*SIN (T):T=T+C:
    NEXT
1000 PRINT "КОРНИ "N"-Й СТЕПЕНИ РАВНЫ:"
1010 FOR I=1 TO N: PRINT
1020 PRINT "X ("I-1")+I*Y ("I-1")="R1 (I);
1030 IF I1 (I)>=0 THEN PRINT "+";
1040 PRINT I1 (I)"*I"
1050 NEXT
1060 END

JRUN
ПРОГРАММА НАХОДИТ N КОРНЕЙ КОРНЯ N-Й СТЕПЕНИ КОМПЛЕКСНО
ГО ЧИСЛА
ВВЕДИТЕ ДЕЙСТВИТЕЛЬНУЮ ЧАСТЬ R=-8
ВВЕДИТЕ МНИМУЮ ЧАСТЬ U=6
ПОКАЗАТЕЛЬ КОРНЯ N=6
КОРНИ 6-Й СТЕПЕНИ РАВНЫ:
X (0)+I*Y (0)=1.34240775+.593612782*I
X (1)+I*Y (1)=.157120124+1.4593656*I
```

```
X (2) + I * Y (2) = -1.18528762 + .865752821 * I
X (3) + I * Y (3) = -1.34240775 - .593612778 * I
X (4) + I * Y (4) = -.157120127 - 1.4593656 * I
X (5) + I * Y (5) = 1.18528762 - .865752823 * I
```

3.4. Действительная степень комплексного числа

Действительная степень комплексного числа определяется соотношением:

$$z^w = (x+iy)^w = r^w (\cos w\varphi + i \sin w\varphi),$$

где

$$r = \sqrt{x^2+y^2}, \quad -\pi < \varphi \leqslant \pi, \qquad (1)$$

φ — главное значение аргумента z, см. (2) п. 3.3.
Если $w = 1/n$, то программа вычисляет главное значение $\sqrt[n]{z}$.

В программе используются переменные: X, Y — действительная и мнимая части z; W — показатель степени; R, F — модуль $|z|$ в степени w и аргумент z; A, B — действительная и мнимая части z^w; $P1 = \pi = 3.14159265359...$.

Программа получена переводом на Бейсик Алгол-программы 106б [6].

В контрольном примере найдено:

$$(-4+3i)^3 = 44 + 117i.$$

```
JLIST
  10 PRINT "ПРОГРАММА ВЫЧИСЛЯЕТ ДЕЙСТВИТЕЛЬНУЮ СТЕПЕНЬ
     КОМПЛЕКСНОГО ЧИСЛА": PRINT "(X+Y * I)^ W"
  20 INPUT "ВВЕДИТЕ X=";X
  30 INPUT "ВВЕДИТЕ Y=";Y
  40 INPUT "ПОКАЗАТЕЛЬ W=";W
 110 A=0:B=0: P1=3.14159265359
 120 IF X=0 THEN F=.5 * P1 * SGN (Y): GOTO 150
 130 F=ATN (Y/X)
 140 IF X<0 THEN F=F+P1: IF Y<0 THEN F=F-2 * P1
 150 R=(X * X+Y * Y)^(W/2)
 160 A=R * COS (W * F):B=R * SIN (W * F)
1000 IF Y<0 THEN PRINT "(";X;"Y" * I)^"W"=": GOTO 1020
1010 PRINT "(";"X"+"Y" * I)^"W"="
1020 PRINT:IF B<0 THEN PRINT A;"B" * I": GOTO 1030
1030 PRINT A"+"B" * I"
1040 END

JRUN
ПРОГРАММА ВЫЧИСЛЯЕТ ДЕЙСТВИТЕЛЬНУЮ СТЕПЕНЬ КОМПЛЕКСНО
ГО ЧИСЛА (X+Y * I)^ W
ВВЕДИТЕ X=-4
ВВЕДИТЕ Y=3
ПОКАЗАТЕЛЬ W=3
(-4+3 * I)^ 3=
44+117 * I
```

3.5. Комплексная степень комплексного числа

Пусть заданы комплексные числа w и z:

$w=c+id$; $z=a+ib=e^r[\cos(\varphi+2\pi n)+i\sin(\varphi+2\pi n)]$, где $r=0.5\ln(a^2+b^2)$, φ — главное значение аргумента числа z, n — количество оборотов.

Тогда (см., например, [14])

$$z^w=(e^{r+i(\varphi+2\pi n)})^{c+id}=e^{cr-d(\varphi+2\pi n)}[\cos(dr+c\varphi+2cn\pi)+ +i\sin(dr+c\varphi+2cn\pi)]. \quad (1)$$

Вычисления производятся по формуле (1), аргумент определен в п. 3.3 (2).

В программе используются переменные: A, B (C,D) — действительная и мнимая части z (w); X, Y — то же для результата вычисления z^w; N — количество оборотов аргумента числа z; $P1=\pi=3.14159265359...$; $P=\varphi+2\pi n$; $R=0.5\ln(a^2+ +b^2)$; $V=CP+DR$; $W=\exp(CR-DP)$.

Структура программы:

10—40 — ввод a, b, c, d, n;

100—160 — вычисление по формуле (1);

1000—1010 — вывод x, y.

Программа получена переводом на язык Бейсик Алгол-программы 190б [7].

В контрольном примере вычислено при $n=1$:

$$(2+i)^{1-i}=1794.03591-636.78002i.$$

```
JLIST
  10 PRINT "ПРОГРАММА ВЫЧИСЛЯЕТ КОМПЛЕКСНУЮ СТЕПЕНЬ
     C+D*I": PRINT "КОМПЛЕКСНОГО ЧИСЛА A+B*I"
  20 INPUT "ОСНОВАНИЕ (A, B)=";A, B
  30 INPUT "ПОКАЗ. СТЕПЕНИ (C, D)=";C,D
  40 INPUT "КОЛИЧЕСТВО ОБОРОТОВ N=";N
 100 X=0:Y=0: P1=3.1415926539:P=2*N*P1: IF A=0 AND B=0 THEN
     GOTO 1000
 110 IF A=0 THEN P=P+P1/2*SGN (B): GOTO 140
 120 P=P+ATN (B/A): IF A<=0 AND B<0 THEN P=P-P1: GOTO 140
 130 IF A<0 AND B>=0 THEN P=P+P1
 140 R=.5*LOG (A*A+B*B)
 150 V=C*P+D*R:W=EXP (C*R-D*P)
 160 X=W*COS (V):Y=W*SIN (V)
1000 PRINT "["A"+I*("B")]^["C"+I*("D")]="X"+I*("Y")"
1010 END
JRUN
ПРОГРАММА ВЫЧИСЛЯЕТ КОМПЛЕКСНУЮ СТЕПЕНЬ C+D*I
КОМПЛЕКСНОГО ЧИСЛА A+B*I
ОСНОВАНИЕ (A, B)=2,1
ПОКАЗ. СТЕПЕНИ (C, D)=1, -1
КОЛИЧЕСТВО ОБОРОТОВ N=1
[2+I*(1)]^[1+I*(-1)]=1794.03591+I*(-636.78002)
```

Глава IV

ИНТЕГРАЛЫ

4.1. Интегрирование методом Симпсона с оценкой точности

Собственное значение определенного интеграла

$$S=\int_a^b F(x)\,dx$$

находится методом Симпсона (парабол). Отрезок $[a, b]$ разбивается на $n=2m$ частей $x_0=a$, $x_1=a+h$, ..., $x_n=b$ с шагом

$$h=\frac{b-a}{n}. \tag{1}$$

Вычисляются значения $y_i=F(x_i)$ функции в точках x_i и находится значение интеграла по формуле Симпсона [9]:

$$S=S_n+R_n, \tag{2}$$

где

$$S_n=\frac{h}{3}[y_0+4y_1+2y_2+4y_3+\ldots+y_{2m}],$$

$$R_n=-h^5/90\sum_{k=1}^{2m} y^{IV}(\xi_k),\ \xi_k\in[x_{k-1}, x_k].$$

Затем количество точек разбиения удваивается и производится оценка точности вычислений [9]:

$$R_n\approx\frac{|S_{2n}-S_n|}{15}. \tag{3}$$

Если $R_n>\varepsilon_1$, то количество точек разбиения удваивается. Значения суммы $2(y_1+y_2+\ldots+y_{2m-1})$ сохраняются, поэтому для вычисления интеграла при удвоении количества точек разбиения требуется вычислять значения y_i лишь в новых точках разбиения.

В программе используются переменные: A, B — пределы интегрирования; $E1$ — точность; F — значение функции; X — аргумент функции; H — шаг h; S, $S1$, $S2$, $S3$ — рабочие переменные; $X1=x_i+h$.

Структура программы:

10—20 — ввод A, B, ε_1;

100—180 — вычисление по формулам (2) и (3);

500 — подпрограмма вычисления функции $F(x)$;

1000—1010 — вывод S и R.

В контрольном примере вычислен интеграл (при $\varepsilon_1 = 10^{-6}$):

$$\int_2^0 e^x dx = 1 - e^2 = -6.389056099\ldots .$$

```
JLIST
  10 PRINT "ИНТЕГРАЛ ОТ А ДО В ПО СИМПСОНУ"
  20 E1=1E-6: INPUT "A=";A: INPUT "B=";B
 100 S2=1:H=B-A
 110 X=A: GOSUB 500:S=F
 120 X=B: GOSUB 500:S=S+F
 130 S3=S2:H=H/2: S1=0: X1=A+H
 140 X=X1: GOSUB 500: S1=S1+2*F
 150 X1=X1+2*H: IF (X1<B)=(H>0) THEN 140
 160 S=S+S1:S2=(S+S1)*H/3
 170 X=ABS (S3-S2)/15: IF X>E1 THEN 130
 180 GOTO 1000
 500 F=EXP (X): RETURN
1000 PRINT "ИНТЕГРАЛ=";S2
1010 PRINT "ТОЧНОСТЬ=";X
1020 END

JRUN
ИНТЕГРАЛ ОТ А ДО В ПО СИМПСОНУ
A=2
B=0
ИНТЕГРАЛ= -6.38905665
ТОЧНОСТЬ=5.40539622E-07
```

4.2. Вычисление интеграла методом Симпсона от функции, заданной таблично

Если заданы значения функции $F(x)$ на интервале $[a, b]$ y_0, y_1, ..., y_n в равностоящих точках и n четно, то приближенное значение интеграла вычисляется по формуле

$$S = \int_a^b F(x)\,dx \approx \frac{b-a}{3n}[y_0 + 4y_1 + 2y_2 + \ldots + y_n]. \qquad (1)$$

В программе используются переменные: A, B — пределы интегрирования; N — число разбиений интервала $[a, b]$; H — шаг $h=(b-a)/n$; S — рабочая переменная, значение интеграла; $Y(N)$ — массив значений функции; I — переменная цикла.

Структура программы:

10—40 — ввод a, b, n, y_i;

100—120 — вычисление по (1);

1000 — вывод S.

Программа получена переводом на язык Бейсик Алгол-программы 84б [5].

В контрольном примере найдено приближенное значение интеграла $\int_2^0 e^x dx$ по значению $F(x)$ в пяти точках ($n=4$).

```
JLIST
  10 PRINT "ИНТЕГРАЛ ПО СИМПСОНУ ОТ ТАБЛИЧНО ЗАДАННОЙ
     ФУНКЦИИ.": PRINT "ИНТЕРВАЛ — [A, B]": PRINT "КОЛИЧЕСТВО ТО-
     ЧЕК N — ЧЕТНОЕ"
  20 INPUT "N=";N: INPUT "(A, B) =";A,B
  30 H=(B-A)/N
  40 FOR I=0 TO N: PRINT "Y(";A+I*H;: INPUT") =";Y (I): NEXT
 100 S=(Y (0)-Y (N))/2
 110 FOR I=2 TO N STEP 2:S=S+2*Y (I-1)+Y (I): NEXT I
 120 S=2*H*S/3
1000 PRINT "ИНТЕГРАЛ=";S
1010 END

JRUN
ИНТЕГРАЛ ПО СИМПСОНУ ОТ ТАБЛИЧНО ЗАДАННОЙ ФУНКЦИИ.
ИНТЕРВАЛ — [A, B]
КОЛИЧЕСТВО ТОЧЕК N — ЧЕТНОЕ
N=4
(A, B) =2,0
Y (2) =7.389561
Y (1.5) =4.481689
Y (1) =2.71828183
Y (.5) =1.6487213
Y (0) =1
ИНТЕГРАЛ= -6.39129431
```

4.3. Вычисление интеграла методом Ромберга

Задается значение n (на первом шаге $n=2$) и вычисляется приближенное значение интеграла по формуле трапеций (деленное на $b-a$):

$$S_n^{(1)}=\frac{1}{2}S_{n-1}^{(1)}+\frac{1}{n}\sum_{j=1}^{n-1}F\left(a+\frac{2j-1}{n}(b-a)\right). \qquad (1)$$

Затем число n удваивается (k раз). Каждый раз значение $S_n^{(1)}$ запоминается в массиве T (I). При каждом i после вычисления $S_2^{(1)}$ вычисляется $S_2^{(2)}$, $S_2^{(3)}$, ..., $S_2^{(k+1)}$ по рекуррентной формуле

$$S_2^{(j)}=S_2^{(j-1)}+\frac{1}{2^{j-1}}(S_2^{(j-1)}-S_2^{(j-1)}). \qquad (2)$$

Погрешность метода имеет порядок $2n+1$ ($\varepsilon=10^{-2n-1}$).

Программа получена переводом на язык Бейсик Алгол-программы 60б [5].

В программе используются переменные: A, B — пределы интегрирования; K — порядок погрешности; $D=B-A$; $Z(K+1)$ — массив, в котором запоминаются значения $S_2^{(j)}$; $H=(B-A)/N$; M, N — рабочие переменные (целые); I, J — переменные циклов; X — аргумент; F — значение функции.

Структура программы:

10—30 — ввод n, a, b;

100—220 — вычисление по формулам (1) и (2);

500 — подпрограмма вычисления $F(x)$;

1000 — вывод значения интеграла.

В контрольном примере найдено значение интеграла при $n=4$:

$$\int_1^2 \ln x dx = 2\ln 2 - 1 = 0.38629436.$$

```
JLIST
  10 PRINT "ИНТЕГРАЛ ПО РОМБЕРГУ.ПОРЯДОК К."
  20 INPUT "K=";K: DIM T (K+1)
  30 INPUT "ПРЕДЕЛЫ ИНТЕГРИРОВАНИЯ (A, B)="; A, B
 100 D=B-A:X=A: GOSUB 500: F1=F
 110 X=B: GOSUB 500:T (1)=(F1+F)/2:N=1
 120 FOR I=1 TO K
 130 S=0:N=2*N:H=D/N
 140 FOR J=1 TO N STEP 2
 150 X=A+J*H: GOSUB 500:S=S+F
 160 NEXT J
 170 T (I+1)=(2*S/N+T (I))/2:M=1
 180 FOR J=I TO 1 STEP-1
 190 M=4*M:T (J)=T (J+1)+(T (J+1)-T (J))/(M-1)
 200 NEXT J: NEXT I
 210 I1=T (1)*D
 220 GOTO 1000
 500 F=LOG (X): RETURN
1000 PRINT "ИНТЕГРАЛ=" I1
1010 END

JRUN
ИНТЕГРАЛ ПО РОМБЕРГУ. ПОРЯДОК К.
К=4
ПРЕДЕЛЫ ИНТЕГРИРОВАНИЯ (A, B)=1,2
ИНТЕГРАЛ=.386294361
```

4.4. Вычисление криволинейного интеграла в комплексной области

По программе вычисляется значение криволинейного интеграла для комплексной переменной z.

$$S=S_1+iS_2=\int_a^b f(z)\,z'(t)\,dt \qquad (1)$$

с помощью конечной суммы Римана-Стильтьеса

$$S \approx \sum_{i=1}^{n} f(z_i)[z_i - z_{i-1}]. \qquad (2)$$

В программе используются переменные: A, B — пределы интегрирования по t ($a \leqslant b$); N — число интервалов, на которые разбивается отрезок $[a, b]$; $S1$, $S2$ — действительная и мнимая части интеграла (1); $F1$, $F2$, $K1$, $K2$ — действительная и мнимая части функции и аргумента $f(z)$; $Z1$, $Z2$ ($Z3$, $Z4$) — то же для $Z(t_i)$ ($Z(t_{i-1})$); $D1$; $D2 - \mathrm{Re}\Delta Z(t_i)$ и $Im\Delta Z(t_i)$; D — шаг интегрирования $d = (b-a)/n$.

Структура программы:

10—20 — ввод $a < b$, n;

100 — начальные значения переменных;

110—160 — вычисления по формуле (2);

150 — проверка условия $t \in [a, b]$;

500—510 — подпрограмма вычисления $z = z(t)$;

600—610 — подпрограмма вычисления $f(z)$;

1000—1010 — вывод значения интеграла.

Программа получена переводом на Бейсик Алгол-программы 98б [5].

В контрольном примере вычислено значение интеграла $\int dz/\sqrt{z} = -2+2i$ по полуокружности $|z| = 1$, $y \geqslant 0$:

$$z(t) = z_1 + iz_2 = \cos t + i \sin t, \quad t \in [0, \pi],$$
$$f(z) = 1/\sqrt{z} = \sqrt{(1+z_1)/2} - i\sqrt{(1-z_1)/2}.$$

```
JLIST
  10 PRINT "КОМПЛ. КРИВОЛИНЕЙНЫЙ ИНТЕГРАЛ"
  20 INPUT "A<B, (A, B)=";A, B: INPUT "КОЛ. ИНТЕРВАЛОВ N=";N
 100 S1=0: S2=0:D=(B-A)/N:T=A
 110 GOSUB 500: IF T=A THEN 160
 120 D1=Z1-Z3: D2=Z2-Z4: K1=Z3+D1/2: K2=Z4+D2/2
 130 GOSUB 600
 140 S1=S1+F1*D1-F2*D2: S2=S2+F1*D2+F2*D1
 150 IF T>B-.2*D THEN 1000
 160 Z3=Z1: Z4=Z2:T=T+D: GOTO 110
 500 Z1=COS (T):Z2=SIN (T)
 510 RETURN
 600 F1=SQR (.5*(1+K1)):F2=-SQR (.5*(1-K1))
 610 RETURN
1000 PRINT "S1+I*S2=" S1"+I*("S2")"
1010 END

JRUN
КОМПЛ. КРИВОЛИНЕЙНЫЙ ИНТЕГРАЛ
A<B, (A, B)=0,3.14159265
КОЛ. ИНТЕРВАЛОВ N=100
S1+I*S2=-1.99966384+I*(1.99966384)
```

4.5. Вычисление интеграла методом Гаусса

Квадратурная формула Гаусса имеет вид:

$$\int_a^b f(x)\,dx=\frac{b-a}{2}\sum_{i=1}^{n} A_i f(t_i)+R_n, \qquad (1)$$

где $t_i=(b+a)/2+(b-a)\,x_i/2$; x_i, A_i — узлы и коэффициенты квадратурной формулы Гаусса. Причем узлы x_i являются корнями многочленов Лежандра степени n.

Для достаточно гладкой подынтегральной функции формула (1) обеспечивает высокую точность уже при небольшом числе узлов n. Для оценки погрешности вычислений по формуле Гаусса с n узлами справедливо соотношение

$$R_n\approx\frac{b-a}{2{,}5\sqrt{n}}\left(\frac{b-a}{3n}\right)^{2n}\max|f^{(2n)}(x)|,\quad x\in[a,\ b]. \qquad (2)$$

Поскольку концы интервала интегрирования $[a, b]$ не входят в число узлов x_i, то интегрирование по формуле (1) удобно для вычисления несобственных интегралов.

В приведенной ниже программе вычисляется интеграл методом Гаусса с восемью узлами: $x_1=-x_8=-0.96028986$, $A_1=A_8=0.10122854$, $x_2=-x_7=-0.79666648$, $A_2=A_7=0.22238103$, $x_3=-x_6=-0.52553242$, $A_3=A_6=0.31370664$, $x_4=-x_5=-0.18343464$, $A_4=A_5=0.36268378$.

В программе используются переменные: A, B — пределы интегрирования; $A1$, $A2$ равны $(b+a)/2$ и $(b-a)/2$; G — значение интеграла; $X(4)$, $A(4)$ — массивы x_i и A_i $(i=\overline{1,4})$.

Структура программы:

10—20 — ввод a, b;

100 — задание коэффициентов x_i, a_i;

110—140 — вычисления по формуле (1);

500—510 — подпрограмма вычисления $f(t_i)$;

1000—1010 — вывод значения интеграла.

В контрольном примере вычислено значение интеграла

$$\int_0^4 x^2dx=\frac{x^3}{3}\Big|_0^4=21\frac{1}{3}.$$

```
JLIST
  10 PRINT "ИНТЕГРАЛ ПО ГАУССУ, N=8"
  20 INPUT "A,B=";A,B
 100 X(1)=-.96028986:X(2)=-.79666648:X(3)=-.52553242:X(4)=
     =-.18343464:A(1)=.10122854:A(2)=.22238103:A(3)=.3137066
     4:A(4)=.36268378
 110 A1=.5*(A+B):A2=.5*(B-A):G=0
 120 FOR I=1 TO 4:X=A1+A2*X(I):GOSUB 500:G=G+A(I)*F:
     NEXT
 130 FOR I=1 TO 4:X=A1-A2*X(5-I):GOSUB 500:G=G+A(5-I)*F:
```

```
    NEXT
 140 G=G*A2: GOTO 1000
 500 F=X*X
 510 RETURN
1000 PRINT "G="G
1010 END

JRUN
ИНТЕГРАЛ ПО ГАУССУ, N=8
A,B=0,4
G=21.3333332
```

Обычно оценка точности при интегрировании по Гауссу требует вычисления высших производных, что не всегда просто сделать. Поэтому часто оказывается удобным следующий прием. Промежуток интегрирования $[a, b]$ разбивается пополам и производятся вычисления интеграла на этих частях. Результат сравнивается со значением результата интегрирования на всем отрезке $[a, b]$. Если это различие превышает требуемую точность, то производится дальнейшее дробление интервалов. Деление интервала интегрирования пополам приводит не менее чем к 2^{2n}-кратному улучшению точности (при $n=8$ к 65 536-кратному).

Отметим также, что формула (1) точна для полиномов до $(2n-1)$ степени (в нашем случае до 15 степени).

Глава V

ОБЫКНОВЕННЫЕ ДИФФЕРЕНЦИАЛЬНЫЕ УРАВНЕНИЯ

5.1. Модифицированные методы Эйлера решения уравнения первого порядка

При численном интегрировании дифференциального уравнения первого порядка

$$y' = F(x, y) \tag{1}$$

с начальным условием $y(x_0) = y_0$ (задача Коши) сначала выбираем фиксированное приращение аргумента $h = \frac{x_f - x_0}{n}$, где x_f — конечная точка интервала интегрирования, n — число шагов. Затем, применяя процедуру модифицированного метода Эйлера [3], вычисляем y_k по рекуррентной формуле.

В первой программе:

$$y_k = y_{k-1} + h\,[F_{k-1} + F(x_k,\ y_{k-1} + hF_{k-1})]/2. \tag{2}$$

Во второй программе

$$y_k = y_{k-1} + F(x_{k-1} + h/2,\ y_{k-1} + F_{k-1}h/2),\ k = \overline{1, n}, \tag{3}$$

где $F_k = F(x_k, y_k)$, $k = \overline{1,n}$.

В программах используются переменные: X — начальное, затем текущее значение аргумента; H — шаг интегрирования (h); N — количество шагов (n); $Y1$ — значение $y_{k-1} + hF_{k-1}$ (или $y_{k-1} + F_{k-1}h/2$); F — значение $F(x, y)$; $X1$ — конечное значение аргумента; I — переменная цикла.

Структура программы:

10—40 — ввод x_0, x_f, n, y_0;

100—150 — вычисление по формулам (2) (и (3) во второй программе);

500 — подпрограмма вычисления $F(x, y)$;

1000 — вывод $y = y(x_f)$.

В контрольном примере найдено решение дифференциального уравнения $y' = -e^{-x}$, $y(0) = 1$, при $n = 20$ в точке $x = 2$. Точное решение $y = e^{-x}$. В этой точке $y = e^{-2} = 0{,}13533528...$

```
JLIST
  10 PRINT "МЕТОД ЭЙЛЕРА ДЛЯ Y'=F (X, Y)"
  20 INPUT "НАЧ. И КОН. ЗНАЧЕНИЕ АРГУМЕНТА (X, X1) =";X, X1
  30 INPUT "КОЛИЧЕСТВО ШАГОВ N=";N
  40 INPUT "НАЧАЛЬНОЕ ЗНАЧЕНИЕ ФУНКЦИИ Y=";Y
```

```
 100 H=(X1-X)/N: Y1=Y
 110 FOR I=1 TO N
 120 GOSUB 500: F1=F
 130 X=X+H:Y=Y+F*H: GOSUB 500
 140 Y=Y1+H*(F1+F)/2: Y1=Y
 150 NEXT:GOTO 1000
 500 F=-EXP(-X): RETURN
1000 PRINT "Y("X1")="Y
1010 END
```

```
JRUN
МЕТОД ЭЙЛЕРА ДЛЯ Y'=F (X, Y)
НАЧ. И КОН. ЗНАЧЕНИЕ АРГУМЕНТА (X, X1)=0,2
КОЛИЧЕСТВО ШАГОВ N=20
НАЧАЛЬНОЕ ЗНАЧЕНИЕ ФУНКЦИИ Y=1
Y (2)=.13461485
```

```
JLIST
  10 PRINT "МЕТОД ЭЙЛЕРА ДЛЯ Y'=F (X, Y)"
  20 INPUT "НАЧ. И КОН. ЗНАЧЕНИЕ АРГУМЕНТА (X, X1)=";X, X1
  30 INPUT "КОЛИЧЕСТВО ШАГОВ N=";N
  40 INPUT "НАЧАЛЬНОЕ ЗНАЧЕНИЕ ФУНКЦИИ Y=";Y
 100 H=(X1-X)/N/2: Y1=Y
 110 FOR I=1 TO N
 120 GOSUB 500:
 130 X=X+H:Y=Y+F*H: GOSUB 500
 140 Y1=Y1+F*2*H:X=X+H:Y=Y1
 150 NEXT: GOTO 1000
 500 F=-EXP(-X): RETURN
1000 PRINT "Y(" X1")="Y
1010 END
```

```
JRUN
МЕТОД ЭЙЛЕРА ДЛЯ Y'=F (X, Y)
НАЧ. И КОН. ЗНАЧЕНИЕ АРГУМЕНТА (X, X1)=0,2
КОЛИЧЕСТВО ШАГОВ N=20
НАЧАЛЬНОЕ ЗНАЧЕНИЕ ФУНКЦИИ Y=1
Y (2)=.135695455
```

5.2. Метод Рунге-Кутта четвертого порядка для решения уравнения первого порядка

Наиболее употребительным методом Рунге-Кутта является метод четвертого порядка, в котором вычисления производятся по формуле

$$y_{k+1}=y_k+\frac{1}{6}(k_1+2k_2+2k_3+k_4), \qquad (1)$$

где

$k_1=F_k\cdot h=F(x_k, y_k)\cdot h$, $k_2=F(x_k+h/2, y_k+k_1/2)\cdot h$, $k_3=F(x_k+h/2,$

$y_k + k_2/2) \cdot h$, $k_4 = F(x_{k+1}, y_k + k_3) \cdot h$, $k = \overline{0, n-1}$, $h = (x_f - x_0)/n$.

Структура программы и обозначения совпадают с программами п. 51, где *K1, K2, K3* соответственно k_1, k_2, k_3.

Контрольный пример тот же, что в п. 5.1.

```
JLIST
  10 PRINT "МЕТОД РУНГЕ-КУТТА ДЛЯ Y'=F (X, Y)"
  20 INPUT "НАЧ. И КОН. ЗНАЧЕНИЕ АРГУМЕНТА (X, X1)=";X, X1
  30 INPUT "КОЛИЧЕСТВО ШАГОВ N=";N
  40 INPUT "НАЧАЛЬНОЕ ЗНАЧЕНИЕ ФУНКЦИИ Y=";Y
 100 H=(X1-X)/N: Y1=Y
 110 FOR I=1 TO N
 120 GOSUB 500: K1=F*H
 130 X=X+H/2:Y=Y1+K1/2: GOSUB 500: K2=F*H
 140 Y=Y1+K2/2: GOSUB 500: K3=F*H
 150 X=X+H/2:Y=Y1+K3: GOSUB 500
 160 Y=Y1+(K1+K2*2+K3*2+F*H)/6:Y1=Y
 170 NEXT : GOTO 1000
 500 F=-EXP (-X): RETURN
1000 PRINT "Y("X1")="Y
1010 END
JRUN
МЕТОД РУНГЕ-КУТТА ДЛЯ Y'=F (X, Y)
НАЧ. И КОН. ЗНАЧЕНИЕ АРГУМЕНТА (X, X1)=0,2
КОЛИЧЕСТВО ШАГОВ N=20
НАЧАЛЬНОЕ ЗНАЧЕНИЕ ФУНКЦИИ Y=1
Y (2) =.135335253
```

5.3. Обыкновенные дифференциальные уравнения высших порядков и системы дифференциальных уравнений

Система обыкновенных дифференциальных уравнений высшего порядка путем введения новых функций может быть сведена к системе уравнений первого порядка (см. [3], [9]). Рассмотрим такую систему

$$y_i' = F_i(x, y), \tag{1}$$

где $i = \overline{1, n}$, $y = (y_1, y_2, \ldots, y_n)$, $y(x_0) = y^{(0)} = (y_1^{(0)}, \ldots, y_n^{(0)})$.

Систему (1) будем решать методом Рунге-Кутта, вычисляя y_i^k по формуле

$$y_i^{(k+1)} = y_i^{(k)} + \frac{1}{6}(k_1^{(i)} + 2k_2^{(i)} + 2k_3^{(i)} + k_4^{(i)}), \tag{2}$$

где

$$k_1^{(i)} = F_i(x^{(k)}, y^{(k)})\, h,\ k_2^{(i)} = h \cdot F_i(x^{(k)} + h/2,\ y^{(k)} + k_1/2),$$
$$k_3^{(i)} = h \cdot F_i(x^{(k)} + h/2,\ y^{(k)} + k_2/2),\ k_4^{(i)} = hF(x^{(k+1)}, y^{(k)} + k_3),$$
$$k_s = (k_3^{(1)}, k_s^{(2)}, \ldots, k_s^{(n)}),\ s = \overline{1,4},\ y^{(k)} = (y_1^{(k)}, y_2^{(k)}, \ldots, y_n^{(k)}),$$
$$h = (x_f - x_0)/m, \tag{3}$$

m — количество шагов интегрирования.

В программе используются переменные: N — количество интегрируемых функций (n); M — количество шагов (m); X — начальное, затем текущее значение аргумента ($x_k = x_0 + kh$, $k = \overline{1, m}$); $X1$ — конечное значение аргумента x_f; $Y(N)$ — массив, содержащий начальное, затем текущее значение функций ($y_i^{(k+1)}$); $Y1(N)$ — то же для предыдущего приближения ($y_i^{(k)}$); $F(N)$ — массив, содержащий правые части уравнений (1), умноженные на h; $K1(N)$ — массив, содержащий $(k_1^{(i)} + 2k_2^{(i)} + 2k_3^{(i)} + k_4^{(i)})/6$; I, L — переменные циклов.

Структура программы:

10—70 — ввод x_0, x_f, m, $y_i^{(0)}$, задание n;
100 — вычисление h, задание $y_i^{(0)}$;
110—200 — цикл по количеству шагов;
120—130 — вычисление $k_1^{(i)}$;
140—150 — вычисление $k_2^{(i)}$ и суммы $k_1^{(i)} + 2k_2^{(i)}$;
160—170 — вычисление $k_1^{(i)} + 2k_2^{(i)} + 2k_3^{(i)}$;
180—190 — вычисление $y_i^{(k+1)}$;
500—530 — подпрограмма вычислений $hF_i(x, y)$;
1000—1010 — вывод $y_i(x_f)$.

В контрольном примере решена система уравнений

$$\begin{cases} y_1' = y_2, \\ y_2' = -y_1 \end{cases}$$

при заданных начальных условиях $y_1(0) = 1$, $y_2(0) = 0$ и $m = 20$.

Точное решение этой системы $y_1 = \cos x$, $y_2 = -\sin x$, при $x_f = 2$ принимает значения $y_1(2) = -0{,}41614683\ldots$, $y_2(2) = -0{,}90929742\ldots$

```
JLIST
  10 PRINT "МЕТОД РУНГЕ-КУТТА ДЛЯ N УРАВНЕНИЙ."
  20 INPUT "НАЧ. И КОН. ЗНАЧЕНИЯ АРГУМЕНТА (X, X1) =";X, X1
  30 INPUT "КОЛ. ШАГОВ M=";M
  40 N=2
  50 DIM Y (N), Y1 (N), K1 (N), F (N)
  60 PRINT "НАЧ. ЗНАЧЕНИЯ ФУНКЦИЙ:"
  70 FOR I=1 TO N: PRINT "Y ("I;: INPUT ") =";Y (I): NEXT
 100 H= (X1-X)/M: FOR I = 1 TO N: Y1 (I) =Y (I): NEXT
 110 FOR I=1 TO M
 120 GOSUB 500
 130 FOR L=1 TO N: K1 (L) =F (L):Y (L) =Y1 (L) +F (L)/2: NEXT
 140 X=X+H/2: GOSUB 500
 150 FOR L=1 TO N: K1 (L) =K1 (L) +2*F (L):Y (L) =Y1 (L) +F (L)/2:
     NEXT
 160 GOSUB 500
 170 FOR L=1 TO N: K1 (L) =K1 (L) +2*F (L):Y (L) =Y1 (L) +F (L):NEXT
 180 X=X+H/2: GOSUB 500
 190 FOR L=1 TO N:Y (L) =Y1 (L) + (K1 (L) +F (L))/6: Y1 (L) =Y (L):
     NEXT
 200 NEXT I
```

```
 210 GOTO 1000
 500 F (1)=H * Y (2)
 510 F (2)=-H * Y (1)
 530 RETURN
1000 PRINT "РЕШЕНИЕ СИСТЕМЫ:"
1010 FOR I=1 TO N: PRINT "Y"I"(" X1")="Y (I): NEXT
1020 END

JRUN
МЕТОД РУНГЕ-КУТТА ДЛЯ N УРАВНЕНИЙ.
НАЧ. И КОН. ЗНАЧЕНИЯ АРГУМЕНТА (X, X1)=0,2
КОЛ. ШАГОВ M=20
НАЧ. ЗНАЧЕНИЯ ФУНКЦИЙ:
Y (1)=1
Y (2)=0
РЕШЕНИЕ СИСТЕМЫ:
Y1 (2)=-.416145269
Y2 (2)=-.909297992
```

5.4. Метод Рунге-Кутта с автоматическим выбором шага

По программе интегрируется система дифференциальных уравнений первого порядка

$$y_i' = F_i(x, y),\ i = \overline{1, n}, \tag{1}$$

с начальными условиями $y(x_0) = (y_1^{(0)}, y_2^{(0)}, \ldots, y_n^{(0)})$ методом Рунге-Кутта (см. п. 5.3) с автоматическим выбором шага. Вначале шаг интегрирования выбирается равным (100—130):

$$h = (x_f - x_0)/2. \tag{2}$$

По формулам (2), (3) п. 5.3 вычисляются значения функций $y_{1i} = y_i(x_0 + 2h)$ (т. е. на шаге интегрирования $2h$, см. строки 140—160), затем эти же формулы применяются дважды с шагом, равным h: вначале для нахождения решения y_{2i} в точках $x_0 + h$ строки (170—190) и по этим значениям для нахождения $y_{3i} = y_i(x_0 + 2h)$ строки (200—210). Полученные значения y_{1i} и y_{3i} сравниваются (220—250). Если абсолютные значения разности мантисс y_{1i} и y_{3i} (после выравнивания порядков этих величин и 10^{ε_2} см. строки 900—910) больше ε_1, то происходит уменьшение шага вдвое (300—320).

После пятикратного интегрирования с шагом h происходит двукратное увеличение шага (280—290). После вычисления $y(x_0 + 2h)$ за исходную принимается точка $x + 2h$, а за исходное значение функции — $y_3(x + 2h)$ (260—270), затем процесс вычисления повторяется. Начальный шаг интегрирования может быть задан и извне. Для этого необходимо в строке 70 записать $P1 = 0$, $H1$ равно выбранному шагу. Аналогично при повторном обращении к этой программе как к подпрограмме надо положить $P1 = 0$ и перейти к строке 100. Начальный шаг при этом будет автоматически равен

последнему шагу интегрирования в предыдущем обращении к программе (см. строку 110).

В программе используются переменные: N — количество уравнений (n); X, $X1$ — начальное и конечное значения аргумента (x_0, x_f); $Y(N)$ — массив, содержащий начальные, затем текущие (270) значения искомых функций $y_k(x)$; $Y1(N)$, $Y2(N)$, $Y3(N)$ — соответственно y_{1i}, y_{2i}, y_{3i} (см. выше); $V(N)$ — текущие значения $y_i(X)$ при вычислении правых частей уравнений (1); $V1(N)$ — рабочий массив; $F(N)$ — правые части уравнений (1), умноженные на h, при этом $F(I)=h_iF_i(T, V_i)$; $E1$ — точность численного интегрирования на каждом шаге; $E2$ — показатель степени, определяющий точность интегрирования (показатель степени переменной eta в программе 9б [4]); $K1(N)$ — массив, содержащий $(k_1^{(i)}+2k_2^{(i)}+2k_3^{(i)}+k_4^{(i)})/6$; $P1$ — значение равно 1, если необходимо вначале определить оптимальный шаг интегрирования; $H1$ — последний шаг интегрирования; $S1$ — целое число, определяющее, когда можно увеличить шаг интегрирования; U — значение равно 1 на последнем шаге интегрирования и 0 в противном случае; A — показатель степени наибольшего из чисел $|y_1|$, $|y_3|$, 10^{E2}; $E3$ — рабочая переменная; I — переменная циклов.

Программа получена переводом на язык Бейсик (с переработкой) Алгол-программы 9б [4].

В контрольном примере найдено решение системы двух уравнений

$$\begin{cases} y_1' = y_2, \\ y_2' = -y_1 \end{cases}$$

с начальными условиями $y_1(0)=0$.

Программа работает медленнее по сравнению с предыдущей (при одинаковой погрешности вычислений).

```
JLIST
 10 PRINT "МЕТОД Р-К ДЛЯ СИСТЕМЫ N УРАВНЕНИЙ."
 20 N=2: DIM Y (N), V (N), Y1 (N), Y2 (N), Y3 (N), V1 (N), F (N), K1 (N)
 30 INPUT "НАЧАЛЬНОЕ ЗНАЧЕНИЕ X="; X: INPUT "КОНЕЧНОЕ ЗНА
    ЧЕНИЕ X4="; X4
 40 PRINT "НАЧАЛЬНЫЕ ЗНАЧЕНИЯ ФУНКЦИЙ:"
 50 FOR I=1 TO N: PRINT "Y0 ("I;: INPUT")="; Y (I): NEXT I
 60 E1=1E-4: E2=-3
 70 P1=1
100 IF P1=1 THEN H=X4-X: S1=0: GOTO 120
110 H=H1
120 U=0
130 IF (X+2.01 * H-X4>0)=(H>0) THEN U=1: H1=H:H=(X4-X)/2
140 T=X: FOR I=1 TO N:V (I)=Y (I): NEXT
150 H=2 * H: GOSUB 500:H=H/2:X1=T
160 FOR I=1 TO N: Y1 (I)=V (I): NEXT
170 T=X: FOR I=1 TO N:V (I)=Y (I): NEXT
180 GOSUB 500
190 X2=T: FOR I=1 TO N: Y2 (I)=V (I): NEXT
```

```
200 GOSUB 500
210 X3=T: FOR I=1 TO N: Y3 (I)=V (I): NEXT
220 I=1
230 IF I>N THEN 260
235 A=Y1 (I): GOSUB 900: E3=A:A=Y3 (I): GOSUB 900: IF A<E3 THEN
    A=E3
240 IF A<E2 THEN A=E2
245 IF ABS (Y3 (I)-Y1 (I))/10^A>E1 THEN 300
250 I=I+1: GOTO 230
260 X=X3: IF U=1 THEN 1000
270 FOR I=1 TO N:Y (I)=Y3 (I): NEXT
280 IF S1=5 THEN S1=0:H=2*H
290 S1=S1+1: GOTO 130
300 H=H/2:U=0: X1=X2
310 FOR I=1 TO N: Y1 (I)=Y2 (I): NEXT
320 GOTO 170
500 GOSUB 600: FOR I=1 TO N: K1 (I)=F (I): V1 (I)=V (I)+F (I)/2:
    NEXT
510 T=T+H/2: GOSUB 600
520 FOR I=1 TO N: K1 (I)=K1 (I)+2*F (I): V1 (I)=V (I)+F (I)/2:
    NEXT
530 GOSUB 600: FOR I=1 TO N: K1 (I)=K1 (I)+2*F (I): V1 (I)=V (I)+
    F (I): NEXT
540 T=T+H/2: GOSUB 600
550 FOR I=1 TO N:V (I)=V (I)+(K1 (I)+F (I))/6: NEXT
560 RETURN
600 F (1)=H*V (2)
610 F (2)=-H*V (1)
620 PETURN
900 IF A=0 THEN A=-999: RETURN
910 A=INT (.4342944819*LOG (ABS (A)))+1: RETURN
1000 PRINT "РЕШЕНИЕ В ТОЧКЕ X=";X;":"
1010 FOR I=1 TO N: PRINT "YF ("I")="Y3 (I): NEXT
1020 END
```

```
JRUN
МЕТОД Р-К ДЛЯ СИСТЕМЫ N УРАВНЕНИЙ.
НАЧАЛЬНОЕ ЗНАЧЕНИЕ X=0
КОНЕЧНОЕ ЗНАЧЕНИЕ X4=1
НАЧАЛЬНЫЕ ЗНАЧЕНИЯ ФУНКЦИЙ:
Y0 (1)=1
Y0 (2)=0
РЕШЕНИЕ В ТОЧКЕ X=1:
YF (1)=.542847407
YF (2)=-.845361473
```

Глава VI

ИНТЕГРАЛЬНЫЕ УРАВНЕНИЯ

Интегральными уравнениями называются функциональные уравнения, содержащие интегральные преобразования над неизвестной функцией $y(x)$ [3], [20]. Интегральное уравнение называется однородным, если $\alpha y(x)$ есть решение уравнения для произвольного α. Линейное интегральное уравнение в общем виде может быть представлено:

$$g(x)\,y(x)-\lambda\int_{v}k(x,s)\,y(s)\,ds=f(x), \tag{1}$$

где $k(x, s)$ — ядро интегрального преобразования, правая часть $f(x)$ и $g(x)$ являются заданными функциями, λ — параметр уравнения. Область интегрирования v может быть фиксированной (интегральные уравнения типа фредгольмовых) или переменной (интегральные уравнения типа вольтерровых).

Линейное интегральное уравнение первого рода получается из (1) при $g(x)=0$, $\lambda=-1$ и имеет вид:

$$\int_{v}k(x,s)\,y(s)\,ds=f(x). \tag{2}$$

Однородное линейное интегральное уравнение второго рода получается из (1) при $f(x)=0$, $g(x)=1$ и имеет вид:

$$\lambda\int_{v}k(x,s)\,y(s)\,ds=y(x). \tag{3}$$

Неоднородное линейное интегральное уравнение второго рода получается из (1) при $g(x)=1$ и имеет вид:

$$y(x)-\int_{v}k(x,s)\,y(s)\,ds=f(x). \tag{4}$$

Уравнения вида

$$y(x)-\int_{v}k(x,s)\,y(s)\,ds=f(x), \tag{5}$$

$$y(x)-\int_{v}k(x,s)\,F(y(s))\,ds=f(x) \tag{6}$$

являются нелинейными.

В справочном пособии [20] изложены методы решения наиболее распространенных интегральных уравнений. В пособии приведены 33 Алгол-программы, реализующие эти методы. Некоторые из этих программ переведены нами на язык Бейсик и приведены здесь.

6.1. Линейное уравнение Вольтерра второго рода

Линейное интегральное уравнение Вольтерра второго рода имеет вид [20]:

$$y(x)-\int_a^x k(x, s)\, y(s)\, ds = f(x),\ x, s \in [a, b]. \qquad (1)$$

Причем независимые переменные x, s изменяются на промежутке $[a, b]$, ядро $k(x, s)$ непрерывно внутри и на сторонах треугольника, ограниченного прямыми $s=a$, $x=b$, $x=s$. Функция $f(x)$ на $[a, b]$ непрерывна.

В приведенной ниже программе уравнение (1) решается с помощью метода квадратурных формул, суть которого состоит в замене интегрального уравнения апроксимирующей системой алгебраических уравнений относительно дискретных значений искомой функции и решении этой системы. В основе такой замены лежит приближение интеграла квадратурными формулами. Применение формулы трапеций с постоянным шагом h приводит к рекуррентной формуле (1.74) в [20]:

$$y(a)=f(a),\ y(x_i)=\frac{f(x_i)+h\sum_{j=1}^{i-1} A_j k(x_i, x_j)\cdot y(x_j)}{1-hk(x_i, x_i)/2}, \qquad (2)$$

где
$i=2, 3, \ldots, 1+(b-a)/h$, $x_i=a+(i-1)h$, $A_j=1$
при $j>1$ и $A_j=0{,}5$ при $j=1$.

В программе используются переменные: A, B — пределы изменения x и s; H — шаг интегрирования h; X, S — переменные x и s; N — целое число $n=(b-a)/h+1$; $F(N)$ — массив значений $f(x_i)$, $i=\overline{1, n}$; I — переменная цикла; $FNK(S)$ — функция, вычисляющая значения ядра $k(x, s)$; $K1$ — значение $k(x, a+(i-1)h)$; $N1$, значение решения выводится с шагом $n_1 - y(x_1)$, $y(x_{1+n_1})$, ...

Структура программы:

10—20 — ввод a, b, h;

100 — определение функции, вычисляющей значение ядра $k(x, s)$;

110 — вычисление n;

120—140 — вычисление значений массива $F(N)$ — правой части $f(x_i)$; $y(a)=f(a)$;

150—220 — вычисления по формуле (2);

170 — 200 — вычисление числителя в (2);

1000 — 1020 — вывод решения $y(x_i)$ с шагом n_1.

Программа получена переводом на язык Бейсик Алгол-программы *V2* [20].

В контрольном примере решается уравнение (1) при $a=0$, $b=2.5$, $k(x, s)=1-(x-s)e^{2x}$, $f(x)=(1-xe^{2x})\cos 1-e^{2x}\sin 1$. Точное решение этого уравнения $y(x)=e^x(\cos e^x-e^x\sin e^x)$. Решение получено для шага $h=0.01$, решение выведено с шагом вывода $n_1=20$. Контрольные значения с точностью до шести знаков после запятой приведены в правом столбце.

```
JLIST
  10 PRINT "ИНТЕГРАЛЬНОЕ УРАВНЕНИЕ ВОЛЬТЕРРА 2 РОДА"
  20 INPUT "A, B="; A, B: INPUT "ШАГ H="; H
 100 DEF FN K (S)=1-(X-S)*EXP (2*X)
 110 N=INT ((B-A)/H+1): DIM Y (N), F (N)
 120 FOR I=1 TO N
 130 X=A+(I-1)*H:F (I)=COS (1)*(1-X*EXP (2*X))-SIN (1)*
     EXP (2*X)
 140 NEXT: Y (1)=F (1)
 150 FOR I=2 TO N
 160 X=A+(I-1)*H:G=F (I)
 170 FOR J=1 TO I-1
 180 F1=FN K (A+(J-1)*H): IF J=1 THEN F1=.5*F1
 190 G=G+H*F1*Y (J)
 200 NEXT J
 210 Y (I)=G/(1-.5*H*FN K (X))
 220 NEXT I
1000 INPUT "ШАГ ВЫВОДА="; N1
1010 FOR I=1 TO N STEP N1: PRINT "Y ("A+(I-1)*H")="Y (I): NEXT
1020 END
JRUN
ИНТЕГРАЛЬНОЕ УРАВНЕНИЕ ВОЛЬТЕРРА 2 РОДА
A, B=0,2.5
ШАГ H=.01
ШАГ ВЫВОДА=20
Y (0)= -.301168679
Y (.2)= -.983602005
Y (.4)= -2.10101451
Y (.6)= -3.66915334
Y (.8)= -5.28433389
Y (1)= -5.51384225
Y (1.2)= -1.30916764
Y (1.4)=10.5452955
Y (1.6)=25.0106403
Y (1.8)=14.3462028
Y (2)= -45.5272357
Y (2.2)= -39.95559
Y (2.4)=121.881219
```

КОНТРОЛЬНЫЕ ЗНАЧЕНИЯ

```
Y1 (0)= -.301168679
Y1 (.2)= -.983568912
Y1 (.4)= -2.10091531
Y1 (.6)= -3.66894723
Y1 (.8)= -5.28402058
Y1 (1)= -5.51363573
Y1 (1.2)= -1.3098817
Y1 (1.4)=10.5421372
Y1 (1.6)=25.0060653
Y1 (1.8)=14.3550137
Y1 (2)= -45.4898976
Y1 (2.2)= -40.0142057
Y1 (2.4)=121.768357
```

6.2. Уравнение Вольтерра первого рода

Линейное интегральное уравнение Вольтерра первого рода имеет вид [20]:

$$\int_a^x k(x, s)\, y(s)\, ds = f(x), \quad x, s \in [a, b] \qquad [1]$$

Если $k(a, a) \neq 0$, $f(a) = 0$ и если функции $f(x)$, $k(x, s)$ имеют производные $f'(x)$, $k'_x(x, s)$, непрерывные в интервале (a, b), заключенном в интервале интегрирования, внутри которого $k(x, s)$ не обращается в нуль, то уравнение Вольтерра первого рода (1) допускает в интервале (a, b) непрерывное и единственное решение.

В приведенной ниже программе уравнение (1) решается методом квадратурных формул. Вычисление интеграла производится по формуле трапеций с постоянным шагом h:

$$y(a) = f'(a)/k(a, a),$$

$$y_i = \frac{2}{k(x_i, x_i)} \left[f(x_i)/h - \sum_{j=1}^{i-1} A_j\, k(x_i, x_j)\, y_j \right], \qquad (2)$$

где $x_i = a + (i-1)\,h$, $i = 2, 3, \ldots$, $A_j = 1$ при $j > 1$ и $A_j = 0{,}5$ при $j = 1$.

В программе используются переменные: A, B — пределы интегрирования; H — шаг интегрирования h; N — целое число $n = (b-a)/h + 1$; X, S — переменные x и s; $F(N)$ — массив значений $f(x_i)$, $i = \overline{1, n}$; I — переменная цикла; $FNK(S)$ — функция, значение ядра $k(x, s)$; $K1$ — значение $k(x, s)$; $N1$ — шаг вывода решения $n_1 - y(x_1)$, $y(x_{1+n_1})$, ...

Структура программы:

10 — 20 — ввод a, b, h;

100 — определение функции, вычисляющей значение ядра $k(x, s)$;

110 — вычисление n;

120 — 140 — вычисление значений массива $F(N)$ — значений $f(x_i)$;

150 — вычисление $y(a)$ по формулам (2) и (3);

160 — 230 — вычисление выражения в скобках в (2);

1000 — 1020 — вывод решения $y(x_i)$ с шагом n_1.

Программа получена переводом на язык Бейсик Алгол-программы V1 [20]. В контрольном примере решается уравнение (1) при $a = 0$, $b = 3{,}5$, $h = 0{,}05$, $k(x, s) = 2 + x^2 - s^2$, $f(x) = x^2$. Точное решение этого уравнения $y = xe^{-x^2/2}$. Решение получено с шагом $n_1 = 2$. Контрольные значения с точностью до шести знаков после запятой приведены в правом столбце.

```
JLIST
  10 PRINT "УРАВНЕНИЕ ВОЛЬТЕРРА 1 РОДА"
  20 INPUT "(A, B) ="; A, B: INPUT "ШАГ H="; H
 100 DEF FN K (S) =2+ (X-S) * (X+S)
 110 N=INT ((B-A)/H) +1: DIM Y (N), F (N)
```

```
120 FOR I=1 TO N
130 X=A+(I-1)*H:F(I)=X*X
140 NEXT
150 X=A: K1=FN K(A):Y(1)=(-3*F(1)+4*F(2)-F(3))/(2*H*K1)
160 FOR I=2 TO N
170 X=A+(I-1)*H:G=F(I)/H
180 FOR J=1 TO I-1
190 K1=FN K(A+(J-1)*H): IF J=1 THEN K1=.5*K1
200 G=G-K1*Y(J)
210 NEXT J
220 Y(I)=(G+G)/FN K(X)
230 NEXT I
1000 INPUT "ШАГ ВЫВОДА="; N1
1010 FOR I=1 TO N STEP N1: PRINT "Y("A+(I-1)*H")="Y(I): NEXT
1020 END
```

```
JRUN
УРАВНЕНИЕ ВОЛЬТЕРРА 1 РОДА
(A, B)=0,3.5
ШАГ H=.05
ШАГ ВЫВОДА=2
Y(0)=0
Y(.1)=.099625
Y(.2)=.196278043
Y(.3)=.287134566
Y(.4)=.36965458
Y(.5)=.441700359
Y(.6)=.501626991
Y(.7)=.548340666
Y(.8)=.581321985
Y(.9)=.600614771
Y(1)=.606783238
Y(1.1)=.600842575
Y(1.2)=.58416997
Y(1.3)=.558403438
Y(1.4)=.525336624
Y(1.5)=.486816675
Y(1.6)=.444651617
Y(1.7)=.400532204
Y(1.8)=.355970779
Y(1.9)=.312259303
Y(2)=.270445876
Y(2.1)=.231328432
Y(2.2)=.195463135
Y(2.3)=.16318425
Y(2.4)=.134631805
Y(2.5)=.109785273
Y(2.6)=.088497171
```

```
КОНТРОЛЬНЫЕ ЗНАЧЕНИЯ
Y2(0)=0
Y2(.1)=.0995012479
Y2(.2)=.196039735
Y2(.3)=.286799245
Y2(.4)=.369246539
Y2(.5)=.441248451
Y2(.6)=.501162127
Y2(.7)=.547893177
Y2(.8)=.58091923
Y2(.9)=.60027913
Y2(1)=.60653066
Y2(1.1)=.600681869
Y2(1.2)=.584102707
Y2(1.3)=.558424565
Y2(1.4)=.525435538
Y2(1.5)=.486978701
Y2(1.6)=.44485968
Y2(1.7)=.40076833
Y2(1.8)=.356217658
Y2(1.9)=.312501467
Y2(2)=.270670566
Y2(2.1)=.231526103
Y2(2.2)=.195627558
Y2(2.3)=.163312314
Y2(2.4)=.134723431
Y2(2.5)=.109842334
Y2(2.6)=.0885233826
Y2(2.7)=.0705278069
```

Y (2.7) = .0705277116
Y (2.8) = .055576131
Y (2.9) = .0433078068
Y (3) = .0333764461
Y (3.1) = .0254426037
Y (3.2) = .0191867295
Y (3.3) = .0143172667
Y (3.4) = .0105753373

Y2 (2.8) = .0555550656
Y2 (2.9) = .0432702799
Y2 (3) = .0333269898
Y2 (3.1) = .0253849734
Y2 (3.2) = .0191232735
Y2 (3.3) = .0142488722
Y2 (3.4) = .0105016325

6.3. Уравнение Фредгольма второго рода

Линейное интегральное неоднородное уравнение Фредгольма второго рода имеет вид:

$$y(x)-\lambda\int_a^b k(x, s)\, y(s)\, ds = f(x), \qquad (1)$$

где $x\in[a, b]$, $s\in[a, b]$, а ядро определено в квадрате $V=[a, b]\times[a, b]$. Кроме того, полагается, что ядро непрерывно в V. При $\lambda=1$, используя квадратурную формулу трапеций с постоянным шагом h, получим [20]:

$$y_i = -h\sum_{j=1}^{n} A_j k_{ij} y_j = f_i, \quad i=\overline{1, n}, \qquad (2)$$

где $n=(b-a)/h+1$ — целое, $A_j=1$ при $j\neq 1$, n и $A_j=0.5$ при $j=1$ или $j=n$.

В программе используются переменные: $A0$, $B0$ — пределы изменения x и s; H — шаг интегрирования $h=(b-a)/(n-1)$; N — целое число, количество интервалов интегрирования; X, S — переменные x и s; $F(N)$ — массив значений $f(x_i)$, $i=\overline{1, n}$; $FNK(S)$ — функция, значение ядра $k(x, s)$; $N1$ — шаг вывода решения $n_1 — y(x_1)$, $y(x_{1+n_1})$, ...; $A(N+1, N+1)$ — рабочий массив, значение элементов матрицы $\delta_{ij}-hk_{ij}A_j$, где δ_{ij} — символ Кронекера.

Структура программы:

10 — 30 — ввод a, b, n;

100 — определение функции, вычисляющей значение ядра $k(x, s)$;

110 — задание числа π, вычисление g и h.

120 — вычисление массива $F(N)$, значений $f(x_i)$;

130 — 180 — вычисления по (2);

150 — 180 — вычисление $\delta_{ij}-hA_ik_{ij}$ для $j=\overline{1, n}$, i — фиксировано;

190 — 370 — решение системы линейных уравнений (2) методом Гаусса с выбором максимального элемента (см. § 2.3);

1000 — 1020 — вывод решения $y(x_i)$ с шагом n_1.

Программа получена переводом на язык Бейсик Алгол-программы $F2$ [20].

В контрольном примере решается уравнение (1)

при $a=-\pi$, $b=\pi$, $n=37$, $k(x, s)=0.3 / \left[0{,}64\pi \cos^2 \frac{x+s}{2}-\pi\right]$, $f(x)=25-16 \sin^2 x$. Точное решение этого уравнения $y(x)=8.5+$ $+\frac{128}{17} \cos 2x$. Решение получено с шагом $n_1=2$. Контрольные значения с точностью до шести знаков после запятой приведены в правом столбце.

```
JLIST
  10 PRINT "УРАВНЕНИЕ ФРЕДГОЛЬМА II РОДА"
  20 INPUT "(A, B)="; A0, B0: INPUT "КОЛИЧЕСТВО ИНТЕРВАЛОВ
     N="; N
  30 DIM F (N), Y (N), A (N+1, N+1)
 100 DEF FN K (S)=G/(.64 * (COS ((X+S)/2))^2-1)
 110 P1=3.14159265:G=.3/P1:H=(B0-A0)/(N-1)
 120 FOR I=1 TO N:F (I)=25-16 * SIN (A0+(I-1) * H)^2: NEXT
 130 FOR I=1 TO N
 140 X=A0+(I-1) * H:A (I, N+1)=F (I)
 150 FOR J=1 TO N
 160 S=A0+(J-1) * H:A (I, J)=-FN K (S) * H:IF J=1 0R J=N THEN
     A (I, J)=.5 * A (I, J)
 170 IF I=J THEN A (I, J)=1+A (I, J)
 180 NEXT J: NEXT I
 190 U=0
 200 U=U+1:K=U
 210 IF A (K, U)<>0 THEN 240
 220 K=K+1: IF K<=N THEN 210
 230 PRINT "СИСТЕМА НЕ ОПРЕДЕЛЕНА.": END
 240 IF K=U THEN 280
 250 FOR M=U TO N+1
 260 T=A (U, M):A (U, M)=A (K, M):A (K, M)=T
 270 NEXT
 280 FOR J=N+1 TO U STEP-1:A (U, J)=A (U, J)/A (U, U): NEXT
 290 M=N+1: IF K+1>N THEN 340
 300 FOR I=K+1 TO N
 310 FOR J=U+1 TO M
 320 A (I, J)=A (I, J)-A (I, U) * A (U, J)
 330 NEXT J: NEXT I
 340 IF U<>N THEN 200
 350 FOR I=N TO 1 STEP-1:Y (I)=A (I, M): IF I=1 THEN 370
 360 FOR K=I-1 TO 1 STEP-1:A (K, M)=A (K, M)-A (K, I) * Y (I): NEXT
 370 NEXT I
1000 INPUT "ШАГ ВЫВОДА="; N1
1010 FOR I=1 TO N STEP N1: PRINT "Y ("A0+(I-1) * H")="Y (I): NEXT
1020 END
JRUN
УРАВНЕНИЕ ФРЕДГОЛЬМА II РОДА
(A, B) = -3.141592654, 3.141592654
```

КОЛИЧЕСТВО ИНТЕРВАЛОВ N = 37
ШАГ ВЫВОДА = 2

Y (−3.14159266) = 16.0294118
Y (−2.7925268) = 14.267864
Y (−2.44346095) = 9.80746864
Y (−2.0943951) = 4.73529411
Y (−1.74532925) = 1.42466731
Y (−1.3962634) = 1.42466731
Y (−1.04719755) = 4.73529412
Y (−.698131701) = 9.8074686
Y (−.34906585) = 14.267864
Y (0) = 16.0294118
Y (.34906585) = 14.267864
Y (.698131701) = 9.80746863
Y (1.04719755) = 4.73529411
Y (1.3962634) = 1.42466732
Y (1.74532925) = 1.42466731
Y (2.0943951) = 4.73529411
Y (2.44346095) = 9.80746864
Y (2.7925268) = 14.267864
Y (3.14159266) = 16.0294118

КОНТРОЛЬНЫЕ ЗНАЧЕНИЯ

Y3 (−3.14159266) = 16.0294118
Y3 (−2.7925268) = 14.267864
Y3 (−2.44346095) = 9.80746863
Y3 (−2.0943951) = 4.73529412
Y3 (−1.74532925) = 1.42466733
Y3 (−1.3962634) = 1.42466733
Y3 (−1.04719755) = 4.73529412
Y3 (−.698131701) = 9.80746864
Y3 (−.34906585) = 14.267864
Y3 (0) = 16.0294118
Y3 (.34906585) = 14.267864
Y3 (.698131701) = 9.80746863
Y3 (1.04719755) = 4.73529412
Y3 (1.3962634) = 1.42466733
Y3 (1.74532925) = 1.42466733
Y3 (2.0943951) = 4.73529412
Y3 (2.44346095) = 9.80746864
Y3 (2.7925268) = 14.267864
Y3 (3.14159266) = 16.0294118

7.1. Гамма-функция и связанные с ней функции

Гамма-функция $\Gamma(z)$ представляет собой решение функционального уравнения [19]

$$\Gamma(z+1)=z\Gamma(z), \quad \Gamma(1)=1. \tag{1}$$

Она является мероморфной функцией от $z=x+iy$ с простыми полюсами в точках $z=-n$ ($n=0, -1, -2, \ldots$). Для целых n имеем:

$$\Gamma(n+1)=n!, \quad n=0, 1, 2, \ldots . \tag{2}$$

Гамма-функция удовлетворяет следующему уравнению [3], [19]:

$$\Gamma(z)\,\Gamma(-z)=-\pi/z \sin \pi z. \tag{3}$$

В дальнейшем будем рассматривать только действительные $z=x$. Для больших значений x имеет место асимптотическое разложение Стирлинга [19]:

$$\Gamma(x)=\sqrt{\frac{2\pi}{x}}\, e^{x(\ln x-1)} \left[1+\frac{1}{12x}+\frac{1}{288x^2}-\frac{139}{51840x^3}-\frac{571}{2488320x^4}- \ldots\right], \tag{4}$$

причем абсолютная величина ошибки меньше модуля последнего из взятых членов.

Функция $1/\Gamma(x)$ в интервале $[-1, 1]$ может быть разложена в ряд [5]

$$1/\Gamma(x)=x(1+x)(1+a_1x+a_2x^2+ \ldots), \tag{5}$$

где $a_1=-0.422\,784\,335\,092$, $a_2=-0.233\,093\,736\,365$,
$a_3=0.191\,091\,101\,162$, $a_4=-0.024\,552\,490\,887$,
$a_5=-0.017\,645\,242\,118$, $a_6=0.008\,023\,278\,113$,
$a_7=-0.000\,804\,341\,335$, $a_8=-0.000\,360\,851\,496$,
$a_9=0.000\,145\,624\,324$, $a_{10}=1.7527917\cdot10^{-5}$,
$a_{11}=2.625721\cdot10^{-6}$, $a_{12}=1.328554\cdot10^{-6}$,
$a_{13}=-1.8122\cdot10^{-7}$.

Логарифмическая производная $\psi(x)$ гамма-функции определяется выражением [18]

$$\psi(z)=\Gamma'(z)/\Gamma(z). \tag{6}$$

В частности, $\psi(1) = -C = -0.57721566490153\ldots$. Вычисление значений $\psi(x)$ проводится на основании соотношений

$$\psi(x) = \psi(x+1) - 1/x, \tag{7}$$

$$\psi(x) = -\pi \operatorname{ctg}(\pi x) + \psi(1-x) \tag{8}$$

и асимптотической формулы [18]

$$\psi(x) = \ln x - \frac{1}{2x} - \frac{1}{12x^2} + \frac{1}{120x^4} - \frac{1}{252x^6} - \ldots. \tag{9}$$

Полная бета-функция определяется как [3]

$$B(p, q) = \frac{\Gamma(p)\,\Gamma(q)}{\Gamma(p+q)} \tag{10}$$

Неполная гамма-функция $\Gamma_z(p)$ и неполная бета-функция $B_z(p, q)$ определяются аналитическими продолжениями интегралов соответственно:

$$\Gamma_z(p) = \int_0^z t^{p-1} e^{-t} dt \ (Re\ p > 0), \tag{11}$$

$$B_z(p, q) = \int_0^z t^{p-1} (1-t)^{q-1} dt \ (Re\ p > 0,\ Re\ q > 0,\ 0 \leqslant z \leqslant 1). \tag{12}$$

Величина

$$I_z(p, q) = \frac{B_z(p, q)}{B(p, q)} \tag{13}$$

называется отношением неполной бета-функции.

В первой из приведенных здесь программ значение гамма-функции $\Gamma(x)$ вычисляется следующим образом. Вначале исключаются точки $x = 0, -1, -2, \ldots$, в которых $\Gamma(x)$ имеет полюсы (строка 100), затем аргумент функции на основании (1) приводится к интервалу $(-1, 1]$ (строки 110—140) и, наконец, используется разложение $1/\Gamma(x)$ в степенной ряд (5) (см. [5], алгоритм 80б].

В программе используются переменные: $X1$ — начальное значение аргумента; X — рабочая переменная, аргумент $\Gamma(x)$ в (1) и (5); G — значение $\Gamma(x)$; $G1$ — рабочая переменная.

Структура программы:

10—20 — ввод аргумента X;

100—150 — исключение полюсов и приведение аргумента к интервалу $(-1, 1]$;

160—180 — вычисление по формуле (5);

1000—1010 — вывод $\Gamma(x)$.

В контрольном примере найдено значение $\Gamma(-2.5)$. Точное значение [19] $\Gamma(-2.5) = -8\sqrt{\pi}/15 = -0.945308720\ldots$.

```
JLIST
  10 PRINT "ГАММА-ФУНКЦИЯ G (X)"
  20 INPUT "X="; X
 100 X1=X: IF INT (X)=-ABS (X) THEN PRINT "ПРИ X="X"-ПОЛЮС":
     END
 110 G=1
```

```
 120 IF X>1 THEN X=X-1:G=G*X: GOTO 120
 130 G=1/G
 140 IF X<-1 THEN G=G*X:X=X+1: GOTO 140
 150 IF X=1 THEN 180
 160 G1=((((((((-.00000018122*X+.000001328554)*X-.000002625721)*X
     -.000017527917)*X+.000145624324)*X-.000360851496)*X
     -.000804341335)*X+.008023278113)*X-.017645242118)*X
     -.024552490887)*X
 170 G=G*((((G1+.191091101162)*X-.233093736365)*X-.422784335092)
     *X+1)*X*(1+X)
 180 G=1/G
1000 PRINT "G ("X1") ="G
1010 END

JRUN
ГАММА-ФУНКЦИЯ G (X)
X=-2.5
G (-2.5) = -.94530872
```

Во второй программе на основании (1) и (5) вычисляется значение функции $1/\Gamma(x)$. Программа отличается от предыдущей отсутствием строки 180 и тем, что при вычислении $1/\Gamma(x)$ нет необходимости исключать точки $x=0, -1, -2, \ldots$ (строка 100).

Программа получена переводом на язык Бейсик Алгол-программы 80б [5]. В контрольных примерах найдены значения $1/\Gamma(5)$, $1/\Gamma(-1.5)$, $1/\Gamma(1.3)$, совпадающие с табличными [19].

```
JLIST
  10 PRINT "ОБРАТНАЯ ГАММА-ФУНКЦИЯ (G (X))^-1"
  20 INPUT "X=": X
 100 X1=X
 110 G=1
 120 IF X>1 THEN X=X-1:G=G*X: GOTO 120
 130 G=1/G
 140 IF X<-1 THEN G=G*X:X=X+1: GOTO 140
 150 IF X=1 THEN 1000
 160 G1=(((((((((-.00000018122*X+.000001328554)*X-.000002625721)
     *X-.000017527917)*X+.000145624324)*X-.000360851496)*X
     -.000804341335)*X+.008023278113)*X-.017645242118)*X
     -.024552490887)*X
 170 G=G*((((G1+.191091101162)*X-.233093736365)*X-.422784335092)
     *X+1)*X*(1+X)
1000 PRINT "G (" X1")^-1="G
1010 END

JRUN
ОБРАТНАЯ ГАММА-ФУНКЦИЯ (G (X))^-1
X=5
G (5)^-1=.0416666667
```

```
JRUN
ОБРАТНАЯ ГАММА—ФУНКЦИЯ (G (X))^—1
X=—1.5
G (—1.5)^—1=.423142188

JRUN
ОБРАТНАЯ ГАММА—ФУНКЦИЯ (G (X))^—1
X=1.3
G (1.3)^—1=1.11424251
```

По третьей программе значения $\Gamma(x)$ вычисляются по формуле Стирлинга (4). После исключения полюсов (строка 110) аргумент x приводится (если это необходимо) по формуле (1) к значению $x \geqslant A = 15$ (строка 120) и затем производится вычисление по (4).

В программе используются переменные: $X1$ — аргумент $\Gamma(x)$; X — рабочая переменная, аргумент $\Gamma(x)$ в (2) и (4); A — минимальное значение аргумента (при $A=15$ относительная точность $\sim 5.10^{-9}$).

Структура программы:

10—30 — ввод X, задание A;

100—110 — исходные значения G и $X1$, исключение полюсов;

120 — вычисление по формуле (1);

130—150 — вычисление по формуле (4);

1000—1010 — вывод значения $\Gamma(x)$.

В контрольном примере найдены значения $\Gamma(5)=24$, $\Gamma(-2.5)$. При этом $\Gamma(-2.5)=-8\sqrt{\pi}/15=-0.945308720\ldots$.

```
JLIST
  10 PRINT "ГАММА—ФУНКЦИЯ G (X)"
  20 INPUT "X=";X
  30 A=15
 100 G=1: X1=X
 110 IF INT (X)=—ABS (X) THEN PRINT "ПРИ X="X"—ПОЛЮС": END
 120 IF X<A THEN G=G/X:X=X+1: GOTO 120
 130 G=G*SQR (2*3.1415926535/X)*EXP (—X+X*LOG (X))
 140 X=1/X
 150 G=G*(1+X*(1/12+X*(1/288—X*(139/51840+X*571/2488320))))
1000 PRINT "G ("X1")="G
1010 END

JRUN
ГАММА—ФУНКЦИЯ G (X)
X=5
G (5)=24
JRUN
ГАММА—ФУНКЦИЯ G (X)
X=—2.5
G (—2.5)=—.945308727
```

В четвертой программе на основании асимптотической фор-

мулы (9) производится вычисление логарифмической производной $\psi(x)$ гамма-функции.

В программе используются переменные: $X1$ — начальное значение аргумента; X — рабочая переменная, аргумент $\psi(x)$ в (7—9); A — минимальное значение аргумента при вычислениях по (9); $P1=\pi\approx 3.14159265$; P — значение $\psi(x)$.

Структура программы:

10 — 30 — ввод x, задание A;

100 — выделение полюсов;

110 — 140 — вычисление по формулам (7), (8);

150 — 160 — вычисление по формуле (9);

1000 — 1010 — вывод $\psi(x)$.

В контрольных примерах вычислены значения ψ (1) и $\psi(+0.5)$, при этом $\psi(+0.5)=-C-\ln 2=-1.96351002602\ldots$ совпадает с вычисленным значением до семи знаков после запятой.

```
JLIST
  10 PRINT "ЛОГАРИФМИЧЕСКАЯ ПРОИЗВОДНАЯ ГАММА-ФУНКЦИИ"
  20 A=15
  30 INPUT "X="; X: X1=X
 100 IF INT (X)=-ABS (X) THEN PRINT "ПОЛЮС ПРИ X="X: END
 110 IF X=1 THEN P=-.5772156649: GOTO 1000
 120 P=0: P1=3.141592654
 130 IF X<-2 THEN P=-P1*COS (P1*X)/SIN (P1*X):X=1-X
 140 IF X<A THEN P=P-1/X:X=X+1: GOTO 140
 150 X=1/(X-1):P=P-LOG (X)+X/2
 160 X=X*X:P=P-X*((X/21-.1)*X+1)/12
1000 PRINT "ПСИ ("X1")="P
1010 END
JRUN
ЛОГАРИФМИЧЕСКАЯ ПРОИЗВОДНАЯ ГАММА-ФУНКЦИИ
X=1
ПСИ (1)=-.577215665
JRUN
ЛОГАРИФМИЧЕСКАЯ ПРОИЗВОДНАЯ ГАММА-ФУНКЦИИ
X=.5
ПСИ (.5)=-1.96351003
```

В пятой программе значение неполной гамма-функции $\Gamma_x(a)$ вычисляется на основании разложения в степенной ряд [19]:

$$\Gamma_x(a)=x^a\sum_{i=0}^{\infty}\frac{(-1)^i x^i}{i!\,(a+i)}. \qquad (14)$$

В программе используются переменные: X, A — аргументы функции $\Gamma_x(a)$; $E1$ — относительная точность вычисления $\Gamma_x(a)$; G — значение $\Gamma_x(a)$; S, $S1$ — рабочие переменные; I — переменная i в (14).

Структура программы:

10 — 30 — ввод A, X, задание $E1$;

100 — выделение недопустимых значений a;

110—150 — вычисление по формуле (14);
1000—1010 — вывод $\Gamma_x(a)$.

В контрольном примере вычислено значение $\Gamma_2(3)$.

```
JLIST
  10 PRINT "НЕПОЛНАЯ ГАММА-ФУНКЦИЯ G (A, X)"
  20 INPUT "A, X="; A, X
  30 E1=1E-8
 100 IF INT (A)=-ABS (A) THEN PRINT "G ("A","X") -НЕ ОПРЕДЕЛЕНО"
 110 G=1/A:I=1:S=1
 120 S=-S*X/I: S1=S/(A+I)
 130 G=G+S1:I=I+1
 140 IF ABS (S1)>E1*ABS (G) THEN 120
 150 G=G*X^A
1000 PRINT "G ("A", "X") ="G
1010 END

JRUN
НЕПОЛНАЯ ГАММА-ФУНКЦИЯ G (A, X)
A, X=3,2
G (3,2) =.646647168
```

В шестой программе производится вычисление отношения неполной бета-функции (13) на основании разложения $I_x(p, q)$ в степенной ряд. Для улучшения сходимости ряда при $x>0.5$ используется соотношение симметрии

$$I_x(p, q)=1-I_{1-x}(q, p). \qquad (15)$$

В программе используются переменные: $X1$ — начальное значение аргумента x; X — рабочая переменная, аргумент z в (13); P, Q — аргументы p, q; $E1$ — относительная точность; $G1$, T, $T1$, Q, $Q1$, $S1$, $S2$ — рабочие переменные; W — если $x>0.5$, то $W=1$, если $x\leqslant 0.5$, то $W=0$; B — значение $I_z(p, q)$; I — переменная циклов; G — значение $1/\Gamma(u)$; U — аргумент функции $1/\Gamma(u)$.

Структура программы:

10—30 — ввод X, P, Q, задание $E1$;

100—110 — определение правильных значений аргументов, если $X=0$ или $X=1$, то $I_0(p, q)=0$, $I_1(p, q)=1$;

120 — если $x>0.5$, то происходит перестановка значений аргументов (15);

130—190 — вычисление суммы степенного ряда, проверка условия сходимости (строка 180);

200—240 — вычисление значений гамма-функций;

250—290 — вычисление $I_x(p, q)$;

500—590 — подпрограмма вычисления $1/\Gamma(u)$;

1000—1010 — вывод $I_x(p, q)$.

В контрольных примерах найдены значения $I_{0.7}(0.5, 0.5)$ и $I_{0.2}(2, 1.5)$. Их табличные значения соответственно равны [7] 0.6309899, 0.6631506. Программа получена переводом на язык Бейсик Алгол-программы 179б [7].

```
JLIST
  10 PRINT "ОТНОШЕНИЕ НЕПОЛНЫХ БЕТА-ФУНКЦИЙ IX (P, Q)"
  20 INPUT "X=";X: INPUT "P,Q=";P,Q: X1=X
  30 E1=1E-8
 100 IF X>1 OR X<0 OR P<=0 OR Q<=0 THEN PRINT "НЕПРА
     ВИЛЬНЫЕ ЗНАЧЕНИЯ АРГУМЕНТОВ" : END
 110 IF X=0 OR X=1 THEN B=X: GOTO 1000
 120 W=0: IF X>.5 THEN T=P:P=Q:Q=T:X=1-X:W=1
 130 S2=0:R=1:T=1-X: Q1=Q:I=Q
 140 I=I-1: IF I<=0 THEN 170
 150 Q1=I:R=R*(Q1+1)/T/(P+Q1):S2=S2+R
 160 GOTO 140
 170 R=1: S1=1:I=0
 180 I=I+1: IF R<E1*S1 THEN 200
 190 R=R*X*(I-Q1)*(P+I-1)/I/(P+I):S1=S1+R: GOTO 180
 200 U=Q1: GOSUB 500:T=G: T1=G
 210 U=Q1+P: GOSUB 500:R=G: R1=G
 220 I=Q1
 230 IF I>Q-.5 THEN 250
 240 T1=T1*I: R1=R1*(I+P):I=I+1: GOTO 230
 250 T=X^P*(S1*R/P/T+S2*R1*(1-X)^Q/Q/T1)
 260 U=P: GOSUB 500
 270 T=T/G
 280 IF W=1 THEN B=1-T: GOTO 1000
 290 B=T: GOTO 1000
 500 G=1
 530 IF U>1 THEN U=U-1:G=G/U: GOTO 530
 550 IF U=1 THEN 590
 560 G1=(((((((((-.00000018122*U+.000001328554)*U-.000002625721)*
     U-.000017527917)*U+.000145624324)*U-.000360851496)*U
     -.000804341335)*U+.008023278113)*U-.017645242118)*U
     -.024552490887)*U
 570 G=G*((((G1+.191091101162)*U-.233093736365)*U-.422784335092)
     *U+1)*U*(1+U)
 580 G=1/G
 590 RETURN
1000 PRINT "I" X1"("P", "Q")="B
1010 END
JRUN
ОТНОШЕНИЕ НЕПОЛНЫХ БЕТА-ФУНКЦИЙ IX (P, Q)
X=.7
P,Q=.5,.5
I.7 (.5,.5)=.630989881
JRUN
ОТНОШЕНИЕ НЕПОЛНЫХ БЕТА-ФУНКЦИЙ IX (P, Q)
X=.2
P,Q=2,1.5
I.2 (2,1.5)=.0697957214
```

7.2. Некоторые интегральные функции

Вещественная интегральная функция $E_1(x)$ определяется как [3], [18].

$$E_1(x) = -Ei(-x) = \int_x^\infty \frac{e^{-t}}{t}\,dt. \qquad (1)$$

Согласно [18] $E_1(x)$ можно апроксимировать следующими выражениями:

при $0 < x < 1$

$$E_1(x) = -\ln x - C + C_1 x + \ldots + C_5 x^5 + \varepsilon(x), \qquad (2)$$

где $C = +0.57721566$, $C_1 = 0.99999193$,
$C_2 = -0.24991055$, $C_3 = 0.05519968$,
$C_4 = -9.76004 \cdot 10^{-3}$, $C_5 = 1.07857 \cdot 10^{-3}$,

$$|\varepsilon(x)| < 2 \cdot 10^{-7},$$

при $x \geqslant 1$

$$E_1(x) = \frac{e^{-x}}{x}\left[\frac{a_0 + a_1 x + a_2 x^2 + a_3 x^3 + x^4}{b_0 + b_1 x + b_2 x^2 + b_3 x^3 + x^4} + \varepsilon(x)\right], \qquad (3)$$

где $a_0 = 0.2677737343$, $a_1 = 8.6347608925$,
$a_2 = 18.0590169730$, $a_3 = 8.5733287401$,
$b_0 = 3.9584969228$, $b_1 = 21.0996530827$,
$b_2 = 25.6329561486$, $b_3 = 9.5733223454$,

$$|\varepsilon(x)| < 2 \cdot 10^{-8}.$$

В программе значения $E_1(x)$ вычисляются на основании формул (2) и (3).

В программе используются переменные: X — аргумент функции $E_1(x)$; $E1$ — рабочая переменная, значение $E_1(x)$.

Структура программы:

10—20 — ввод x;

110—120 — вычисления по формуле (3);

140 — вычисления по формуле (2);

1000—1010 — вывод значения $E_1(x)$.

Программа получена переводом на язык Бейсик Алгол-программы 20б [4].

В контрольном примере найдены значения $E_1(0.59)$, $E_1(10)$. Их значения по таблицам [18] равны: $E_1(0.59) = 0.463649849$, $E_1(10) = 0.091563339 \cdot e^{-10} = 0.4156969012 \cdot 10^{-5}$.

```
JLIST
  10 PRINT "ВЕЩЕСТВ. ИНТЕГР. ПОКАЗАТЕЛЬНАЯ ФУНКЦИЯ E1(X)"
  20 INPUT "X=";X
 100 IF X<1 THEN 140
 110 E1=(((X+8.5733287401)*X+18.059016973)*  X+8.6347608925)  *  X
     +.2677737343
 120 E1=EXP (-X)*E1/X/((((X+9.5733223454)*X+25.6329561486)*X
     +21.0996530827)*X+3.9584969228)
```

```
130 GOTO 1000
140 E1 = -LOG (X) -.57721566+ ((((.00107857 * X-.00976004) * X
    +.05519968) * X-.24991055) * X+.99999193) * X
1000 PRINT "E1 ("X") ="E1
1010 END

JRUN
ВЕЩЕСТВ. ИНТЕГР. ПОКАЗАТЕЛЬНАЯ ФУНКЦИЯ E1 (X)
X=.59
E1 (.59) =.463649764
JRUN
ВЕЩЕСТВ. ИНТЕГР. ПОКАЗАТЕЛЬНАЯ ФУНКЦИЯ E1 (X)
X=10
E1 (10) =4.15696901E-06
```

Комплексная интегральная показательная функция $E_k(z)$ определяется интегралом [18]:

$$E_k(z)=\int_1^\infty \frac{e^{-zt}}{t^k}\,dt \quad (k=0,\ 1,\ 2,\ \ldots,\ Re\ z>0). \tag{4}$$

В приведенной ниже программе вычисляется комплексная интегральная функция

$$W_k(z)=U+iV=z^k e^z E_k(z) \quad (z=x+iy). \tag{5}$$

Вычисления производятся на основании представления $W_k(z)$ в виде непрерывной дроби [18]:

$$W_k(z)=\cfrac{z}{z+\cfrac{k}{1+\cfrac{1}{z+\cfrac{k+1}{1+\cfrac{2}{z+\ldots}}}}} \tag{6}$$

Последовательные приближения вычисляются следующим образом [4]:

$$D_n=\frac{z}{z+mD_{n-1}},\quad R_n=(D_n-1)\,R_{n-1},\quad C_n=C_{n-1}+R_n, \tag{7}$$

где $D_1=R_1=C_1=1$, $R_n=a_n+ib_n$, $D_n=c_n+id_n$, $C_n=U_n+iV_n$,

$$m=\begin{cases} k-1+(n-2)/2, & n \text{ — четное},\\ (n-2)/2, & n \text{ — нечетное}.\end{cases}$$

Вычисления производятся до тех пор, пока

$$|C_n-C_{n-1}|>\varepsilon_1. \tag{8}$$

Метод вычислений применим на всей комплексной плоскости, кроме $z=0$ и отрицательной вещественной оси. Сходимость очень медленная при $|z|<0.05$ и для некоторой области внутри полуполосы $|y|<2$, $x<0$.

В программе используются переменные: K — действительное число k, X, Y — действительная и мнимая части аргумента $z=x+iy$; $E1$ — точность вычисления; U,

$V-W_k=u+iv$; C, D, W — рабочие переменные; A, B определяют предыдущее приближение для W_k.

Программа получена переводом на Бейсик Алгол-программы 14б [4].

В контрольных примерах найдены значения, совпадающие с точностью до шести цифр со значениями, приведенными в [4].

```
JLIST
  10 PRINT "КОМПЛ. ИНТЕГР. ПОКАЗАТЕЛЬНАЯ ФУНКЦИЯ W (K, X
     +I*Y)"
  20 INPUT "K=";K: INPUT "X, Y=";X, Y
  30 E1=1E-8
 100 E1=E1^2:A=1:C=1:U=1:N=1
 110 W=K-1:V=0:D=0:B=0
 120 N=N+1:R=INT (N/2)
 130 M=R: IF 2*R=N THEN M=R+W
 140 P=X+M*C:Q=Y+M*D
 150 M=P*P+Q*Q:C=(X*P+Y*Q)/M:
 160 D=(Y*P-X*Q)/M:P=C-1
 170 Q=A:A=A*P-B*D:B=Q*D+P*B:U=U+A:V=V+B
 180 IF (A*A+B*B)/(U*U+V*V)>E1 THEN 120
1000 PRINT "W ("X"+I*"Y", "K") ="U"+I*"V
1010 END

JRUN
КОМПЛ. ИНТЕГР. ПОКАЗАТЕЛЬНАЯ ФУНКЦИЯ W (K, X+I*Y)
K=1
X, Y=1,1
W (1+I*1,1) =.673321227+I*.147863858
JRUN
КОМПЛ. ИНТЕГР. ПОКАЗАТЕЛЬНАЯ ФУНКЦИЯ W (K, X+I*Y)
K=2
X,Y=4,0
W (4+I*0,2) =.698469604+I*0
```

Интегральный синус *Si (x)* действительного аргумента определяется интегралом [3], [18]:

$$Si\,(x)=\int_0^x \frac{\sin t}{t}\,dt. \tag{9}$$

$Si\,(x)$ разлагается в ряд

$$Si\,(x)=\sum_{n=0}^{\infty} \frac{(-1)^n\,x^{2n+1}}{(2n+1)\,(2n+1)!}, \tag{10}$$

а в области больших значений $x\geqslant 1$ апроксимируется рациональными функциями [18]:

$$Si\,(x)=\frac{\pi}{2}-f\,(x)\cos x-g\,(x)\sin x, \tag{11}$$

где

$$f(x)=\frac{1}{x}\,\frac{x^8+a_1x^6+a_2x^4+a_3x^2+a_4}{x^8+b_1x^6+b_2x^4+b_3x^2+b_4}+\varepsilon(x), \qquad (12)$$

$$|\varepsilon(x)|<5\cdot10^{-7},$$

$a_1=38.027264,$ $b_1=40.021433,$
$a_2=265.187033,$ $b_2=322.624911,$
$a_3=335.677320,$ $b_3=570.236280,$
$a_4=38.102495,$ $b_4=157.105423,$

$$g(x)=\frac{1}{x^2}\,\frac{x^8+a_1x^6+a_2x^4+a_3x^2+a_4}{x^8+b_1x^6+b_2x^4+b_3x^2+b_4}+\varepsilon(x), \qquad (13)$$

$$|\varepsilon(x)|<3\cdot10^{-7},$$

$a_1=42.242855,$ $b_1=48.196927,$
$a_2=302.757865,$ $b_2=482.485984,$
$a_3=352.018498,$ $b_3=1114.978885,$
$a_4=21.821899,$ $b_4=449.690326.$

В программе значения $Si(x)$ вычисляются на основании формул (10) и (11) при $0<x<1$ и $x\geqslant1$ соответственно. Точность вычисления для $x<1$ меньше $3\cdot10^{-9}$, при $x\geqslant1$ меньше чем $8\cdot10^{-7}$.

В программе используются переменные: $X1$ — начальное значение аргумента $Si(x)$; X — рабочая переменная (x или x^2); S — функции f и g, значение $Si(x)$.

Структура программы:

10—20 — ввод x;

110—150 — вычисления по формуле (11);

160—170 — вычисления по формуле (10) с пятью слагаемыми;

1000—1010 — вывод значения $Si(x)$.

В контрольных примерах вычислены значения $Si(0.5)$ и $Si(10)$. Их значения по таблицам [18] равны: $Si(0.5)=0.4931074180$, $Si(10)=1.6583475942$.

```
JLIST
  10 PRINT "ИНТЕГРАЛЬНЫЙ СИНУС SI (X)"
  20 INPUT "X=";X: X1=X
 100 IF X<1 THEN 160
 110 S=1/X:X=X*X
 120 S=S*((((X/100+.38027264)*X+2.65187033)*X+3.3567732)*X
     +.38102495)/((((X/100+.40021433)*X+3.22624911)*X+5.7023628)*X
     +1.57105423)
 130 S=1.570796326-S*COS(X1)
 140 S=S-SIN(X1)/X*((((X/100+.42242855)*X+3.02757865)*X+3.52018
     498)*X+.21821899)/((((X/100+.48196927)*X+4.82485984)*X+11.1497
     8885)*X+4.49690326)
 150 GOTO 1000
 160 S=X:X=X*X
 170 S=S*((((X/3265920-2.834467E-5)*X+1.66666666E-3)*X-.0555555
     556)*X+1)
```

```
1000 PRINT "SI ("X1") ="S
1010 END

JRUN
ИНТЕГРАЛЬНЫЙ СИНУС SI (X)
X=.5
SI (.5) =.493107418
JRUN
ИНТЕГРАЛЬНЫЙ СИНУС SI (X)
X=10
SI (10) =1.65834786
```

Интегральный косинус $Ci(x)$ действительного аргумента определяется интегралом [3], [18]:

$$Ci(x)=C+\ln x+\int_0^x \frac{\cos t-1}{t}\,dt. \tag{14}$$

Функция $Ci(x)$ представляется в виде ряда

$$Ci(x)=C+\ln x+\sum_{n=1}^{\infty} \frac{(-1)^n x^{2n}}{2n\,(2n)!}. \tag{15}$$

В области больших значений $x\geqslant 1$ $Ci(x)$ апроксимируется рациональными функциями [18]:

$$Ci(x)=f(x)\sin x-g(x)\cos x, \tag{16}$$

где функции $f(x)$ и $g(x)$ определены в (12), (13).

В программе $Ci(x)$ вычисляется на основании формул (15) и (16) при $0<x<1$ и $x\geqslant 1$ соответственно. Точность вычислений для $x<1$ меньше чем $3\cdot 10^{-8}$, для $x\geqslant 1$ меньше чем $8\cdot 10^{-7}$.

В программе используются переменные: $X1$ — начальное значение аргумента $Ci(x)$; X — рабочая переменная (x или x^2); C — функции f, g, значение $Ci(x)$.

Структура программы:

10—20 — ввод x;

110—150 — вычисления по формуле (16);

160—170 — вычисления по формуле (15);

1000—1010 — вывод значения $Ci(x)$.

В контрольном примере вычислены значения $Ci(0.5)$ и $Ci(10)$. Их значения по таблицам [18] равны: $Ci(0.5)=-0.1777840788$, $Ci(10)=-0.0454564330$.

```
JLIST
  10 PRINT "ИНТЕГРАЛЬНЫЙ КОСИНУС CI (X)"
  20 INPUT "X=": X: X1=X
 100 IF X<1 THEN 160
 110 C=1/X:X=X*X
 120 C=C*((((X/100+.38027264)*X+2.65187033)*X+3.3567732)*X+.381
     02495)/((((X/100+.40021433)*X+3.22624911)*X+5.7023628)*X+1.571
     05423)
 130 C=C*SIN (X1)
```

```
 140 C=C-COS (X1)/X*((((X/100+.42242855)*X+3.02757865)*X+3.5201
     8498)*X+.21821899)/((((X/100+.48196927)*X+4.82485984)*X+11.149
     78885)*X+4.49690326)
 150 GOTO 1000
 160 C=.577215664+LOG (X):X=X*X
 170 C=C+(((X/322560-2.3148148E-4)*X+.010416666667)*X-.25)*X
1000 PRINT "CI ("X1") ="C
1010 END

JRUN
ИНТЕГРАЛЬНЫЙ КОСИНУС CI (X)
X=.5
CI (.5) = -.177784079
JRUN
ИНТЕГРАЛЬНЫЙ КОСИНУС CI (X)
X=10
CI (10) = -.045456288
```

7.3. Ортогональные полиномы

Полиномы Чебышева 1-го рода $T_n(x)$ и 2-го рода $U_n(x)$ определяются как [3], [19]

$$T_n(x)=\cos(n \arccos x),$$
$$U_n(x)=\sin(n \arccos x). \qquad (1)$$

Функции T_n и U_n удовлетворяют рекуррентной формуле

$$W_{n+1}=2xW_n(x)-W_{n-1}(x), \qquad (2)$$

причем, как следует из (1),

$$T_0(x)=1, \qquad T_1(x)=x,$$
$$U_0(x)=0, \qquad U_1(x)=\sqrt{1-x^2}.$$

По первой программе вычисляются значения полинома Чебышева $T_n(x)$ на основании формулы (2).

В программе используются переменные: N, X соответственно n и x; T, B, A — значения полиномов T_{n+1}, T_n, T_{n-1} в формуле (2); I — переменные цикла.

Структура программы:

10—20 — ввод n, x;

100—130 — вычисления по формуле (2);

1000—1010 — вывод значения $T_n(x)$.

Программа получена переводом на язык Бейсик Алгол-программы 10б [4].

В контрольном примере вычислено значение $T_3(0.25)=-0.6875$

```
JLIST
  10 PRINT "ПОЛИНОМ ЧЕБЫШЕВА TN (X)"
  20 INPUT "N="; N: INPUT "X="; X
 100 A=1:B=X
 110 IF N=0 THEN T=A: GOTO 1000
```

```
 120 IF N=1 THEN T=B: GOTO 1000
 130 FOR I=2 TO N:T=2*X*B-A:A=B:B=T: NEXT
1000 PRINT "T"N" ("X") ="T
1010 END

JRUN
ПОЛИНОМ ЧЕБЫШЕВА TN (X)
N=3
X=.25
T3 (.25) = -.6875
```

По второй программе вычисляется значение полинома Чебышева 2-го рода $U_n(x)$. Обозначения те же, что и в предыдущей программе, за исключением замены T на U.

В контрольном примере вычислено значение $U_5(0.20)$, совпадающее с результатом ручного счета ($0.21824\sqrt{6} = 0.534576642...$).

```
JLIST
  10 PRINT "ПОЛИНОМ ЧЕБЫШЕВА II-РОДА UN (X)"
  20 INPUT "N="; N: INPUT "X="; X
 100 A=0:B=SQR (1-X*X)
 110 IF N=0 THEN U=A: GOTO 1000
 120 IF N=1 THEN U=B: GOTO 1000
 130 FOR I=2 TO N:U=2*X*B-A:A=B:B=U: NEXT
1000 PRINT "U"N" ("X") ="U
1010 END

JRUN
ПОЛИНОМ ЧЕБЫШЕВА II-РОДА UN (X)
N=5
X=.2
U5 (.2) = .534576642
```

По третьей программе производится вычисление коэффициентов полинома Чебышева 1-го рода $T_n(x)$ на основании явного выражения для многочлена $T_n(x)$ [3]:

$$T_0(x)=1,\ T_n(x)=\sum_{m=0}^{k} a_m x^{n-2m}\ (n>0), \tag{3}$$

где

$$k=\operatorname{int}(n/2),\ a_m=(-1)^m 2^{n-2m-1}\frac{n(n-m-1)!}{m!(n-2m)!}\ (n>0).$$

Из (3) следует рекуррентное соотношение, на основании которого производятся вычисления коэффициентов a_m:

$$a_{m+1}=-a_m\frac{(n-2m-1)(n-2m)}{4(m+1)(n-m-1)}, \tag{4}$$

причем

$$a_0=\begin{cases}1\ (n=0),\\ 2^{n-1}\ (n>0).\end{cases}$$

В программе используются переменные: N — порядок полинома $T_n(x)$; M — переменная цикла в (4); A (INT($N/2$)) — массив коэффициентов a_m.

Структура программы:

10—20 — ввод n;

100—140 — вычисления по формуле (4);

1000—1050 — вывод полинома $T_n(x)$.

В контрольном примере показано, что $T_5(x) = 16x^5 - 20x^3 + 5x$.

```
JLIST
  10 PRINT "КОЭФФ. ПОЛИНОМА ЧЕБЫШЕВА TN (X)"
  20 INPUT "N=": N: DIM A (INT (N/2))
 100 IF N=0 OR N=1 THEN A (0)=1: GOTO 1000
 110 A (0)=2^(N-1)
 120 FOR M=0 TO INT (N/2)-1
 130 A (M+1)=-A (M) * (N-2*M) * (N-2*M-1)/4/(M+1)/(N-M-1)
 140 NEXT M
1000 PRINT "T"N" (X)=";
1010 FOR M=0 TO INT (N/2)
1020 IF A (M)>0 AND M>0 THEN PRINT "+";
1030 PRINT A (M);: IF N>2*M THEN PRINT "X^"N-2*M;
1040 NEXT
1050 END

JRUN
КОЭФФ. ПОЛИНОМА ЧЕБЫШЕВА TN (X)
N=5
T5 (X)=16X^5-20X^3+5X^1
```

Полиномы Эрмита (функции параболического цилиндра) являются решением дифференциального уравнения [3], [18], [19]

$$W'' - 2zW' + 2nW = 0. \tag{5}$$

Они равны

$$H_n(x) = (-1)^n e^{x^2} \frac{d^n}{dx^n}(e^{-x^2}) \tag{6}$$

и удовлетворяют следующим рекуррентным соотношениям:

$$\begin{gathered} H_{n+1}(x) = 2xH_n(x) - 2nH_{n-1}(x), \\ H_0(x) = 1,\ H_1(x) = 2x, \\ \frac{dH_n(x)}{dx} = 2nH_{n-1}(x). \end{gathered} \tag{7}$$

В первой программе значения полинома Эрмита вычисляются на основании соотношения (7).

В программе используются переменные: X, N — аргумент и степень $H_n(x)$; H, B, A — значения H_{n+1}, H_n и H_{n-1} соответственно; I — переменная цикла.

Структура программы:

10—20 — ввод n и x;

100—130 — вычисления по формуле (7);

1000—1010 — вывод значения $H_n(x)$.

Программа получена переводом на Бейсик Алгол-программы 11 б [4].

В контрольном примере вычислено значение $H_5(0.5)$, совпадающее с табличным значением.

```
JLIST
  10 PRINT "ПОЛИНОМ ЭРМИТА HN (X)"
  20 INPUT "N=";N: INPUT "X=";X
 100 A=1:B=2*X:N=N-1
 110 IF N=-1 THEN H=A: GOTO 1000
 120 IF N=0 THEN H=B: GOTO 1000
 130 FOR I=1 TO N:H=2*(X*B-I*A):A=B:B=H: NEXT
1000 PRINT "H"N+1" ("X")="H
1010 END

JRUN
ПОЛИНОМ ЭРМИТА HN (X)
N=5
X=.5
H5 (.5)=41
```

Во второй программе вычисляются коэффициенты a_m полинома Эрмита [3]:

$$H_n(x)=\sum_{m=0}^{k} a_m x^{n-2m},$$

где

$$a_m=(-1)^m n!\,\frac{2^{n-2m}}{m!\,(n-2m)!}, \qquad (8)$$

$$k=\operatorname{int}(n/2).$$

Из (8) следует, что

$$a_{m+1}=-a_m\,\frac{(n-2m)(n-2m-1)}{4(m+1)},\quad a_0=2^n. \qquad (9)$$

В программе используются переменные: N — порядок полинома $H_n(x)$; M — переменная цикла в (9); A (int $(N/2)$) — массив коэффициентов a_m.

Структура программы:

10—20 — ввод n;

100—130 — вычисления по формуле (9);

1000—1050 — вывод полинома $H_n(x)$.

В контрольном примере показано, что $H_5(x)=32x^5-160x^3+120x$.

```
JLIST
  10 PRINT "КОЭФФ. ПОЛИНОМА ЭРМИТА HN (X)"
  20 INPUT "N=";N: DIM A (INT (N/2))
 100 A (0)=2^N: IF N=0 THEN 1000
 110 IF N=1 THEN A (0)=2: GOTO 1000
 120 FOR M=0 TO INT (N/2)-1
 130 A (M+1)=-A (M)*(N-2*M)*(N-2*M-1)/4/(M+1)
```

```
140 NEXT M
1000 PRINT "H"N" (X)=";
1010 FOR M=0 TO INT (N/2)
1020 IF A (M)>0 AND M>0 THEN PRINT "+";
1030 PRINT A (M);: IF N>2*M THEN PRINT "X^"N-2*M;
1040 NEXT
1050 END
JRUN
КОЭФФ. ПОЛИНОМА ЭРМИТА HN (X)
N=5
H5 (X)=32X^5-160X^3+120X^1
```

Полиномы Лежандра $P_n(x)$ 1-го рода степени n могут быть определены как [3], [19]

$$P_n(x)=\frac{1}{2^n n!}\frac{d^n}{dx^n}(x^2-1)^n. \tag{10}$$

Они удовлетворяют рекуррентному соотношению

$$P_{n+1}(x)=\frac{2n+1}{n+1}xP_n(x)-\frac{n}{n+1}P_{n-1}(x), \tag{11}$$

$$P_0(x)=1,\ P_1(x)=x$$

и могут быть представлены в виде

$$P_n(x)=\sum_{m=0}^{k} a_m x^{n-2m}, \tag{12}$$

где $k=\text{int}(n/2)$, $a_m=(-1)^m\frac{(2n-2m)!}{2^n m!\,(n-m)!\,(n-2m)!}$

Из выражения для a_m следует, что

$$a_{m+1}=-a_m\frac{(n-2m)(n-2m-1)}{2(2n-2m-1)(m+1)}, \tag{13}$$

$$a_0=\frac{(n+1)(n+2)\ldots 2n}{2^n n!}.$$

В первой программе значения полинома Лежандра $P_n(x)$ вычисляются на основании рекуррентного соотношения (11).

В программе используются переменные: X, N — аргумент и степень $P_n(x)$; P, B, A соответственно значения P_{n+1}, P_n и P_{n-1} в (11); I — переменная цикла.

Структура программы:

10—20 — ввод n и x;

100—150 — вычисления по формуле (2);

1000—1010 — вывод значения $P_n(x)$.

Программа получена переводом на язык Бейсик Алгол-программы 13б [4].

В контрольном примере вычислено значение $P_5(0.8)=-0.39952$.

```
JLIST
  10 PRINT "ПОЛИНОМ ЛЕЖАНДРА PN (X)"
  20 INPUT "N=";  N: INPUT "X=";X
```

```
 100 A=1:B=X
 110 IF N=0 THEN P=A: GOTO 1000
 120 IF N=1 THEN P=B: GOTO 1000
 130 FOR I=2 TO N
 140 P=B*X+(I-1)/I*(B*X-A):A=B:B=P
 150 NEXT
1000 PRINT "P"N" ("X")="P
1010 END

JRUN
ПОЛИНОМ ЛЕЖАНДРА PN (X)
N=5
X=.8
P5 (.8)=-.39952
```

Во второй программе вычисляются коэффициенты a_m полинома Лежандра $P_n(x)$ на основании соотношения (13).

В программе используются переменные: N — степень полинома $P_n(x)$; M — переменная цикла m.

Структура программы:

10—20 — ввод n;

100—120 — вычисление коэффициента a_0;

130—150 — вычисление коэффициентов a_m по формуле (13);

1000—1050 — вывод полинома $P_n(x)$.

В контрольных примерах показано, что $P_4(x)=35/8x^4-15/4x^2+3/8$ и $P_6(x)=231/16x^6-315/16x^4+105/16x^2-5/16$.

```
JLIST
  10 PRINT "КОЭФФИЦИЕНТЫ ПОЛИНОМА ЛЕЖАНДРА PN (X)"
  20 INPUT "N="; N: DIM A (INT (N/2))
 100 A (0)=1
 110 IF N=0 OR N=1 THEN 1000
 120 FOR M=1 TO N:A (0)=A (0)*(N+M)/2/M: NEXT
 130 FOR M=0 TO INT (N/2)-1
 140 A (M+1)=-A (M)*(N-2*M)*(N-2*M-1)/2/(M+1)/(2*(N
     -M)-1)
 150 NEXT M
1000 PRINT "P"N" (X)=";
1010 FOR M=0 TO INT (N/2)
1020 IF A (M)>0 AND M>0 THEN PRINT "+";
1030 PRINT A (M);: IF N>2*M THEN PRINT "*X^"N-2*M;
1040 NEXT
1050 END

JRUN
КОЭФФИЦИЕНТЫ ПОЛИНОМА ЛЕЖАНДРА PN (X)
N=4
P4 (X)=4.375*X^4-3.75*X^2+.375
```

```
JRUN
КОЭФФИЦИЕНТЫ ПОЛИНОМА ЛЕЖАНДРА PN (X)
N=6
P6 (X) = 14.4375 * X^6 - 19.6875 * X^4 + 6.5625 * X^2 - .3125
```

Полиномы Лежандра $Q_n(x)$ ($|x|<1$) **2-го рода** удовлетворяют тем же рекуррентным соотношениям (11), причем (см. [3], [19]):

$$Q_0(x)=\frac{1}{2}\ln\frac{1+x}{1-x},\quad Q_1(x)=\frac{x}{2}\ln\frac{1+x}{1-x}-1. \tag{14}$$

В нижеследующей программе вычисляются значения $Q_n(x)$ на основании формул (11) и (14). При этом используются переменные те же, что и в первой программе.

В контрольном примере показано, что $Q_2(0.5)$ и $Q_9(0.8)$ совпадают с табличными: $Q_2(0.5)=-\frac{1}{16}\ln 3-\frac{3}{4}=$ $=-0.818663268\ldots$, $Q_9(0.8)=0.43299312\ldots$.

```
JLIST
  10 PRINT "ПОЛИНОМ ЛЕЖАНДРА QN (X)"
  20 INPUT "N="; N: INPUT "X="; X
 100 Q=LOG ((1+X)/(1-X))/2: IF N=0 THEN 1000
 110 A=Q:Q=Q*X-1: IF N=1 THEN 1000
 130 B=Q
 140 FOR I=2 TO N
 150 Q=B*X+(I-1)/I*(B*X-A):A=B:B=Q
 160 NEXT
1000 PRINT "Q"N" ("X") ="Q
1010 END

JRUN
ПОЛИНОМ ЛЕЖАНДРА QN (X)
N=2
X=.5
Q2 (.5) = -.818663268
JRUN
ПОЛИНОМ ЛЕЖАНДРА QN (X)
N=9
X=.8
Q9 (.8) = .432993123
```

Присоединенные функции Лежандра $P_n^m(x)$ **1-го рода** определяются при $|x|<1$ как [3], [19]

$$P_n^m(x)=(1-x^2)^{m/2}\frac{d^mP_n}{dx^m}. \tag{15}$$

Таким же соотношением определяются присоединенные функции Лежандра в комплексной плоскости, разрезанной между точками -1 и 1. В программе вычисляется значение функции Лежандра $P_n^m(x)$ (если $r=1$) или значение $|P_n^m(ix)|$ (если $r=0$) при $m\leqslant n$, $|x|<1$.

В программе используются переменные: R задает вид аргумента x; X — аргумент P_n^m; M, N — параметры m и n; Z, W, Y — рабочие переменные; P — значение $P_n^m(x)$ или $|P_n^m(ix)|$; I, J — переменные циклов; $G(2N+1)$ — рабочий массив; $K=M-N$ — рабочая переменная.

Структура программы:

10—20 — ввод N, M, X;

100—270 — вычисление значения P_n^m;

1000—1020 — вывод $P_n^m(x)$ или $|P_n^m(ix)|$.

Программа получена переводом на язык Бейсик Алгол-программы 47 б [4].

В контрольном примере вычислены значения $P_4^2(0.5)=135/12=4.21875$ и $|P_4^3(ix)|=34.335\sqrt{1.09}=35.8467924\ldots$.

```
JLIST
  10 PRINT "ПРИСОЕДИНЕННЫЕ Ф-И ЛЕЖАНДРА I-РОДА PMN (X)"
  20 INPUT "M,N=";M,N: DIM G (2*N+1)
  30 PRINT "ЕСЛИ АРГУМЕНТ МНИМЫЙ R=0, ИНАЧЕ R=1": INPUT
     "R=";R: INPUT "X=";X
 100 P=0:K=N-M: IF N<M THEN 1000
 110 IF N=0 THEN P=1: GOTO 1000
 120 G (1)=1:J=2*N
 130 FOR I=1 TO J:G (I+1)=I*G (I): NEXT
 140 J=INT (K/2)
 150 IF X<>0 THEN 190
 160 I=J: IF K<>2*I THEN 1000
 170 P=G (M+N+1)/G (I+1)/G (M+I+1): IF R=1 THEN P=P*(-1)^I
 180 GOTO 240
 190 Z=X^K:Y=1/X/X:IF R=1 THEN Y=-Y
 200 I=1
 210 K=N-I: IF J+1<I THEN 240
 220 P=P+G (2*K+3)*Z/G (I)/G (K+2))/G (K-I-M+3)
 230 Z=Z*Y:I=I+1: GOTO 210
 240 P=P/2^N:W=-1
 250 IF R=1 THEN 270
 260 W=1:I=N-4*INT (N/4): IF I>1 THEN P=-P
 270 Z=SQR (ABS (W+X*X)):P=P*Z^M
1000 PRINT "P"M", "N" ("X;: IF R=0 THEN PRINT"*I";
1010 PRINT ")="P
1020 END

JRUN
ПРИСОЕДИНЕННЫЕ Ф-И ЛЕЖАНДРА I-РОДА PMN (X)
M,N=2,4
ЕСЛИ АРГУМЕНТ МНИМЫЙ R=0, ИНАЧЕ R=1
R=1
X=.5
P2,4 (.5)=4.21875
JRUN
ПРИСОЕДИНЕННЫЕ Ф-И ЛЕЖАНДРА I-РОДА PMN (X)
```

```
M,N=3,4
ЕСЛИ АРГУМЕНТ МНИМЫЙ R=0, ИНАЧЕ R=1
R=0
X=.3
P3,4 (.3 * I) =35.8467925
```

Полиномы Лагерра $L_n(x)$ согласно определению [3] задаются выражением

$$L_n(x)=e^x\frac{d^n}{dx^n}(e^{-x}x^n). \tag{16}$$

В программе вычисление значения $L_n(x)$ производится на основании рекуррентного соотношения

$$\begin{aligned} &L_i(x)=(2i-1-x)\,L_{i-1}(x)-(i-1)^2\,L_{i-2}(x),\\ &L_0(x)=1,\ L_1(x)=1-x. \end{aligned} \tag{17}$$

В программе используются переменные: N, X — порядок и аргумент $L_n(x)$; L, B, A — значения L_{n+1}, L_n, L_{n-1} в (17) соответственно; I — переменная цикла.

Структура программы:

10—20 — ввод n, x;

100—150 — вычисления по (17);

1000—1010 — вывод значения $L_n(x)$.

Программа получена переводом на язык Бейсик Алгол-программы 12 б [4].

В контрольных примерах показано, что $L_4(1)=-15$, $L_2(0.5)=0.25$.

```
JLIST
  10 PRINT "ПОЛИНОМ ЛАГЕРРА LN (X)"
  20 INPUT "N=";N: INPUT "X="; X
 100 A=1: IF N=0 THEN L=A: GOTO 1000
 110 B=1-X: IF N=1 THEN L=B: GOTO 1000
 120 FOR I=2 TO N
 130 L=(2*I-1-X)*B-(I-1)*(I-1)*A:A=B:B=L
 140 NEXT I
1000 PRINT "L"N" ("X") ="L
1010 END

JRUN
ПОЛИНОМ ЛАГЕРРА LN (X)
N=4
X=1
L4 (1) = -15
JRUN
ПОЛИНОМ ЛАГЕРРА LN (X)
N=2
X=.5
L2 (.5) =.25
```

Во второй программе вычисляются коэффициенты полинома Лагерра на основании явного выражения для $L_n(x)$ [3]:

$$L_n(x)=\sum_{m=0}^{n} a_m x^m, \tag{18}$$

где $a_m=\frac{(-1)^m (n!)^2}{(m!)^2 (n-m)!}$.

Из (17) следует, что

$$a_{m+1}=-a_m \frac{n-m}{(m+1)^2}, \quad a_0=n!. \tag{19}$$

В программе используются переменные: N — порядок полинома $L_n(x)$; M — переменная цикла в (19) и (18); $A(N)$ — массив коэффициентов a_m; B — рабочая переменная.

Структура программы:

10—20 — ввод n;

100—150 — вычисления a_m по формуле (19);

1000—1010 — вывод полинома $L_n(x)$.

В контрольном примере показано, что $L_4(x)=24-96x+72x^2-16x^3+x^4$.

```
JLIST
  10 PRINT "КОЭФФИЦИЕНТЫ ПОЛИНОМА ЛАГЕРРА LN (X)"
  20 INPUT "N=";N: DIM A (N)
 100 B=1: IF N=0 THEN A (0)=1: GOTO 1000
 110 FOR M=1 TO N:B=B*M: NEXT M
 120 A (0)=B
 130 FOR M=0 TO N-1
 140 B=-B*(N-M)/(M+1)/(M+1):A (M+1)=B
 150 NEXT
1000 PRINT "L"N" (X)=";
1010 FOR M=0 TO N
1020 IF A (M)>=0 AND M>0 THEN PRINT "+";
1030 PRINT A (M);: IF M>0 THEN PRINT "X^"M;
1040 NEXT
1050 END

JRUN
КОЭФФИЦИЕНТЫ ПОЛИНОМА ЛАГЕРРА LN (X)
N=4
L4 (X)=24-96X^1+72X^2-16X^3+1X^4
```

7.4. Эллиптические интегралы

Полный эллиптический интеграл 1-го рода $K(m)$ в нормальной форме определяется как [3], [18]

$$K_m=\int_0^1 \frac{dt}{\sqrt{(1-t^2)(1-mt^2)}} \quad (0\leqslant m<1). \tag{1}$$

Значение $K(m)$ может быть апроксимировано многочленом [18]:

$$K(m)=a_0+a_1m_1+a_2m_1^2+a_3m_1^3+a_4m_1^4- \\ -(b_0+b_1m_1+b_2m_1^2+b_3m_1^3+b_4m_1^4)\ln m_1+\varepsilon(m), \qquad (2)$$

где $m_1=1-m$, $|\varepsilon(m)|\leqslant 2\cdot 10^{-8}$,

$a_0=1.38629436112$, $b_0=0.5$,
$a_1=0.09666344259$, $b_1=0.12498593597$,
$a_2=0.03590092383$, $b_2=0.06880248576$,
$a_3=0.03742563713$, $b_3=0.03328355346$,
$a_4=0.01451196212$, $b_4=0.00441787012$.

В программе вычисление $K(m)$ производится на основании соотношения (2).

В программе используются переменные: M — значение m; K — значение $K(m)$.

Структура программы:

10—20 — ввод m;

100—110 — вычисление по формуле (2);

1000—1010 — вывод значения $K(m)$.

В контрольном примере найдено значение $K(0.44)$. Табличное значение [18] $K(0.44)=1.80632756$.

```
JLIST
  10 PRINT "ПОЛНЫЙ ЭЛЛИПТИЧЕСКИЙ ИНТЕГРАЛ ПЕРВОГО РО-
     ДА K (M)"
  20 INPUT "M=";M
 100 M=1-M
 110 K=(((.01451196212*M+.03742563713)*M+.03590092383)*M
     +.09666344259)*M+1.38629436112-LOG (M)*((((.00441787012*M
     +.03328355346)*M+.06880248576)*M+.12498593597)*M+.5)
1000 PRINT "K ("1-M") ="K
1010 END

JRUN
ПОЛНЫЙ ЭЛЛИПТИЧЕСКИЙ ИНТЕГРАЛ ПЕРВОГО РОДА K (M)
M=.44
K (.44) =1.80632757
```

Полный эллиптический интеграл 2-го рода $E(m)$ определяется как интеграл [3], [18]

$$E(m)=\int_0^1\sqrt{\frac{1-mt^2}{1-t^2}}\,dt \quad (0\leqslant m<1). \qquad (3)$$

Значение $E(m)$ может быть апроксимировано многочленом [18]:

$$E(m)=1+a_1m_1+a_2m_1^2+a_3m_1^3+a_4m_1^4- \\ -(b_1m_1+b_2m_1^2+b_3m_1^3+b_4m_1^4)\ln m_1+\varepsilon(m), \qquad (4)$$

где $m_1=1-m$, $|\varepsilon(m)|<2\cdot 10^{-8}$,

$a_1=0.44325141463$, $b_1=0.24998368310$,
$a_2=0.06260601220$, $b_2=0.09200180037$,

$a_3 = 0.04757383546$, $b_3 = 0.04069697526$,
$a_4 = 0.01736506451$, $b_4 = 0.00526449639$.

В программе вычисление $E(m)$ производится на основании (4).

В программе используются переменные: M — значение m; E — значение $E(m)$.

Структура программы:

10—20 — ввод m;

100—110 — вычисления по формуле (4);

1000—1010 — вывод значения $E(m)$.

В контрольном примере найдено значение $E(0.44)$. Табличное значение [18] $E(0.44) = 1.38025$.

```
JLIST
  10 PRINT "ПОЛНЫЙ ЭЛЛИПТИЧЕСКИЙ ИНТЕГРАЛ ВТОРОГО РО-
     ДА E (M)"
  20 INPUT "M=": M
 100 M=1-M
 110 E=(((.01736506451 * M+.04757383546) * M+.06260601220) * M
     +.44325141463) * M+1-LOG (M) * M * (((.00526449639 * M
     +.04069697526) * M+.09200180037) * M+.24998368310)
1000 PRINT "E ("1-M") ="E
1010 END

JRUN
ПОЛНЫЙ ЭЛЛИПТИЧЕСКИЙ ИНТЕГРАЛ ВТОРОГО РОДА E (M)
M=.44
E (.44) = 1.38025879
```

Полные эллиптические интегралы 1-го и 2-го родов $K(m_1)$ и $E(m_1)$, могут быть записаны в виде

$$K(m_1) = \int_0^{\pi/2} (1 - m \sin^2 \varphi)^{-1/2} d\varphi, \qquad (5)$$

$$E(m_1) + \int_0^{\pi/2} \sqrt{1 - m \sin^2 \varphi}\, d\varphi,$$

где $m = 1 - m_1 = k^2 = 1 - \cos^2 \alpha$.

Они вычисляются с помощью арифметико-геометрического среднего [18].

Процесс начинается с задания тройки чисел $a_0 = 1$, $b_0 = \cos \alpha$, $c_0 = \sin \alpha$. Затем вычисляются значения a_i, b_i, такие, что

$$a_i = \frac{1}{2}(a_{i-1} + b_{i-1}),\ b_i = \sqrt{a_{i-1} b_{i-1}}\ (i = \overline{1,\ n}). \qquad (6)$$

Процесс заканчивается на n-м шаге, когда $a_n = b_n$ (с требуемой относительной точностью ε_1). Тогда

$$K(\alpha) = \frac{\pi}{2a_n},$$

$$E(\alpha) = [1 - \frac{1}{2}(c_0^2 + 2c_1^2 + 4c_2^2 + \ldots + 2^n c_n^2)] \cdot K(\alpha), \qquad (7)$$

где $c_i=\frac{1}{2}(a_{i-1}-b_{i-1})$, $i=\overline{1, n}$.

В программе используются переменные: $M1$, K, E соответственно значения m_1, $K(m_1)$, $E(m_1)$; $E1$ — относительная точность вычислений; A, B, C соответственно значения a_i, b_i, c_i; S — сумма $\sum_{i=0}^{n} 2^i c_i^2$; I — рабочая переменная цикла.

Структура программы:

10—20 — ввод m_1, задание ε_1;

100 — проверка правильности значения параметра m_1 ($0<m_1\leqslant 1$);

110—140 — вычисления по формулам (6);

150 — вычисление значений $K(m_1)$ и $E(m_1)$ по формулам (7);

1000—1010 — вывод значений $K(m_1)$ и $E(m_1)$.

Программа получена переводом на язык Бейсик Алгол-программы 165б [7].

В контрольном примере вычислены значения $K(m_1=0.6)$ и $E(m_1=0.6)$. Их значения по таблицам [18] соответственно равны 1.777519371491253 и 1.399392139.

```
JLIST
 10 PRINT "ЭЛЛИПТИЧЕСКИЕ ИНТЕГРАЛЫ ПЕРВОГО И ВТОРОГО РОДОВ K (M1) И E (M1)"
 20 INPUT "M1=": M1: E1=1E-8
100 IF M1>1 OR M1<=0 THEN PRINT "ЗНАЧЕНИЕ M1 НЕПРАВИЛЬНОЕ!": END
110 A=1:I=1:B=SQR (M1):T=1-M1:S=0
120 S=S+T:C=(A-B) / 2:I=2*I:T=(A+B) / 2
130 B=SQR (A*B):A=T:T=I*C*C
140 IF (ABS (C)>E1*A) OR (T>E1*S) THEN 120
150 K=3.14159265359 / (A+B):S=S+T:E=K*(1-S / 2)
1000 PRINT "K ("M1")="K: PRINT "E (" M1")="E
1010 END

JRUN
ЭЛЛИПТИЧЕСКИЕ ИНТЕГРАЛЫ ПЕРВОГО И ВТОРОГО РОДОВ K (M1) И E (M1)
M1=.6
K (.6)=1.77751937
E (.6)=1.39939214
```

7.5. Функции Бесселя целого порядка

Функции Бесселя являются решениями дифференциального уравнения [3], [18]

$$z^2w''+zw'+(z^2-\nu^2)\,w=0, \quad z=x+iy. \tag{1}$$

Функцией Бесселя 1-го рода называется такое решение уравнения (1), которое для произвольного порядка ν и аргумента z имеет представление

$$J_\nu(z)=\left(\frac{z}{2}\right)^\nu \sum_{k=0}^{\infty} \frac{(-1)^k (z/2)^{2k}}{k!\Gamma(k+\nu+1)}. \qquad (2)$$

Функцией Бесселя 2-го рода (функцией Вебера $Y_\nu(z)$ или Неймана $N_\nu(z)$) является функция $N_\nu(z)$ ($Y_\nu(z)$), такая, где

$$N_\nu(z)=[J_\nu(z)\cos \pi z - J_{-\nu}(z)]/\sin \pi\nu. \qquad (3)$$

Функции Бесселя 3-го и 4-го родов (функции Ханкеля) есть соответственно функции $H_\nu^{(1)}(z)$ и $H_\nu^{(2)}(z)$, такие, что

$$\begin{aligned} H_\nu^{(1)}(z)&=J_\nu(z)+iN_\nu(z),\\ H_\nu^{(2)}(z)&=J_\nu(z)-iN_\nu(z). \end{aligned} \qquad (4)$$

Каждая из этих функций является аналитической функцией z во всей плоскости, разрезанной вдоль отрицательной части действительной оси. Когда $\nu=\pm n$ $J_\nu(z)$ не имеет особых точек и является целой функцией z. Функции Бесселя всех родов удовлетворяют рекуррентному соотношению

$$B_{\nu+1}(z)=\frac{2\nu}{z}B_\nu(z)-B_{\nu-1}(z). \qquad (5)$$

В первой программе значение бесселевой функции $J_n(x)$ (n — неотрицательное целое, x — вещественное) вычисляется по разложению в ряд (2). Надежность вычислений тем лучше, чем меньше значение x.

В программе используются переменные: N, J, $X1$ — порядок, значение и аргумент $J_n(x)$; $E1$ — параметр, определяющий точность (абсолютная точность меньше чем $E1\,(x/2)^n/n!$); X — рабочая переменная ($x/2$ или $x^2/4$); T — значение i-го слагаемого, равного $(-1)^i(x/2)^{2i}/i!\,(n+1)\ldots(n+i)$; S — рабочая переменная.

Структура программы:

10—30 — ввод n, x;

100 — особый случай $J_0(0)=1$;

110—130 — вычисление множителя $(x/2)^n/n!$;

140—170 — вычисление $J_n(x)$;

1000—1010 — вывод значения $J_n(x)$.

В контрольных примерах найдены значения J_0 (5) и J_2 (1.4). Их значения по таблицам [18] соответственно равны: -0.177596771314338 и 0.2073558995.

```
JLIST
  10 PRINT "ФУНКЦИЯ БЕССЕЛЯ JN (X)"
  20 INPUT "N="; N: INPUT "X="; X1
  30 E1=1E-8
 100 J=1:IF X1=0 AND N=0 THEN 1000
 110 X=.5*X1
 120 IF N=0 THEN 140
 130 FOR I=1 TO N:J=J*X/I:NEXT
 140 X=X*X:T=1:I=1:S=1
 150 T=-T*X/(I*(I+N)):S=S+T
 160 IF  ABS (T)>E1  THEN I=I+1:GOTO 150
```

```
 170 J=J*S
1000 PRINT "J"N"(" X1") ="J
1010 END

JRUN
ФУНКЦИЯ БЕССЕЛЯ JN (X)
N=0
X=5
J0 (5) = -.177596771
JRUN
ФУНКЦИЯ БЕССЕЛЯ JN (X)
N=2
X=1.4
J2(1.4) =.2073559
```

В следующей программе вычисляется значение $J_0(x)$ на основании аппроксимации J_0 многочленами [18]. При $|x| \leqslant 3$

$$J_0(x) = 1 + a_1 (x/3)^2 + \ldots + a_6 (x/3)^{12} + \varepsilon(x), \qquad (6)$$

где $a_1 = -2.2499997$, $a_2 = 1.2656208$, $a_3 = -0.3163866$, $a_4 = 0.0444479$, $a_5 = -0.0039444$, $a_6 = 0.0002100$, $|\varepsilon(x)| < 5 \cdot 10^{-8}$; при $3 < x < \infty$

$$J_0(x) = \frac{1}{\sqrt{x}} \cos\theta_0 [a_0 + a_1 (3/x) + \ldots + a_6 (3/x)^6 + \varepsilon_1(x)],$$
$$\theta_0 = x + b_0 + b_1 (3/x) + \ldots + b_6 (3/x)^6 + \varepsilon_2(x), \qquad (7)$$

где
$a_0 = 0.79788456$, $a_1 = -7.7 \cdot 10^{-7}$, $a_2 = -0.00552740$,
$a_3 = -0.00009512$, $a_4 = 0.00137237$, $a_5 = -0.00072805$,
$a_6 = 0.00014476$, $|\varepsilon_1(x)| < 1.6 \cdot 10^{-8}$,
$b_0 = -0.78539816$, $b_1 = -0.04166397$, $b_2 = -0.00003954$,
$b_3 = 0.00262573$, $b_4 = -0.00054125$, $b_5 = -0.00029333$,
$b_6 = 0.00013558$, $|\varepsilon_2(x)| < 7 \cdot 10^{-8}$.

В программе используются переменные: *X1*, *J* — аргумент и значение $J_0(x)$; *X* — рабочая переменная ($3/x$ или $x^2/9$).

Структура программы:

10—20 — ввод аргумента x;

110—130 — вычисление по формуле (6) при $|x| \leqslant 3$;

140—160 — вычисление по формуле (7) при $x > 3$;

1000—1010 — вывод значения $J_0(x)$.

В контрольных примерах вычислены значения $J_0(15)$ и $J_0(2.9)$. Их значения по таблицам [18] соответственно равны -0.014224472826781 и -0.224311545791968, что с точностью до $5 \cdot 10^{-8}$ совпадает с вычислениями.

```
JLIST
  10 PRINT "БЕССЕЛЬ J0 (X)"
  20 INPUT "X="; X1
 100 X=X1:IF X>3 THEN 140
 110 X=X*X/9
```

```
 120 J=(((((2.1E-4*X-.0039444)*X+.0444479)*X-.3163866)*X
     +1.2656208)*X-2.2499997)*X+1
 130 GOTO 1000
 140 X=3/X
 150 J=(((((1.3558E-4*X-2.9333E-4)*X-5.4125E-4)*X+2.62573E
     -3)*X-3.954E-5)*X-.04166397)*X-.78539816+X1
 160 J=COS (J)/SQR (X1)*((((((1.4476E-4*X-7.2805E-4)*X
     +1.37237E-3)*X-9.512E-5)*X-5.5274E-3)*X-7.7E-7)*X
     +.79788456)
1000 PRINT "J0 (" X1")="J
1010 END

JRUN
БЕССЕЛЬ J0 (X)
X=15
J0 (15)= -.0142244704
JRUN
БЕССЕЛЬ J0 (X)
X=2.9
J0 (2.9)= -.224311595
```

В следующей программе производится вычисление значения $Y_0(x)$ на основании аппроксимации J_0 полиномом [18]:

а) при $0<x\leqslant 3$

$$Y_0(x)=\frac{2}{\pi}J_0(x)\ln(x/2)+a_0+a_1(x/3)^2+\ldots+a_6(x/3)^{12}+\varepsilon_1(x), \quad (8)$$

где $a_0=0.36746691$, $a_1=0.60559366$, $a_2=-0.74350384$, $a_3=0.25300117$, $a_4=-0.04261214$, $a_5=0.00427916$, $a_6=-0.00024846$, $|\varepsilon_1(x)|<1.4\cdot 10^{-8}$;

б) при $x>3$

$$Y_0(x)=\frac{1}{\sqrt{x}}\sin\theta_0\,[a_0+a_1(3/x)+\ldots+a_6(3/x)^6+\varepsilon_2(x)], \quad (9)$$

где коэффициенты a_i, θ_0 и $\varepsilon_2(x)$ определены так же, как в (7).

Структура программы и обозначения такие же, как и в программе для $J_0(x)$ с заменой J на Y.

В контрольных примерах найдены значения $Y_0(2.9)$ и $Y_0(10)$, совпадающие с табличными до семи значащих цифр.

```
JLIST
  10 PRINT "БЕССЕЛЬ Y0 (X)"
  20 INPUT "X="; X1
 100 X=X1:IF X>3 THEN 140
 110 X=X*X/9:Y=(((((2.1E-4*X-.0039444)*X+.0444479)*X-.3163866)
     *X+1.2656208)*X-2.2499997)*X+1
 120 Y=.636619772*Y*LOG (X1/2)+(((((-.00024846*X+.00427916)*X
     -.04261214)*X+.25300117)*X-.74350384)*X+.60559366)*X
     +.36746691
 130 GOTO 1000
 140 X=3/X
```

```
150 Y=(((((1.3558E-4*X-2.9333E-4)*X-5.4125E-4)*X+2.62573E
    -3)*X-3.954E-5)*X-.04166397)*X-.78539816+X1
160 Y=SIN (Y) / SQR (X1)*((((((1.4476E-4*X-7.2805E-4)*X
    +1.37237E-3)*X-9.512E-5)*X-5.5274E-3)*X-7.7E-7)*X
    +.79788456)
1000 PRINT "Y0 (" X1")="Y
1010 END

JRUN
БЕССЕЛЬ Y0 (X)
X=2.9
Y0 (2.9)=.407911746
RUN
БЕССЕЛЬ Y0 (X)
X=10
Y0 (10)=.0556711675
```

В следующей программе производится вычисление значения на основании аппроксимации J_0 полиномом [18]:

а) при $|x| \leqslant 3$

$$x^{-1}J_1(x)=0.5+a_1(x/3)^2+\ldots+a_6(x/3)^{12}+\varepsilon(x), \qquad (10)$$

где $a_1=-0.56249985$, $a_2=0.21093573$, $a_3=-0.03954289$, $a_4=0.00443319$, $a_5=-0.00031761$, $a_6=0.00001109$, $|\varepsilon(x)|<1.3\times10^{-8}$;

б) при $x>3$

$$\sqrt{x}\,J_1(x)=\cos\theta_0\,[a_0+a_1(3/x)+\ldots+a_6(3/x)^6+\varepsilon_1(x)],$$
$$\theta_0=x+b_0+b_1(3/x)+\ldots+b_6(3/x)^6+\varepsilon_2(x), \qquad (11)$$

где

$a_0=0.79788456$, $a_1=0.00000156$, $a_2=0.01659667$,
$a_3=0.00017105$, $a_4=-0.00249511$, $a_5=0.00113653$,
$a_6=-0.00020033$, $|\varepsilon_1(x)|<4\cdot10^{-8}$;
$b_0=-2.35619449$, $b_1=0.12499612$, $b_2=0.00005650$,
$b_3=-0.00637879$, $b_4=0.00074348$, $b_5=0.00079824$,
$b_6=-0.00029166$, $|\varepsilon_2(x)|<9\cdot10^{-8}$.

Обозначения переменных и структура программы такие же, что и в программе для $J_0(x)$.

В контрольных примерах найдены значения $J_1(2.9)$ и $J_1(4)$.

```
JLIST
  10 PRINT "БЕССЕЛЬ J1 (X)"
  20 INPUT "X="; X1
 100 X=X1:IF X>3 THEN 140
 110 X=X*X/9
 120 J=X1*((((((.00001109*X-.00031761)*X+.00443319)*X-.03954289)
     *X+.21093573)*X-.56249985)*X+.5)
 130 GOTO 1000
 140 X=3/X
 150 J=(((((-.00029166*X+.00079824)*X+.00074348)*X-.00637879)*X
     +.0000565)*X+.12499612)*X-2.35619449+X1
```

```
 160 J=COS (J) / SQR (X1) * ((((((-.00020033 * X+.00113653) * X
     -.00249511) * X+.00017105) * X+.01659667) * X+.00000156) * X
     +.79788456)
1000 PRINT "J1(" X1")="J
1010 END

JRUN
БЕССЕЛЬ J1 (X)
X=2.9
J1 (2.9) =.375427516
JRUN
БЕССЕЛЬ J1 (X)
X=4
J1 (4) = -.06604332
```

В следующей программе производится вычисление значения $Y_1(x)$ на основании аппроксимации J_1 полиномом [18]:

а) при $0<x\leqslant 3$

$$xY_1(x)=\frac{2x}{\pi}J_1(x)\ln(x/2)+a_0+a_1(x/3)^2+\ldots+a_6(x/3)^{12}+\varepsilon(x), \quad (12)$$

где $a_0=-0.6366198$, $a_1=0.2212091$, $a_2=2.1682709$, $a_3=-1.3164827$, $a_4=0.3123951$, $a_5=-0.0400976$, $a_6=0.0027873$, $|\varepsilon(x)|<1.1\cdot 10^{-7}$;

б) при $x>3$

$$\sqrt{x}Y_1(x)=\sin\theta_0\,[a_0+a_1(3/x)+\ldots+a_6(3/x)^6+\varepsilon_1(x)], \quad (13)$$

где коэффициенты a_i, θ_0 и $\varepsilon_1(x)$ определены так же, как в (11).

Обозначения переменных и структура программы такие же, что и в программе для $Y_0(x)$.

В контрольных примерах найдены значения $Y_1(2.9)$ и $Y_1(10)$. Они совпадают с табличными с точностью до семи цифр.

```
JLIST
  10 PRINT "БЕССЕЛЬ Y1 (X)"
  20 INPUT "X="; X1
 100 X=X1:IF X>3 THEN 140
 110 X=X * X / 9:Y=X1 * ((((((.00001109 * X-.00031761) * X+.00443319)
     * X-.03954289) * X+.21093573) * X-.56249985) * X+.5)

 120 Y=.636619772 * LOG (X1 / 2) * Y+((((((.0027873 * X-.0400976) * X
     +.3123951) * X-1.3164827) * X+2.1682709) * X+.2212091) * X-.6366198)
     /X1
 130 GOTO 1000
 140 X=3/X
 150 Y=(((((-.00029166 * X+.00079824) * X+.00074348) * X-.00637879)
     * X+.0000565) * X+.12499612) * X-2.35619449+X1
 160 Y=SIN (Y) / SQR (X1) * ((((((-.00020033 * X+.00113653) * X
     -.00249511) * X+.00017105) * X+.01659667) * X+.00000156) * X
     +.79788456)
```

```
1000 PRINT "Y1(" X1")="Y
1010 END

JRUN
БЕССЕЛЬ Y1 (X)
X=2.9
Y1 (2.9)=.295940029
JRUN
БЕССЕЛЬ Y1 (X)
X=10
Y1 (10)=.249015423
```

В следующей программе вычисление значения бесселевой функции $J_n(x)$ производится на основании рекуррентного соотношения (5). Причем вначале вычисляется значение $J_{|n|}(|x|)$, а затем используются соотношения

$$J_{-n}(x)=(-1)^n J_n(x),\ J_n(-x)=(-1)^n J_n(x). \qquad (14)$$

Значения функций $J_0(x)$ и $J_1(|x|)$ вычисляются на основании разложения в ряд (2) при $|x|\leqslant 3$ и на основании аппроксимации многочленами (7), (11) при $|x|>3$.

В программе используются переменные: N, $X1$ — степень и аргумент $J_n(x)$; J, $J1$ — значение $J_0(x)$, $J_1(|x|)$, затем $J_{k-1}(|x|)$ и $J_k(|x|)$ соответственно; $N1=|n|$; $E1$ — точность вычислений при $|x|\leqslant 3$; X — рабочая переменная (равная $x^2/4$, $3/|x|$ или $|x|$); I — переменная цикла в (2); T — значение i-го слагаемого в (2).

Структура программы:

10—30 — ввод n, x, задание ε_1;

100 — исходные значения J, $N1$, значение $J_0(0)=1$;

120—150 — вычисление $J_0(x)$ и $J_1(|x|)$ при $|x|\leqslant 3$;

160—190 — то же для $|x|>3$;

200 — значение $J_0(x)$;

210—250 — вычисление $J_{|n|}(|x|)$ на основании рекуррентного соотношения (5);

260 — вычисление $J_n(x)$ по формуле (14);

1000—1010 — вывод значения $J_n(x)$.

В контрольном примере найдены значения $J_2(7)$, $J_{-11}(-10)$. Они совпадают с табличными с точностью до восьми значащих цифр. Программа дает правильные результаты для больших значений $|n|$, если $|x|\sim|n|$. Если $|n|\gg|x|$, то вычисления дают большую погрешность.

```
JLIST
  10 PRINT "ФУНКЦИЯ БЕССЕЛЯ JN (X)"
  20 INPUT "N="; N: INPUT "X="; X1
  30 E1=1E-8
 100 J=1:N1=ABS (N): IF X1=0 AND N=0 THEN 1000
 110 IF ABS (X1)>3 THEN X=3 / ABS (X1): GOTO 160
 120 J1=1:T=1:X=.25 * X1 * X1:I=1
 130 T=-T * X / I / I:J=J+T: J1=J1+T / (I+1)
```

```
140 IF ABS (T)>E1 THEN I=I+1: GOTO 130
150 J1=J1*ABS (X1) / 2: GOTO 200
160 J=(((((1.3558E-4 * X-2.9333E-4) * X-5.4125E-4) * X+2.62573E
    -3) * X-3.954E-5) * X-.04166397) * X-.78539816+ABS (X1)
170 J=COS (J) / SQR (ABS (X1)) * ((((((1.4476E-4 * X-7.2805E-4)
  * X+1.37237E-3) * X-9.512E-5) * X-5.5274E-3) * X-7.7E-7) * X
    +.79788456)
180 J1=(((((-.00029166 * X+.00079824) * X+.00074348) * X-.00637879)
    * X+.0000565) * X+.12499612) * X-2.35619449+ABS (X1)
190 J1=COS (J1) / SQR (ABS (X1)) * ((((((-.0002033 * X+.00113653) * X
    -.00249511) * X+.00017105) * X+.01659667) * X+.00000156) * X
    +.79788456)
200 IF N=0 THEN 1000
210 IF X1=0 THEN J=0: GOTO 1000
220 X=ABS (X1): IF N1=1 THEN 260
230 FOR I=1 TO N1-1
240 T=J1:J1=2 * I * T / X-J:J=T
250 NEXT
260 J=J1 * (SGN (X1)^N): IF N<0 THEN J=J * (-1)^N
1000 PRINT "J["N"] (" X1") ="J
1010 END

JRUN
ФУНКЦИЯ БЕССЕЛЯ JN (X)
N=2
X=7
J [2] (7) = -.30141722
JRUN
ФУНКЦИЯ БЕССЕЛЯ JN (X)
N=-11
X=-10
J [-11] (-10) =.123116532
```

В следующей программе вычисляются действительная и мнимая части **функции Ханкеля** $H_n^{(1)}(x)$ от действительного аргумента x и целого порядка n.

Вычисления проводятся на основании разложения функции Бесселя J_n и Неймана N_n в ряд (2) и ряд

$$N_n(x) = -\frac{1}{\pi}(x/2)^{-n}\sum_{k=0}^{n-1}\frac{(n-k-1)!}{k!}\left(\frac{x^2}{4}\right)^k + \frac{2}{\pi}J_n(x)\ln(x/2) -$$

$$-\frac{1}{\pi}\left(\frac{x}{2}\right)^n\sum_{k=0}^{\infty}[\psi(k+1)+\psi(k+n+1)]\frac{(-x^2/4)^k}{k!\,(n+k)!}, \qquad (14)$$

где $\psi(1) = -C$, $\psi(n) = -C + \sum_{k=1}^{n-1} 1/k$, $C = 0.57721566490153$ — постоянная Эйлера.

Ряды (2) и (14) в программе вычисляются одновременно. Используются константы $2C = 1.1544\ldots$ и $1/\pi = 0.318\ldots$. Програм-

ма получена переводом на язык Бейсик Алгол-программы 124б [6].

В программе используются переменные: N, X — порядок и аргумент $H_n^{(1)}(x)$; $H1$ и $H2$ — действительная и мнимая части $H_n^{(1)}(x)=J_n(x)+iN_n(x)$; Q — переменная цикла; P, R, S, T, D, B, A — рабочие переменные; $K=x^2/4$.

Структура программы:

10—20 — ввод n и x;

100—230 — вычисления рядов (2) и (14);

1000—1010 — вывод значений $J_n(x)$ и $N_n(x)$.

В контрольном примере вычислены значения $H_1^{(1)}(8)$ и $H_7^{(1)}(4)$, совпадающие с табличными до 6—8 значащих цифр.

```
JLIST
  10 PRINT "ФУНКЦИИ ХАНКЕЛЯ JN (X)+I*NN (X)"
  20 INPUT "N=";N: INPUT "X=";X
 100 IF X=0 THEN PRINT "ФУНКЦИЯ NN ("X") НЕ ОПРЕДЕЛЕНА": END
 110 A=1:R=1: H1=0: H2=0:S=0
 120 IF N=0 THEN 140
 130 FOR Q=1 TO N:R=R*Q:S=S+1/Q: NEXT
 140 D=0: IF N<>0 THEN D=R/N
 150 R=1/R:K=X*X/4:P=K^N:T=LOG (K)+1.1544313298031
 160 Q=0
 170 IF Q>N AND B=H2 THEN 230
 180 B=H2:H1=H1+A*R: H2=H2+A*R*(T-S)
 190 IF Q<N THEN H2=H2-A*D/P
 200 A=A*K/(Q+1):R=-R/(Q+N+1):S=S+1/(Q+1)+1/(Q+N+1)
 210 IF Q+1<N THEN D=D/(N-Q-1)
 220 Q=Q+1: GOTO 170
 230 P=(X/2)^N: H1=H1*P: H2=.318309886184*H2*P
1000 PRINT "H"N" ("X")=" H1"+I*(" H2")"
1010 END
JRUN
ФУНКЦИИ ХАНКЕЛЯ JN (X)+I*NN (X)
N=1
X=8
H1 (8) =.234636339+I*(-.158060465)
JRUN
ФУНКЦИИ ХАНКЕЛЯ JN (X)+I*NN (X)
N=7
X=4
H7 (4) =.0151760694+I*(-3.70622394)
```

7.6. Модифицированные функции Бесселя

Если аргумент в функциях Бесселя повернуть на 90°, получим соответственно модифицированные функции Бесселя, функцию Бесселя мнимого аргумента $I_\nu(z)$ и модифицированную функ-

цию Ханкеля $K_\nu(z)$ [3], [18], [19]. При повороте на 135° получаем функции Кельвина $\mathrm{ber}_\nu(z)$, $\mathrm{bei}_\nu(z)$, $\mathrm{ker}_\nu(z)$, $\mathrm{kei}_\nu(z)$.

Функции $I_\nu(z)=e^{-i\pi\nu/2}J_\nu(ze^{i\pi/2})$, $K_\nu(z)=i\pi/2e^{i\pi\nu/2}$ и $H_\nu^{(1)}(ze^{i\pi/2})$ являются решениями дифференциального уравнения

$$z^2w''+zw'-(z^2+\nu^2)w=0 \tag{1}$$

и разлагаются в ряды

$$I_n(z)=\left(\frac{z}{2}\right)^n\sum_{k=0}^{\infty}\frac{(z^2/4)^k}{k!\,\Gamma(n+k+1)}, \tag{2}$$

$$K_n(z)=\frac{1}{2}\left(\frac{z}{2}\right)^{-n}\sum_{k=0}^{n-1}\frac{(n-k-1)!}{k!}\left(-\frac{z^2}{4}\right)^k+(-1)^{n+1}I_n(z)\ln\frac{z}{2}+$$

$$+\frac{1}{2}\left(-\frac{z}{2}\right)^n\sum_{k=0}^{\infty}[\psi(k+1)+\psi(k+n+1)]\frac{(z^2/4)^k}{k!\,(n+k)!}, \tag{3}$$

где $\psi(1)=-C$, $\psi(n)=-C+1+\frac{1}{2}+\ldots+\frac{1}{n-1}$, C — постоянная Эйлера.

Функции $I_0(x)$ и $I_1(x)$ могут быть аппроксимированы многочленами [18]:

а) при $|x|\leqslant 3.75$

$$I_0=1+3.5156229t^2+3.0899424t^4+1.2067492t^6+0.2659732t^8+$$
$$+0.0360768t^{10}+0.0045813t^{12}+\varepsilon_1(x), \tag{4}$$

$$x^{-1}I_1(x)=0.5+0.87890594t^2+0.51498869t^4+0.15084934t^6+$$
$$+0.02658733t^8+0.00301532t^{10}+0.00032411t^{12}+\varepsilon_2(x), \tag{5}$$

где $|\varepsilon_1(x)|<1.6\cdot10^{-7}$, $|\varepsilon_2(x)|<8\cdot10^{-9}$, $t=x/3.75$;

б) при $x\geqslant 3.75$

$$\sqrt{x}e^{-x}I_0(x)=0.39894228+0.01328592t^{-1}+0.00225319t^{-2}-$$
$$-0.00157565t^{-3}+0.00916281t^{-4}-0.02057706t^{-5}+$$
$$+0.02635537t^{-6}-0.01647633t^{-7}+0.00392377t^{-8}+\varepsilon_3(x), \tag{6}$$

$$\sqrt{x}e^{-x}I_1(x)=0.39894228-0.03988024t^{-1}-0.00362018t^{-2}+$$
$$+0.00163801t^{-3}-0.01031555t^{-4}+0.02282967t^{-5}-$$
$$-0.02895312t^{-6}+0.01787654t^{-7}-0.00420059t^{-8}+\varepsilon_4(x), \tag{7}$$

где $|\varepsilon_3(x)|<1.9\cdot10^{-7}$, $|\varepsilon_4(x)|<2.2\cdot10^{-7}$, $t=x/3.75$.

Функция $e^xK_p(x)$ имеет интегральное представление [18]:

$$e^xK_p(x)=\int_0^{\infty}e^{x(1-\mathrm{ch}\,t)}ch\,(pt)\,dt. \tag{8}$$

Модифицированные функции Бесселя $\mathrm{ber}_\nu(x)$, $\mathrm{bei}_\nu(x)$ и $\mathrm{ker}_\nu(x)$, $\mathrm{kei}_\nu(x)$ определяются из соотношений

$$\mathrm{ber}_\nu(x)+i\,\mathrm{bei}_\nu(x)=J_\nu(xe^{3i\pi/4}),$$
$$\mathrm{ker}_\nu(x)+i\,\mathrm{ker}_\nu(x)=\frac{i\pi}{2}H_\nu^{(1)}(xe^{3i\pi/4}). \tag{9}$$

В наиболее важном случае $\nu=0$ функции $\mathrm{ber}(x)$, $\mathrm{bei}(x)$, $\mathrm{ker}(x)$, $\mathrm{kei}(x)$ (индекс ν в обозначениях функций опускается) разлагаются в ряды [18].

$$\operatorname{ber}(x)=1-\frac{(x^2/4)^2}{2!}+\frac{(x^2/4)^4}{(4!)^2}-\ldots,$$

$$\operatorname{bei}(x)=\frac{x^2}{4}\left[1-\frac{(x^2/4)^2}{(3!)^2}+\frac{(x^2/4)^4}{(5!)^2}-\ldots\right], \qquad (10)$$

$$\ker(x)=-\operatorname{ber}(x)\ln\frac{x}{2}+\frac{\pi}{4}\operatorname{bei}(x)+\sum_{k=0}^{\infty}(-1)^k\frac{\psi(2k+1)}{((2k)!)^2}\left(\frac{x^2}{4}\right)^{2k}=$$

$$=\sum_{k=0}^{\infty}(-1)^k\frac{(x^2/4)^{2k}}{((2k)!)^2}\left[-\ln\frac{x}{2}+\frac{\pi}{4}\frac{x^2/4}{(2k+1)^2}+\psi(2k+1)\right], \qquad (11)$$

$$\operatorname{kei}(x)=-\operatorname{bei}(x)\ln\frac{x}{2}-\frac{\pi}{4}\operatorname{ber}(x)+\sum_{k=0}^{\infty}(-1)^k\frac{\psi(2k+2)}{((2k+1)!)^2}\left(\frac{x^2}{4}\right)^{2k+1}=$$

$$=\sum_{k=0}^{\infty}(-1)^k\frac{(x^2/4)^{2k}}{((2k)!)^2}\left[-\frac{\pi}{4}+\frac{x^2}{4}\frac{\psi(2k+2)-\ln\frac{x}{2}}{(2k+1)^2}\right]. \qquad (12)$$

Если аргумент x или индекс ν отрицательны, то справедливы следующие соотношения:

$$\operatorname{ber}_n(-x)=(-1)^n\operatorname{ber}_n(x),\quad \operatorname{ber}_{-n}(x)=(-1)^n\operatorname{ber}_n(x). \qquad (13)$$

Аналогичные формулы имеют место и для функций $\operatorname{bei}_n(x)$, $\ker_n(x)$ и $\operatorname{kei}_n(x)$.

Для произвольного значения n имеет место разложение в степенной ряд [18]:

$$\operatorname{ber}_n(x)=\left(\frac{x}{2}\right)^n\sum_{k=0}^{\infty}\frac{\cos\left(\frac{3\pi n}{4}+\frac{k\pi}{2}\right)}{k!\,(n+k)!}\left(\frac{x^2}{4}\right)^k=$$

$$=\left(\frac{x}{2}\right)^n\left[\frac{\cos\frac{3\pi n}{4}}{0!n!}-\frac{\sin\frac{3\pi n}{4}}{1!\,(n+1)!}\frac{x^2}{4}-\frac{\cos\frac{3\pi n}{4}}{2!\,(n+2)!}\left(\frac{x^2}{4}\right)^2+\right.$$

$$\left.+\frac{\sin\frac{3\pi n}{4}}{3!\,(n+3)!}\left(\frac{x^2}{4}\right)^3-\ldots\right], \qquad (14)$$

$$\operatorname{bei}(x)=\left(\frac{x}{2}\right)^n\sum_{k=0}^{\infty}\frac{\sin\left(\frac{3\pi n}{4}+\frac{k\pi}{2}\right)}{k!\,(n+k)!}\left(\frac{x^2}{4}\right)^k=$$

$$=\left(\frac{x}{2}\right)^n\left[\frac{\sin\frac{3\pi n}{4}}{0!n!}+\frac{\cos\frac{3\pi n}{4}}{1!\,(n+1)!}\frac{x^2}{4}-\frac{\sin\frac{3\pi n}{4}}{2!\,(n+2)!}\left(\frac{x^2}{4}\right)^2-\right.$$

$$\left.-\frac{\cos\frac{3\pi n}{4}}{3!\,(n+3)!}\left(\frac{x^2}{4}\right)^3+\ldots\right]. \qquad (15)$$

В первой программе вычисляется значение модифицированной функции Бесселя $I_n(x)$ на основании разложения в ряд (2).

Вычисления продолжаются до тех пор, пока значения сумм m и $m+1$ слагаемых не совпадут: $I_m = I_{m+1}$ ($I_m = \sum_{k=0}^{m} (x^2/4)^k / k!\,(n+k)!$).

В программе используются переменные: N, $X1$ — степень, аргумент и значение $I_n(x)$; K — переменная цикла; X — рабочая переменная ($x/2$ или $x^2/4$); T — значение k-го слагаемого в (2).

Структура программы:

10—20 — ввод n, x;

100—110 — $I_0(0)=1$, начальные значения переменных x, I;

120 — вычисление первого слагаемого в (2);

130—150 — вычисление по формуле (2), проверка условия $I_m = I_{m+1}$.

1000—1010 — вывод значения $I_n(x)$.

В контрольном примере найдены значения $I_3(2)$ и $I_4(5)$. Их значения по таблицам [19]: $I_3(2)=0.2127$ и $I_4(5)=5.1082$. Надежность вычислений тем лучше, чем меньше значение x.

```
JLIST
  10 PRINT "МОДИФ. ФУНКЦИЯ БЕССЕЛЯ I N (X)"
  20 INPUT "N="; N: INPUT "X="; X1
 100 I=1: IF N=0 AND X1=0 THEN 1000
 110 X=X1*X1/4
 120 FOR K=1 TO N:I=I/K: NEXT : K=1:T=I
 130 T=T*X/K/(K+N):K=K+1
 140 IF I<>I+T THEN I=I+T:GOTO 130
 150 I=I*(X1/2)^N
1000 PRINT "I "N"(" X1") ="I
1010 END

JRUN
МОДИФ. ФУНКЦИЯ БЕССЕЛЯ I N (X)
N=3
X=2
I 3 (2) =.212739959

JRUN
МОДИФ. ФУНКЦИЯ БЕССЕЛЯ I N (X)
N=4
X=5
I 4 (5) =5.10823477
```

В следующей программе вычисляется значение $I_0(x)$ на основании аппроксимации полиномами (5), (7).

В программе используются переменные: $X1$, I — аргумент и значение $I_0(x)$; X — рабочая переменная ($(x/3.75)^2$ или $3.75/x$).

Структура программы:

10—20 — ввод x;

100—130 — вычисление $I_0(x)$ при $|x| \leqslant 3.75$;

140—150 — вычисление $I_0(x)$ при $x > 3.75$;

1000—1010 — вывод $I_0(x)$.

В контрольном примере вычислены значения $I_0(2.9)$ и $I_0(10)$. Их значения по таблицам [18] соответственно равны 4.50274866 и 2815.71663.

```
JLIST
  10 PRINT "МОДИФИЦИРОВАННАЯ ФУНКЦИЯ БЕССЕЛЯ I0 (X)"
  20 INPUT "X=": X1
 100 X=X1:IF X>3.75 THEN 140
 110 X=X*X/14.0625
 120 I=1+X * (3.5156229+X * (3.0899424+X * (1.2067492+X * (.2659732
 +X* (.0360768+X* .0045813)))))
 130 GOTO 1000
 140 X=3.75/X
 150 I=(((((((.00392377 * X-.01647633) * X+.02635537) * X-.02057706)
     * X+.00916281) * X-.00157565) * X+.00225319) * X+.01328592) * X
     +.39894228) * EXP (X1)/SQR (X1)
1000 PRINT "I0(" X1")="I
1010 END

JRUN
МОДИФИЦИРОВАННАЯ ФУНКЦИЯ БЕССЕЛЯ I0 (X)
X=2.9
I0 (2.9) =4.50274868

JRUN
МОДИФИЦИРОВАННАЯ ФУНКЦИЯ БЕССЕЛЯ I0 (X)
X=10
I0 (10) =2815.71666
```

В следующей программе значение модифицированной функции Бесселя $I_1(x)$ вычисляется на основании аппроксимации полиномом (6), (8). Структура и обозначения такие же, как в предыдущей программе. В контрольном примере вычислены значения $I_1(2.9)$ и $I_1(10)$.

Их значения по таблицам [18] соответственно равны 3.61260721 и 2670.98830

```
JLIST
  10 PRINT "МОДИФИЦИРОВАННАЯ ФУНКЦИЯ БЕССЕЛЯ I1 (X)"
  20 INPUT "X=": X1
 100 X=X1: IF X>3.75 THEN 140
 110 X=X*X/14.0625
 120 I=(.5+X * (.87890594+X * (.51498869+X * (.15084934+X * (.02658733
     +X * (.00301532+X *.00032411))))))  * X1
 130 GOTO 1000
 140 X=3.75/X
 150 I=(((((((-.00420059 * X+.01787654) * X-.02895312) * X+.02282967)
```

```
    * X-.01031555) * X+.00163801) * X-.00362018) * X-.03988024) * X
    +.39894228) * EXP (X1)/SQR (X1)
1000 PRINT "I1(" X1")="I
1010 END

JRUN
МОДИФИЦИРОВАННАЯ ФУНКЦИЯ БЕССЕЛЯ I1 (X)
X=2.9
I1 (2.9)=3.61260722

JRUN
МОДИФИЦИРОВАННАЯ ФУНКЦИЯ БЕССЕЛЯ I1 (X)
X=10
I1 (10)=2670.98832
```

В следующей программе вычисление значения функции $e^x Kp\,(x)$ производится на основании интегрального представления (13).

В программе используются переменные: P, X, R — индекс, аргумент и значение $e^x Kp\,(x)$; $E1$ — относительная точность вычислений; Z, R, H, G, S, $Z1$, U — рабочие переменные.

Структура программы:

10—30 — ввод p, x, задание ε_1;

100—160 — вычисление $e^x Kp\,(x)$;

1000—1010 — вывод значения $e^x Kp\,(x)$.

Программа получена переводом на язык Бейсик (с заменой абсолютной точности вычислений на относительную) Алгол-программы 163б [7].

В контрольном примере вычислены значения $e^4 K_9\,(4)$ и $e^{17} K_1\,(17)$, совпадающие с табличными значениями [18].

```
JLIST
  10 PRINT "МОДИФИЦИРОВАННАЯ ФУНКЦИЯ ХАНКЕЛЯ EXP (X) *
     KP (X)"
  20 INPUT "P="; P: INPUT "X="; X
  30 E1=1E-8
 100 R=0:H=1
 110 G=R:S=0:Z=EXP (H/2):U=Z * Z
 120 Z1=Z^P:F=.5 * EXP (X * (1-.5 * (Z+1/Z))) * (Z1+1/Z1)
 130 S=S+F:Z=Z * U
 140 IF F>E1 THEN 120
 150 R=H * S:H=H/2
 160 IF ABS (R-G)>R * E1 THEN 110
1000 PRINT "EXP ("X") * K"P" ("X")="R
1010 END

JRUN
МОДИФИЦИРОВАННАЯ ФУНКЦИЯ ХАНКЕЛЯ EXP (X) * KP (X)
P=9
X=4
EXP (4) * K9 (4)=1325.16874
JRUN
```

```
МОДИФИЦИРОВАННАЯ ФУНКЦИЯ ХАНКЕЛЯ EXP (X) * KP (X)
P=1
X=17
EXP (17) * K1 (17) = .310561234
```

В следующих двух программах производится вычисление функций ber (x) и bei (x) по разложению в ряды (10) с оценкой точности. Вычисления продолжаются, пока $(x^2/4)^{2k}/((2k)!)^2 > \varepsilon_1$ (во второй программе, пока $(x^2/4)^{2k+1}/((2k+1)!)^2 > \varepsilon_1$).

В программе используются переменные: *X1*, *B* — аргумент и значение функций; *E1* — точность вычислений ε_1; *X* — рабочая переменная $(x^2/4)$; *T* — значение k-го слагаемого в (10); *I* — переменная цикла.

Структура программы:

10—30 — введение x, задание ε_1;

100—130 — вычисление по формуле (10);

1000—1010 — вывод значения функций.

В контрольных примерах найдены значения bei (3) и ber (3), совпадающие с табличными точнее, чем 10^{-8}.

```
JLIST
  10 PRINT "ФУНКЦИЯ КЕЛЬВИНА BEI (X)"
  20 INPUT "X="; X1
  30 E1=1E-8
 100 X=X1*X1/4:B=X:T=X:I=2:X=X*X
 110 T=-T*X/(I*I*(I+1)*(I+1))
 120 B=B+T:I=I+2
 130 IF ABS (T)>E1 THEN 110
1000 PRINT "BEI ("X1") ="B
1010 END

JRUN
ФУНКЦИЯ КЕЛЬВИНА BEI (X)
X=3
BEI (3) = 1.93758679

JLIST
  10 PRINT "ФУНКЦИЯ КЕЛЬВИНА BER (X)"
  20 INPUT "X="; X1
  30 E1=1E-8
 100 B=1:X=X1*X1/4:X=X*X:T=1:I=1
 110 T=-T*X/(I*I*(I+1)*(I+1))
 120 B=B+T:I=I+2
 130 IF ABS (T)>E1 THEN 110
1000 PRINT "BER (" X1") ="B
1010 END

JRUN
ФУНКЦИЯ КЕЛЬВИНА BER (X)
X=3
BER (3) = -.22138025
```

В следующих двух программах производится вычисление значений функций $\mathrm{ber}_n(x)$ и $\mathrm{bei}_n(x)$ по разложению в ряды (14) и (15). Вычисления продолжаются до тех пор, пока не совпадут значения сумм $2i$ и $2i+2$ — слагаемых.

В программах используются переменные: N, $X1$, B — индекс, аргумент и значение функций; X — рабочая переменная, равная $x^2/4$ или $(x^2/4)^2$; $N1$ — остаток от деления N на 8; $F1=\cos(3\pi n_1/4)$, $F2=\sin(3\pi n_1/4)\cdot x^2/4$ (во второй программе $F1=\sin(3\pi n_1/4)$, $F_2=\cos(3\pi n_1/4)\cdot x^2/4$), затем последовательные слагаемые в (14) и (15); I — переменная цикла; B, $B1$ — значения сумм $2i$ и $2i+2$ — слагаемых соответственно.

Структура программы:

10—20 — ввод n, x;

100—120 — начальные значения x, x_1, вычисление $\sin(3\pi n_1/4)$ $x^2/4$ и $\cos(3\pi n_1/4)$;

130—140 — вычисление $n!$ и первых двух слагаемых в (14), (15);

150—210 — вычисления по формулам (14), (15);

1000—1010 — вывод значения функции $\mathrm{ber}_n(x)$ ($\mathrm{bei}_n(x)$).

В контрольных примерах найдены значения $\mathrm{ber}_2(3)$, $\mathrm{ber}_0(3)$ и $\mathrm{bei}_2(3)$, совпадающие с табличными.

```
JLIST
  10 PRINT "ФУНКЦИЯ КЕЛЬВИНА BER N (X)"
  20 INPUT "N=";N: INPUT "X="; X1
 100 X=X1*X1/4: N1=N
 110 IF N1>8 THEN N1=N1-8: GOTO 110
 120 F1=COS (N1*2.35619449019):F2=X*SIN (N1*2.35619449019)
 130 N1=1: FOR I=1 TO N: N1=N1*I: NEXT
 140 X=X*X: F1=F1/N1: F2=F2/N1/(N+1)
 150 I=2:B=F1-F2: N1=1
 160 F1=-X*F1/(I*(I-1)*(N+I-1)*(N+I))
 170 F2=-X*F2/(I*(I+1)*(N+I+1)*(N+I))
 180 B1=B+F1-F2
 190 IF B1=B THEN 210
 200 B=B1:I=I+2: GOTO 160
 210 B=B*(X1/2)^N
1000 PRINT "BER"N"(" X1")="B
1010 END

JRUN
ФУНКЦИЯ КЕЛЬВИНА BER N (X)
N=2
X=3
BER2 (3) = .808368465
JRUN
ФУНКЦИЯ КЕЛЬВИНА BER N (X)
N=0
X=3
BER0 (3) = -.221380249
```

```
JLIST
  10 PRINT "ФУНКЦИЯ КЕЛЬВИНА BEI N (X)"
  20 INPUT "N=";  N: INPUT "X=";  X1
 100 X=X1 * X1/4: N1=N
 110 IF N1>8 THEN N1=N1-8: GOTO 110
 120 F1=SIN (N1 * 2.35619449019):F2=X * COS (N1 * 2.35619449019)
 130 N1=1: FOR I=1 TO N: N1=N1 * I: NEXT
 140 X=X * X: F1=F1/N1:F2=F2/N1/(N+1)
 150 I=2:B=F1+F2:N1=1
 160 F1=-X * F1/(I * (I-1) * (N+I-1) * (N+I))
 170 F2=-X * F2/(I * (I+1) * (N+I+1) * (N+I))
 180 B1=B+F1+F2
 190 IF B1=B THEN 210
 200 B=B1:I=I+2: GOTO 160
 210 B=B * (X1/2)^N
1000 PRINT "BEI"N"(" XI") ="B
1010 END

JRUN
ФУНКЦИЯ КЕЛЬВИНА BEI N (X)
N=2
X=3
BEI2 (3) = -.891022363
```

7.7. Сферические и модифицированные сферические функции Бесселя. Функции Эйри

Сферические функции Бесселя являются решениями дифференциального уравнения

$$x^2w''+2xw'+[x^2-n(n+1)]\,w=0 \quad (n=0,\ \pm 1,\ \pm 2,\ \ldots). \qquad (1)$$

Сферическая функция Бесселя 1-го рода есть функция Бесселя полуцелого индекса:

$$j_n(x)=\sqrt{\frac{\pi}{2x}}\,J_{n+1/2}(x),$$

2-го рода:

$$y_n(x)=\sqrt{\frac{\pi}{2x}}\,Y_{n+1/2}(x),$$

3-го рода:

$$h_n^{(1)}(x)=j_n(x)+iy_n(x),$$
$$h_n^{(2)}(x)=j_n(x)-iy_n(x).$$

Все сферические функции Бесселя удовлетворяют рекуррентному соотношению [18]

$$f_{n+1}(x)=-f_{n-1}(x)+\frac{2n+1}{2}f_n(x), \qquad (2)$$

причем $j_0(x)=\frac{\sin x}{x}$; $j_1(x)=\frac{\sin x}{x^2}-\frac{\cos x}{x}$, $j_{-1}(x)=-y_0(x)$,

$$y_0(x)=-\frac{\cos x}{x},\ y_1(x)=-\frac{\cos x}{x^2}-\frac{\sin x}{x}.$$

В первой программе производится вычисление значения сферической функции Бесселя 1-го рода $j_n(x)$ на основании рекуррентного соотношения (2) для произвольных целых значений $n \geqslant 0$.

В программе используются переменные: X, N, J — аргумент, индекс и значение $j_n(x)$; T, $J1$ — рабочие переменные, последовательные значения j_{k-1} и j_k; I — переменная цикла в (2).

Структура программы:

10—20 — ввод n, x;

100 — проверка условия $x \neq 0$;

110—120 — вычисление значений $J = j_0(x)$, $J1 = j_{-1}(x)$;

130 — вычисление по формулам (2);

1000—1010 — вывод значения $j_n(x)$.

В контрольных примерах показано, что $j_2(3) = -(2/9)\sin 3 - (1/3)\cos 3 = 0.298637497$, $j_3(1) = -14\cos 1 + 9\sin 1 = 9.00658 \cdot 10^{-3}$.

```
JLIST
  10 PRINT "СФЕР. ФУНКЦИЯ БЕССЕЛЯ 1-РОДА JN (X)"
  20 INPUT "N=";N: INPUT "X="; X
 100 IF X=0 THEN PRINT "J"N" (0) НЕ ОПРЕДЕЛЕНА": END
 110 J=SIN (X)/X:J1=COS (X)/X
 120 IF N=0 THEN 1000
 130 FOR I=1 TO N:Z=J:J=J * (2 * I-1)/X-J1:J1=Z:NEXT
1000 PRINT "J"N" ("X") ="J
1010 END

JRUN
СФЕР. ФУНКЦИЯ БЕССЕЛЯ 1-РОДА JN (X)
N=2
X=3
J2 (3) =.298637497
JRUN
СФЕР. ФУНКЦИЯ БЕССЕЛЯ 1-РОДА JN (X)
N=3
X=1
J3 (1) =9.00657941E-03
```

Во второй программе производится вычисление значений сферической функции Бесселя 2-го рода $y_n(x)$ (сферическая функция Неймана). Структура и обозначения в программе такие же, как и в предыдущей (с заменой I, $I1$ на Y, $Y1$).

В контрольном примере показано, что $y_2(3) = (2/9)\cos 3 - (1/3)\sin 3 = -0.26703833$, $y_3(3) = -0.50802305$.

```
JLIST
  10 PRINT "СФЕРИЧЕСКАЯ Ф-Я БЕССЕЛЯ 2-РОДА YN (X)"
  20 INPUT "N=";N: INPUT "X=";X
 100 IF X=0 THEN PRINT "Y"N" (0) НЕ ОПРЕДЕЛЕНА": END
 110 Y=-COS (X)/X:Y1=SIN (X)/X
```

```
 120 IF N=0 THEN 1000
 130 FOR I=1 TO N:Z=Y:Y=Y*(2*I-1)/X-Y1:Y1=Z: NEXT
1000 PRINT "Y"N" ("X")="Y
1010 END

JRUN
СФЕРИЧЕСКАЯ Ф-Я БЕССЕЛЯ 2-РОДА YN (X)
N=2
X=3
Y2 (3) = -.267038335
JRUN
СФЕРИЧЕСКАЯ Ф-Я БЕССЕЛЯ 2-РОДА YN (X)
N=3
X=3
Y3 (3) = -.508023056
```

В двух следующих программах производится вычисление значений $j_n(x)$ и $y_n(x)$ по их разложению в степенные ряды [18]:

$$j_n(x)=\frac{x^n}{(2n+1)!!}\left[1-\frac{x^2/2}{1!\,(2n+3)}+\frac{(x^2/2)^2}{2!\,(2n+3)(2n+5)}-\ldots\right],$$

$$y_n(x)=-\frac{(2n-1)!!}{x^{n+1}}\left[1-\frac{x^2/2}{1!\,(1-2n)}+\frac{(x^2/2)^2}{2!\,(1-2n)(3-2n)}-\ldots\right]. \qquad (3)$$

Программе используются переменные: N, $X1$, I (Y) — индекс, аргумент и значение $j_n(x)$ ($y_n(x)$); T — значение i-го слагаемого в (3); F — рабочая переменная, значение множителей перед скобками в (3); X — рабочая переменная $(x^2/2)$.

Структура программы:

10—20 — ввод n, x;

100 — начальные значения переменных;

110—120 — вычисление суммы k слагаемых в (3), суммирование продолжается, пока $S_k \neq S_{k+1}$;

130—140— вычисление множителя $x^n/(2n+1)!!$ или $-(2n-1)!!/x^{n+1}$;

150 — вычисление значения $j_n(x)$ ($y_n(x)$);

1000—1010 — вывод значения $j_n(x)/(y_n(x))$.

В контрольных примерах вычислены те же значения $j_n(x)$ и $y_n(x)$, что и в двух предыдущих программах.

```
JLIST
  10 PRINT "СФЕР. ФУНКЦИЯ БЕССЕЛЯ 1-РОДА JN (X)"
  20 INPUT "N="; N: INPUT "X="; X1
 100 T=1:F=0:J=I:X=X1*X1/2
 110 F=F+1:T=-T*X/F/ (N+N+F+F+1)
 120 IF J<>J+T THEN J=J+T: GOTO 110
 130 F=1: FOR T=1 TO N
 140 F=F*X1/ (T+T+1): NEXT
 150 J=J*F
1000 PRINT "J"N" ("X1")="J
1010 END
```

```
JRUN
СФЕР. ФУНКЦИЯ БЕССЕЛЯ 1-РОДА JN (X)
N=2
X=3
J2 (3) = .298637497
JRUN
СФЕР. ФУНКЦИЯ БЕССЕЛЯ 1-РОДА JN (X)
N=3
X=1
J3 (1) = 9.00658112E-03

JLIST
  10 PRINT "СФЕР. ФУНКЦИЯ БЕССЕЛЯ 1-РОДА YN (X)"
  20 INPUT "N="; N: INPUT "X="; X1
 100 T=1:F=0:Y=1:X=X1 * X1/2
 110 F=F+1:T=-T * X/F/ (F+F-N-N-1)
 120 IF Y<>Y+T THEN Y=Y+T: GOTO 110
 130 F=1: FOR T=1 TO N
 140 F=F/X1 * (T+T-1): NEXT: F=-F/X1
 150 Y=Y * F
1000 PRINT "Y"N" ("X1") ="Y
1010 END

JRUN
СФЕР. ФУНКЦИЯ БЕССЕЛЯ 1-РОДА YN (X)
N=2
X=3
Y2 (3) = -.267038335
JRUN
СФЕР. ФУНКЦИЯ БЕССЕЛЯ 1-РОДА YN (X)
N=3
X=3
Y3 (3) = -.508023056
```

Модифицированные сферические функции Бесселя есть функции, которые являются решениями дифференциального уравнения

$$z^2w''+2zw'-[z^2+n(n+1)]\,w=0 \quad (n=0,\ \pm1,\ \pm2,\ ...), \qquad (4)$$

причем функции разных родов разлагаются в степенные ряды:

1-го рода:

$$i_n(x)=\sqrt{\frac{\pi}{2x}}\,I_{n+1/2}(x)=\frac{x^n}{(2n+1)!!}\left[1+\frac{x^2/2}{1!\,(2n+3)}+\right.$$
$$\left.+\frac{(x^2/2)^2}{2!\,(2n+3)(2n+5)}+...\right], \qquad (5)$$

2-го рода:

$$i_{-n}(x)=\sqrt{\frac{\pi}{2x}}\,I_{-n-1/2}(x)=\frac{(2n-1)!!}{(-1)^n x^{n+1}}\left[1+\frac{x^2/2}{1!\,(1-2n)}+\right.$$
$$\left.+\frac{(x^2/2)^2}{2!\,(1-2n)(3-2n)}+...\right], \qquad (6)$$

3-го рода:

$$k_n(x)=\frac{\pi}{2}(-1)^{n+1}[i_n(x)-i_{-n}(x)].$$

В двух следующих программах производится вычисление $i_n(x)$ и $i_{-n}(x)$ по формулам (5), (6). Суммирование в (5), (6) производится до тех пор, пока не совпадут суммы k и $k+1$ — слагаемых. Структура и обозначения такие же, как в двух предыдущих программах.

В контрольных примерах показано, что $i_0(2)=1.81343020$, $i_3(1)=0.0100650905$, $i_{-1}(2)=0.872881281$, $i_{-2}(1)=2.64671896$ [18].

```
JLIST
  10 PRINT "МОДИФ. СФЕР. ФУНКЦИЯ БЕССЕЛЯ I N (X)"
  20 INPUT "N=";N: INPUT "X=";X1
 100 I=1:F=1:T=1:X=X1*X1/2:K=1
 110 T=T*X/K/(2*(N+K)+1):F=I+T:K=K+1
 120 IF I<>F THEN I=F: GOTO 110
 130 F=1: IF N=0 THEN 150
 140 FOR K=1 TO N:F=F/(2*K+1)*X1: NEXT
 150 I=I*F
1000 PRINT "I"N" ("X1")="I
1010 END

JRUN
МОДИФ. СФЕР. ФУНКЦИЯ БЕССЕЛЯ I N (X)
N=0
X=2
I0 (2) =1.81343021

JRUN
МОДИФ. СФЕР. ФУНКЦИЯ БЕССЕЛЯ I N (X)
N=3
X=1
I3 (1) =.0100650905

JLIST
  10 PRINT "МОДИФ. СФЕР. ФУНКЦИЯ БЕССЕЛЯ I [-N] (X)"
  20 INPUT "N=";N: INPUT "X=";X1
 100 I=1:T=1:X=X1*X1/2:K=1
 110 T=T*X/(K*(2*(K-N)-1)):F=1+T:K=K+1
 120 IF I<>F THEN I=F: GOTO 110
 130 F=1/X1: IF N=0 THEN 150
 140 FOR K=1 TO N:F=-F*(-1+2*K)/X1: NEXT
 150 I=I*F
1000 PRINT "I [-"N"] ("X1")="I
1010 END

JRUN
МОДИФ. СФЕР. ФУНКЦИЯ БЕССЕЛЯ I [-N] (X)
N=1
X=2
```

```
I [-1] (2) = .872881281
JRUN
МОДИФ. СФЕР. ФУНКЦИЯ БЕССЕЛЯ I [-N] (X)
N=2
X=1
I [-2] (1)=2.64671896
```

Функции Эйри $Ai(x)$ и $Bi(x)$ являются линейно независимыми решениями дифференциального уравнения

$$w'' - zw = 0.$$

Они разлагаются в степенные ряды [18]

$$Ai(x) = C_1 f(x) - C_2 g(x),$$

$$Bi(x) = \sqrt{3}\,[C_1 f(x) + C_2 g(x)],$$

$$f(x) = 1 + \frac{x^3}{3!} + \frac{1\cdot 4}{6!}x^6 + \frac{1\cdot 4\cdot 7}{9!}x^9 + \ldots,$$

$$g(x) = x + \frac{2}{4!}x^4 + \frac{2\cdot 5}{7!}x^7 + \frac{2\cdot 5\cdot 8}{10!}x^{10} + \ldots, \qquad (7)$$

где $C_1 = Ai(0) = 0.3550280538878$,
$C_2 = \Gamma(1/3)/\sqrt[3]{3} = 0.2588194037928$.

В приведенной ниже программе производится вычисление значений $Ai(x)$ и $Bi(x)$ по разложению (7). Вычисления продолжаются, пока не совпадут значения двух последовательных сумм в (7).

В программе используются переменные: $X1$ — аргумент; X — рабочая переменная (x^3); A, B — значения функций $Ai(x)$, $Bi(x)$; $T1$, $T2$ — значения слагаемых в (7); K — переменная цикла; $C1$, $C2$ — значения c_1, c_2; F, G, $G1$ — рабочие переменные.

Структура программы:

10—20 — ввод x;

100—180 — вычисления по формуле (7);

1000—1010 — вывод значений $Ai(x)$, $Bi(x)$.

В контрольных примерах показано, что [18] $Ai(0.8) = 0.16984632$, $Bi(0.8) = 1.04242217$ и $Ai(-3) = -0.37881429$, $Bi(-3) = -0.19828963$. Надежность вычислений тем лучше, чем меньше значение $|x|$.

```
JLIST
  10 PRINT "ФУНКЦИИ ЭЙРИ AI (X), BI (X)"
  20 INPUT "X="; X1
 100 X=X1:F=1:K=1: T1=1:G=X: T2=X:X=X*X*X
 110 C1=.355028053887817: C2=.258819403792807
 120 T1=T1*X/ (3*K* (3*K-1)): T2=T2*X/((3*K+1)*3*K)
 130 F1=F+T1: G1=G+T2
 140 IF F1<>F OR G1<>G THEN F=F1:G=G1:K=K+1: GOTO 120
 150 A=C1*F-C2*G:B=(C1*F+C2*G)*SQR (3)
1000 PRINT "AI ("X1")="A, "BI ("X1")="B
1010 END
```

```
JRUN
ФУНКЦИИ ЭЙРИ AI (X), BI (X)
X=.8
AI (.8) =.169846317                    BI (.8) =1.04242217
JRUN
ФУНКЦИИ ЭЙРИ AI (X), BI (X)
X=-3
AI (-3) =-.378814293                   BI (-3)=-.198289626
```

7.8. Интеграл вероятностей и интегралы Френеля

Интеграл ошибок erf (x) определяется равенством

$$\operatorname{erf}(x)=-\operatorname{erf}(-x)=\frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}\,dt, \tag{1}$$

erf (x) разлагается в степенные ряды

$$\operatorname{erf}(x)=\frac{2}{\sqrt{\pi}}\sum_{k=0}^{\infty}(-1)^k\frac{x^{2k+1}}{k!\,(2k+1)}, \tag{2}$$

$$\operatorname{erf}(x)=\frac{2}{\sqrt{\pi}}e^{-x^2}\sum_{k=0}^{\infty}\frac{2^k x^{2k+1}}{(2k+1)!!}. \tag{3}$$

Аппроксимация рациональной функцией имеет вид [18]:

$$\operatorname{erf}(x)=1-(a_1t+a_2t^2+\ldots+a_5t^5)\,e^{-x^2}+\varepsilon(x), \tag{4}$$

где $t=(1+px)^{-1}$, $|\varepsilon(x)|\leqslant 1.5\cdot 10^{-7}$, $p=0.3275911$, $a_1=0.254829592$, $a_2=-0.284496736$, $a_3=1.421413741$, $a_4=-1.453152027$, $a_5=1.061405429$.

В первой программе производится вычисление erf (x) по разложению в ряд (2) с абсолютной точностью ε_1.

В программе используются переменные: X, E — аргумент и значение erf (x); $E1$ — точность вычислений; $T1$ — значение k-го слагаемого в (2); $X2$ — рабочая переменная, равная x^2; T — рабочая переменная, равная $(-1)^k x^{2k+1}/k!$; $2/\sqrt{\pi}=1.128379167$.

Структура программы:

10 — 30 — ввод x, задание ε_1;

100 — 120 — вычисление по формуле (2);

1000 — 1010 — вывод значения erf (x).

В контрольном примере найдено значение erf (2). По таблицам [18] оно равняется 0.9953222650, что с точностью, лучшей чем 10^{-8}, совпадает с вычислениями.

```
JLIST
  10 PRINT "ИНТЕГРАЛ ВЕРОЯТНОСТЕЙ ERF (X)"
  20 INPUT "X=";X
  30 E1=1E-8
 100 T=1.128379167*X:E=T:K=1:X2=X*X
```

```
 110 T=-T*X2/K: T1=T/(K+K+1):E=E+T1:K=K+1
 120 IF ABS (T1)>E1 THEN 110
1000 PRINT "ERF ("X")="E
1010 END

JRUN
ИНТЕГРАЛ ВЕРОЯТНОСТЕЙ ERF (X)
X=2
ERF (2)=.995322264
```

Во второй программе вычисление erf (x) производится по аппроксимации (4).

В программе используются переменные: X, E — аргумент и значение erf (x); T — рабочая переменная t в (4).

Структура программы:

10—20 — ввод x;

100—110 — вычисления по формуле (4);

1000—1010 — вывод значения erf (x).

Контрольный пример тот же, что и в предыдущей программе.

```
JLIST
  10 PRINT "ИНТЕГРАЛ ВЕРОЯТНОСТЕЙ ERF (X)"
  20 INPUT "X=";X
 100 T=1/(1+.3275911*X)
 110 E=1-EXP (-X*X)*((((1.061405429*T-1.453152027)*T
     +1.421413741)*T-.284496736)*T+.254829592)*T
1000 PRINT "ERF ("X")="E
1010 END

JRUN
ИНТЕГРАЛ ВЕРОЯТНОСТЕЙ ERF (X)
X=2
ERF (2)=.995322139
```

Под **интегралами Френеля** подразумевают функции [18]

$$C(x)=\int_0^x \cos\frac{\pi t^2}{2}\,dt, \quad S(x)=\int_0^x \sin\frac{\pi t^2}{2}\,dt. \tag{5}$$

Степенные ряды для этих функций имеют вид:

$$C(x)=x\sum_{n=0}^{\infty}\frac{(-1)^n(\pi x^2/2)^{2n}}{(2n)!\,(4n+1)}, \quad S(x)=x\sum_{n=0}^{\infty}\frac{(-1)^n(\pi x^2/2)^{2n+1}}{(2n+1)!\,(4n+3)}. \tag{6}$$

Асимптотическое поведение этих функций для $|x|>>1$ описывается формулами

$$C(x)=\frac{1}{2}+f(x)\sin\frac{\pi x^2}{2}-g(x)\cos\frac{\pi x^2}{2},$$

$$S(x)=\frac{1}{2}-f(x)\cos\frac{\pi x^2}{2}-g(x)\sin\frac{\pi x^2}{2}, \tag{7}$$

где

$$\pi x f(x)=1+\sum_{n=1}^{\infty}(-1)^n\frac{1\cdot 3\ldots(4n-1)}{(\pi x^2)^{2n}},$$

$$\pi x g(x)=\sum_{n=0}^{\infty}(-1)^n\frac{1\cdot 3\ldots(4n+1)}{(\pi x^2)^{2n+1}}.$$

В следующей программе производится вычисление значений $C(x)$ и $S(x)$ по формулам (6) при $\pi x^2/2<13.1$ и с относительной точностью ε_1, по формулам (7) при $\pi x^2/2\geqslant 13.1$ абсолютная точность вычислений есть ε_1, но не более чем 10^{-6}. Полное определение точности вычислений по формуле (7) см. в [18].

В программе используются переменные: X, C, S — аргумент и значение функций $C(x)$ и $S(x)$; $E1$ — точность вычислений; $X2$, W — рабочие переменные ($X2=1/\pi x^2$, $w=\pi x^2/2$); $P1=\pi=3.1415\ldots$; N — переменная цикла в (6), (7); $S1$, $S2$, R — значения сумм и n слагаемых в (6) и (7); $H=\pm 0.5$ (плюс берется при $x\geqslant 0$); T, R — значения $\sin(\pi x^2/2)$ и $\cos(\pi x^2/2)$ соответственно.

Структура программы:

10 — 30 — ввод x, задание ε_1;

100 — 110 — $C(0)=S(0)=0$, начальные значения переменных;

130 — 190 — вычисления по формулам (7);

200 — 230 — вычисления по формулам (6);

1000 — 1010 — вывод $C(x)$ и $S(x)$.

Программа получена переводом на язык Бейсик Алгол-программы 244а [8].

В контрольных примерах вычислены значения $C(-4)$, $S(-4)$; $C(2)$, $S(2)$; $C(5)$, $S(5)$. По таблицам [18] имеем $C(-4)=-0.4984260$, $S(-4)=-0.4205158$; $C(2)=0.4882534$, $S(2)=0.3434157$; $C(5)=0.5636312$, $S(5)=0.4991914$.

```
JLIST
  10 PRINT "ИНТЕГРАЛЫ ФРЕНЕЛЯ C (X), S (X)"
  20 INPUT "X="; X
  30 E1=1E-8
 100 IF ABS (X)<1E-12 THEN C=0:S=0: GOTO 1000
 110 P1=3.14159265359:W=X * X/2 * P1:S1=0:S2=0:N=0:R=1
 120 IF W<13.1 THEN 200
 130 X2=.5/W: S1=S1+R:N=N+2:R=R * X2 * (N-1)
 140 S2=S2+R:N=N+2:R=-R * X2 * (N-1)
 150 IF ABS (R)>.5 * E1 AND N<33 THEN 130
 160 H=.5: IF X<0 THEN H=-.5
 170 R=COS (W):T=SIN (W):X2=P1 * X
 180 C=H+(T * S1-R * S2)/X2:S=H-(R * S1+T * S2)/X2
 190 GOTO 1000
 200 S1=S1+R:N=N+1:R=R * W * (N+N-1) / (N * (N+N+1))
 210 S2=S2+R:N=N+1:R=-R * W * (N+N-1) / (N * (N+N+1))
 220 IF ABS (R/S2)>E1 THEN 200
```

```
230 C=S1*X:S=S2*X
1000 PRINT "C ("X") ="C, "S ("X") ="S
1010 END

JRUN
ИНТЕГРАЛЫ ФРЕНЕЛЯ C (X), S (X)
X=-4
C (-4) = -.498426033 S (-4) = -.420515754
JRUN
ИНТЕГРАЛЫ ФРЕНЕЛЯ C (X), S (X)
X=2
C (2) =.488253408 S (2) =.34341568
JRUN
ИНТЕГРАЛЫ ФРЕНЕЛЯ C (X), S (X)
X=5
C (5) =.563631188 S (5) =.499191382
```

7.9. Гипергеометрические функции

Гипергеометрическая функция ${}_2F_1(a; b; c; z)$ может быть определена с помощью ряда Гаусса [18]

$$ {}_2F_1(a; b; c; z) = \frac{\Gamma(c)}{\Gamma(a)\Gamma(b)} \sum_{n=0}^{\infty} \frac{\Gamma(a+n)\Gamma(b+n)}{\Gamma(c+n)} \frac{z^n}{n!}. \qquad (1) $$

Кругом сходимости этого ряда является единичный круг $|z|=1$. Ряд Гаусса сводится к многочлену степени n относительно z, если $a=-n$ или $b=-n$ ($n=0, 1, 2, \ldots$). Ряд (1) теряет смысл, когда $c=-m$ ($m=0, 1, 2, \ldots$), а a или b не равны целому $n<m$.

В приведенной ниже программе вычисляется значение ${}_2F_1(a; b; c; z)$ с комплексными параметрами $a=a_1+ia_2$, $b=b_1+ib_2$, $c=c_1+ic_2$ и $z=z_1+iz_2$. Вычисления продолжаются, пока не совпадут два последующих значения суммы в (1). Если после m_1 итераций такая точность не достигается, то об этом выдается сообщение.

В программе используются переменные: $(A1, A2)$, $(B1, B2)$, $(C1, C2)$, $(Z1, Z2)$, $(S1, S2)$ — действительные и мнимые части соответственно a, b, c, z и значения функции ${}_2F_1$; $D=n!\,(|c+1|^2)$; T, $Y1$, $Y2$ — рабочие переменные; N — переменная цикла.

Структура программы:

10—40 — ввод a, b, c, z, задание m_1;

100—180 — вычисления по формуле (1);

1000—1010 — вывод значения ${}_2F_1$.

Программа получена переводом на язык Бейсик Алгол-программы 191б [7].

В контрольных примерах вычислены значения ${}_2F_1(1/2, 1, 3/2, -0.25) = \operatorname{arctg} 0.5 = 0.927295218$, ${}_2F_1(1, 1, 2, -0.6) = 5/3 \ln 1.6 = 0.78333938$, ${}_2F_1(1+i, i, 5+5i, 0.5i) =$

$=0.9061812-0.007551629i$, ${}_2F_1(2i, 1, 3+6i, 0.2+0.3i)=$
$=1.00174415+0.112605635i$.

```
JLIST
   10 PRINT "ГИПЕРГЕОМЕТРИЧЕСКАЯ ФУНКЦИЯ 2F1 (A, B; C; Z)"
   20 INPUT "A=(A1, A2)=";A1, A2: INPUT "B=(B1, B2)=";B1, B2
   30 INPUT "C=(C1, C2)=";C1, C2: INPUT "Z=(Z1, Z2)=";Z1, Z2
   40 M1=100
  100 S1=1: Y1=1: S2=0: Y2=0
  110 FOR N=0 TO M1
  120 D=(N+1)*((C1+N)*(C1+N)+C2*C2)
  130 T=((A1+N)*Y1-A2*Y2)/D: Y2=((A1+N)*Y2+A2*Y1)/D:
      Y1=T
  140 T=Y1*(B1+N)-Y2*B2: Y2=Y1*B2+Y2*(B1+N): Y1=T
  150 T=Y1*(C1+N)+Y2*C2: Y2=-Y1*C2+Y2*(C1+N): Y1=T
  160 T=Y1*Z1-Y2*Z2: Y2=Y1*Z2+Y2*Z1: Y1=T
  170 IF S1=S1+Y1 AND S2=S2+Y2 THEN 1000
  180 S1=S1+Y1:S2=S2+Y2
  190 NEXT
  200 PRINT "ТОЧНОСТЬ НЕ ДОСТИГНУТА"
 1000 PRINT "2F1=" S1"+I*("S2")"
 1010 END
RUN
ГИПЕРГЕОМЕТРИЧЕСКАЯ ФУНКЦИЯ 2F1 (A, B; C; Z)
A=(A1, A2)=.5,0
B=(B1, B2)=1,0
C=(C1, C2)=1.5,0
Z=(Z1, Z2)=-.25,0
2F1=.927295218+I*(0)

JRUN
ГИПЕРГЕОМЕТРИЧЕСКАЯ ФУНКЦИЯ 2F1 (A, B; C; Z)
A=(A1, A2)=1,0
B=(B1, B2)=1,0
C=(C1, C2)=2,0
Z=(Z1, Z2)=-.6,0
2F1=.783339382+I*(0)
JRUN
ГИПЕРГЕОМЕТРИЧЕСКАЯ ФУНКЦИЯ 2F1 (A, B; C; Z)
A=(A1, A2)=1,1
B=(B1, B2)=0,1
C=(C1, C2)=5,5
Z=(Z1, Z2)=0,.5
2F1=.906181208+I*(-7.55162905E-03)
JRUN
ГИПЕРГЕОМЕТРИЧЕСКАЯ ФУНКЦИЯ 2F1 (A, B; C; Z)
A=(A1, A2)=0,2
B=(B1, B2)=1,0
C=(C1, C2)=3,6
Z=(Z1, Z2)=.2,.3
2F1=1.00174415+I*(.112605635)
```

Вырожденная гипергеометрическая функция ${}_1F_1(a, c; z)$ может быть определена с помощью ряда

$$ {}_1F_1(a, c; z)=1+\frac{a}{c}z+\frac{1}{2!}\frac{a(a+1)}{c(c+1)}z^2+\ldots \quad (|z|<\infty). \qquad (2) $$

В приведенной ниже программе вычисляется значение $s_1+is_2={}_1F_1(a, c; z)$ с комплексными параметрами $a=a_1+ia_2$, $c=c_1+ic_2$, $z=z_1+iz_2$. Если прибавление следующего члена гипергеометрического ряда к сумме предыдущих членов не влияет на эту сумму, то вычисление заканчивается. Если же после m_1 итераций такая точность не достигается, то об этом выдается сообщение.

В программе используются переменные: (A1, A2), (C1, C2), (Z1, Z2), (S1, S2) — действительные и мнимые части a, c, z, ${}_1F_1$ соответственно; $D=n!\,|c+n|^2$; Y1, Y2 — рабочие переменные; N — переменная цикла.

Структура программы:

10—40 — ввод a, c, z, задание m_1;

100—190 — вычисления по формуле (2);

1000—1010 — вывод значения ${}_1F_1$.

Программа получена переводом на язык Бейсик Алгол-программы 192б [7].

В контрольном примере вычислены значения ${}_1F_1(1, 1, 2)=e^2=7.389056099$, ${}_1F_1(1+i, 3+4i, 1+i)=1.367817+0.3714654i$.

```
JLIST
  10 PRINT "ГИПЕРГЕОМЕТРИЧЕСКАЯ ФУНКЦИЯ 1F1 (A, C; Z)"
  20 INPUT "A=(A1, A2)=";A1, A2
  30 INPUT "C=(C1, C2)=";C1, C2: INPUT "Z=(Z1, Z2)=";Z1, Z2
  40 M1=100
 100 S1=1:Y1=1:S2=0: Y2=0
 110 FOR N=0 TO M1
 120 D=(N+1)*((C1+N)*(C1+N)+C2*C2)
 130 T=((A1+N)*Y1-A2*Y2)/D: Y2=((A1+N)*Y2+A2*Y1)/D:Y1=T
 140 T=Y1*(C1+N)+Y2*C2:Y2=-Y1*C2+Y2*(C1+N): Y1=T
 150 T=Y1*Z1-Y2*Z2: Y2=Y1*Z2+Y2*Z1:Y1=T
 160 IF S1=S1+Y1 AND S2=S2+Y2 THEN 1000
 170 S1=S1+Y1: S2=S2+Y2
 180 NEXT
 190 PRINT "ТОЧНОСТЬ НЕ ДОСТИГНУТА"
1000 PRINT "1F1=" S1"+I*("S2")"
1010 END

JRUN
ГИПЕРГЕОМЕТРИЧЕСКАЯ ФУНКЦИЯ 1F1 (A, C; Z)
A=(A1, A2)=1,0
C=(C1, C2)=1,0
Z=(Z1, Z2)=2,0
1F1=7.3890561+I*(0)
```

```
JRUN
ГИПЕРГЕОМЕТРИЧЕСКАЯ ФУНКЦИЯ 1F1 (A, C; Z)
A = (A1, A2) = 1,1
C = (C1, C2) = 3,4
Z = (Z1, Z2) = 1,1
2F1 = 1.3678173 + I * (.371465424)
```

7.10. Функция распределения вероятностей

Функция распределения вероятностей $P(x)$ определяет вероятность события $\{X \leqslant x\}$, где X — случайная величина, т. е. $P(x) = P\{X \leqslant x\}$. Для нормально распределенной непрерывной случайной величины с нулевым средним и единичной дисперсией имеем:

$$P(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-u^2/2} du. \tag{1}$$

Для функции $P(x)$ имеют место следующие разложения [18]: в степенной ряд:

$$P(x) = \frac{1}{2} + \frac{1}{\sqrt{2\pi}} \sum_{k=0}^{\infty} \frac{(-1)^k x^{2k+1}}{k!\, 2^k (2k+1)} \quad (x \geqslant 0), \tag{2}$$

в асимптотический ряд:

$$Q(x) = 1 - P(x) = \frac{e^{-x^2/2}}{\sqrt{2\pi}\, x} \left[1 + \sum_{k=1}^{\infty} \frac{(-1)^k (2k-1)!!}{x^{2k}} \right], \tag{3}$$

рациональная аппроксимация:

$$P(x) = 1 - \frac{e^{-x^2/2}}{\sqrt{2\pi}} (b_1 t + b_2 t^2 + \ldots + b_5 t^5) + \varepsilon(x), \tag{4}$$

где $t = 1/(1+px)$, $p = 0.2316419$, $|\varepsilon(x)| < 7.5 \cdot 10^{-8}$, $b_1 = 0.319381530$, $b_2 = -0.356563782$, $b_3 = 1.781477937$, $b_4 = -1.821275978$, $b_5 = 1.330274429$.

Отметим также соотношения

$$P(-x) = 1 - P(x), \quad Q(-x) = 1 - Q(x). \tag{5}$$

В приведенной ниже программе производится вычисление $P(x)$ при $x < 5$ и $Q(x)$ при $x \geqslant 5$. Вычисления производятся по формулам (2) и (3) с относительной точностью ε_1, причем количество слагаемых в (3) не превышает 12, что соответствует максимальной относительной точности в вычислениях $Q(x) = 6 \cdot 10^{-6}$, при этом относительная точность в значении $P(x) = 1 - Q(x)$ равна $1.6 \cdot 10^{-12}$.

В программе используются переменные: $X1$, P, Q — аргумент и значения вероятностей P, Q; P и Q — также рабочие переменные; $P2 = 1/\sqrt{2\pi} = 0.3989422804$; K — переменная цикла; $Q1$ — значение k-го слагаемого в (2);

X — рабочая переменная, равная x^2; $E1$ — относительная точность вычислений.

Структура программы:

10—30 — ввод x, задание ε_1;

100 — начальные значения переменных, проверка условия $|x|<5$;

110—140 — вычисления по формуле (3) при $|x|\geqslant 5$;

150—170 — вычисления по формуле (2) при $|x|<5$;

1000 — вывод значения $P(x)$;

1010 — вывод значения $Q(x)$.

В контрольных примерах показано, что [18] $P_2(2)=0.977249868$, $P(-2)=0.0227501318$, $P(-6)=9.8659\cdot 10^{-10}$, $Q(7)=1.2798\cdot 10^{-12}$.

```
JLIST
  10 PRINT "ФУНКЦИЯ ОШИБОК P (X) ИЛИ Q (X)=1-P (X)"
  20 INPUT "X="; X1
  30 E1=1E-8
 100 P=1:Q=1:K=1:X=X1*X1:P2=.39894228:IF X<25 THEN 150
 110 P=-P*(K+K-1)/X:Q=Q+P:K=K+1
 120 IF ABS (P/Q)>E1 AND K<13 THEN 110
 130 Q=Q*EXP (-X/2)*P2/X1:IF X1>0 THEN 1010
 140 P=-Q: GOTO 1000
 150 Q=-Q*X/K/2:Q1=Q/(K+K+1):P=P+Q1:K=K+1
 160 IF ABS (Q1/P)>E1 THEN 150
 170 P=.5+P2*P*ABS (X1):IF X1<0 THEN P=1-P
1000 PRINT "P (" X1")="P: END
1010 PRINT "Q(" X1")="Q: END

JRUN
ФУНКЦИЯ ОШИБОК P (X) ИЛИ Q (X)=1-P (X)
X=2
P (2)=.977249868
JRUN
ФУНКЦИЯ ОШИБОК P (X) ИЛИ Q (X)=1-P (X)
X=-2
P (-2)=.0227501318
JRUN
ФУНКЦИЯ ОШИБОК P (X) ИЛИ Q (X)=1-P (X)
X=-6
P (-6)=9.86587672E-10
JRUN
ФУНКЦИЯ ОШИБОК P (X) ИЛИ Q (X)=1-P (X)
X=7
Q (7)=1.27981255E-12
```

Во второй программе вычисление $P(x)$ производится на основании рациональной аппроксимации (4). При этом используются переменные: X, P — аргумент и значение $P(x)$; $P=1/(1+px)$ — рабочая переменная.

Контрольные примеры те же, что и в предыдущей программе.

```
JLIST
  10  PRINT "ПОЛИНОМИАЛЬНАЯ АППРОКСИМАЦИЯ ДЛЯ P (X)"
  20 INPUT "X=";X
 100 P=1/(1+.2316419*ABS (X))
 110 P=1-((((1.330274429 * P-1.821255978) * P+1.781477937) * P
     -.356563782) * P+.319381530) * P * EXP (-X * X * .5) * .3989422804
 120 IF X<0 THEN P=1-P
1000 PRINT "P ("X") ="P
1010 END

JRUN
ПОЛИНОМИАЛЬНАЯ АППРОКСИМАЦИЯ ДЛЯ P (X)
X=2
P (2) =.977249938
JRUN
ПОЛИНОМИАЛЬНАЯ АППРОКСИМАЦИЯ ДЛЯ P (X)
X=-2
P (-2) =.0227500619
```

В третьей программе вычисление $P(x)$ производится посредством полиномиальной аппроксимации, взятой из программы 209а [8].

В программе используются переменные: X, Z — аргумент и значение $P(x)$; Y, W — рабочие переменные.

Структура программы:

10—20 — ввод x;
100 — $P(0)=0.5$;
110—150 — вычисления при $|x|<6$;
160—200 — вычисление при $|x|\geqslant 6$;
1000—1010 — вывод значения $P(x)$.

```
JLIST
  10 PRINT "ИНТЕГРАЛ ОШИБОК"
  20 INPUT "X=";X
 100 IF X=0 THEN Z=0: GOTO 190
 110 Y=ABS (X)/2: IF Y>=3 THEN Z=1: GOTO 190
 120 IF Y>=1 THEN 160
 130 W=Y*Y
 140 Z=(((((((.000124818987 * W-.001075204047) * W+.005198775019) * W
     -.019198292004) * W+.059054035642) * W-.151968751364) * W
     +.319152932694) * W-.531923007300) * W+.797884560593) * Y * 2
 150 GOTO 190
 160 Y=Y-2
 170 Z=(((((((-.000045255659 * Y+.000152529290) * Y-.000019538132) * Y
     -.000676904986) * Y+.001390604284) * Y-.000794620820) * Y
     -.002034254874) * Y+.006549791214) * Y
 180 Z=((((((Z-.010557625006) * Y+.011630447319) * Y-.009279453341) * Y
```

```
     +.005353579108) * Y-.002141268741) * Y+.000535310849) * Y
     +.999936657524
 190 IF X>0 THEN Z=(Z+1)/2: GOTO 1000
 200 Z=(1-Z)/2
1000 PRINT "P("X")="Z
1010 END

JRUN
ИНТЕГРАЛ ОШИБОК
X=2
P (2)=.977249867
JRUN
ИНТЕГРАЛ ОШИБОК
X=-2
P (-2)=.0227501331
JRUN
ИНТЕГРАЛ ОШИБОК
X=7
P (7)=1
JRUN
ИНТЕГРАЛ ОШИБОК
X=-5
P (-5)=2.86847353E-07
```

7.11. Статистические расчеты

Пусть задан массив значений x_i $(i=\overline{1,\ n})$ случайной переменной. Тогда выборочное среднее значение величины x по данной выборке определяется равенством

$$\bar{x}=\frac{1}{n}\sum_{i=1}^{n} x_i. \tag{1}$$

Величины $s^2=\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2$, $\bar{s}^2=\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})^2$ (2)

называются смещенной и несмещенной дисперсиями.

Выборочные среднеквадратичные отклонения определяются формулами

$$s=\sqrt{s^2},\ \bar{s}=\sqrt{\bar{s}^2}. \tag{3}$$

Асимметрия A, эксцесс E и коэффициент вариации V равны:

$$A=\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^3,\ E=\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^4,\ V=\bar{s}/\bar{x}. \tag{4}$$

В приведенной ниже программе производится вычисление $\bar{x}$, s, $\bar{s}$, A и E по соотношениям

$$s^2=m_2-m_1^2;\quad A=m_3-3m_1m_2+2m_1^3;\quad E=m_4-4m_1m_3+6m_1^2m_2-3m_1^4, \tag{5}$$

где

$$m_k = \frac{1}{n} \sum_{i=1}^{n} x_i^k.$$

Предполагается, что данные не сгруппированы, их число известно и разрядность машины достаточна для того, чтобы не возникали переполнения.

В программе используются переменные: N, X, $M1$, $M2$, $M3$, $M4$ соответственно n, x_i, m_1, m_2, m_3, m_4. Эти же переменные используются как рабочие для накопления сумм $\sum x_i^k$. I — переменная цикла; S, $S1$, A, E соответственно s, $\bar{s}$, A, E.

Структура программы:

10—20 — вывод n;

30—70 — ввод x_i, вычисление m_k;

80—100 — вычисление s, $\bar{s}$, A, E;

1000—1010 — вывод $\bar{x}$, s, $\bar{s}$, A, E.

В контрольном примере рассмотрен случай $x_1=6$, $x_2=7$, $x_3=9$, $x_4=10$. Тогда $\bar{x}=8$, $s=\sqrt{2.5}$, $\bar{s}=\sqrt{10/3}$, $A=0$, $E=8.5$.

```
JLIST
  10 PRINT "СРЕДНЕЕ, ОТКЛОНЕНИЕ, АСИММЕТРИЯ, ЭКСЦЕСС"
  20 INPUT "N="; N
  30 M1=0:M2=0:M3=0:M4=0
  40 FOR I=1 TO N
  50 PRINT "X("I;: INPUT ")="; X
  60 M1=M1+X/N:M2=M2+X * X/N: M3=M3+X * X * X/N: M4=M4
     +X^4/N
  70 NEXT
  80 S=SQR (M2-M1 * M1):S1=S * SQR (N/(N-1))
  90 A=M3-3 * M1 * M2+2 * M1^3:E=M4-4 * M1 * M3+6 * M1 * M1 * M2
     -3 * M1^4
1000 PRINT "[X]="M1" ПЛЮС МИНУС" S1
1010 PRINT "S="S" A="A" E="E
1020 END

JRUN
СРЕДНЕЕ, ОТКЛОНЕНИЕ, АСИММЕТРИЯ, ЭКСЦЕСС
N=4
X (1)=6
X (2)=7
X (3)=9
X (4)=10
[X]=8 ПЛЮС МИНУС 1.82574186
S=1.58113883 A=0 E=8.50000382
```

Во второй программе производится расчет $\bar{x}$, s, $\bar{s}$, A и E по сгруппированным данным с неизвестным количеством точек x_i. Кратность точки x_i равна w_i (если $w_i=0$, то ввод новых значений x_i прекращается). Обозначения такие же, как в предыдущей программе, а также $X1=w_i x_i^k$, w — кратность x_i.

Структура программы:

10—70 — ввод w_i, x_i, образование сумм $\sum w_i x_i^k (k=\overline{1.4})$;

80—100 — вычисление $\bar{x}$, s, $\bar{s}$, A, E;

1000—1020 — вывод N, $\bar{x}$, $\bar{s}$, s, A, E.

```
JLIST
  10 PRINT "СТАТИСТИКА ПО СГРУППИРОВАННЫМ ДАННЫМ"
  20 I=1:N=0: M1=0: M2=0: M3=0: M4=0
  30 PRINT "ТОЧКА N%"I;: INPUT (W, X)=": W, X
  40 IF W=0 THEN 80
  50 N=N+W: X1=X*W: M1=M1+X1: X1=X1*X: M2=M2+X1
  60 X1=X1*X: M3=M3+X1: M4=M4+X*X1
  70 I=I+1: GOTO 30
  80 M1=M1/N: M2=M2/N: M3=M3/N: M4=M4/N
  90 S=SQR (M2-M1*M1):S1=S*SQR (N/(N-1))
 100 A=M3-3*M1*M2+2*M1^3:E=M4-4*M1*M3+6*M1*M1*M2-
     -3*M1^4
1000 PRINT "N="N: PRINT "[X]="M1" ПЛЮС МИНУС "S1
1010 PRINT "S="S" A="A" E="E
1020 END

JRUN
СТАТИСТИКА ПО СГРУППИРОВАННЫМ ДАННЫМ
ТОЧКА N%1 (W, X)=3,7
ТОЧКА N%2 (W, X)=4,8
ТОЧКА N%3 (W, X)=0,0
N=7
[X]=7.57142857 ПЛЮС МИНУС .534522488
S=.494871663 A=-.0349848587 E=.0649653375
```

Сглаживание эмпирических данных. Если нам заданы приближенные значения y_i некоторой функции $y(x)$ в равноотстоящих точках x_i, то, используя усреднение на основе интерполяционных формул, можно провести сглаживание зависимости $y=y(x_i)$, т. е. уточнить значение ординат y_i.

В первой программе производится линейное сглаживание по трем ординатам с помощью формул

$$\tilde{y}_0=(5y_0+2y_1-y_2)/6,$$
$$\tilde{y}_i=(y_{i-1}+y_i+y_{i+1})/3,\ i=\overline{1,\ n-1},$$
$$\tilde{y}_n=(5y_n+2y_{n-1}-y_{n-2})/6. \qquad (6)$$

Программе используются переменные: N — количество интервалов n; $Y(N)$ — ординаты эмпирических точек, затем сглаженные ординаты $\tilde{y}_i$; I — переменная цикла; A, B, C — рабочие переменные.

Структура программы:

10—30 — ввод значения $n \geqslant 2$;

40 — ввод y_i, $i=\overline{0,\ n}$;

100 — вычисление $\tilde{y}_0$;

110—130 — вычисление значений $\tilde{y}_i$, $i=\overline{1,\ n-1}$;

140 — вычисление $\tilde{y}_n$;

1000—1010 — вывод значений $\tilde{y}_i$, $i=\overline{0,\ n}$.

Программа получена переводом на язык Бейсик Алгол-программы 188б [7].

В контрольном примере проведено сглаживание функции $y=x^2$ для $x=(0, 1, 2, 3, 4)$. Ручной счет дает результаты: $y=(-1/3,\ 5/3,\ 14/3,\ 29/3,\ 47/3)$.

```
JLIST
  10 PRINT "ЛИНЕЙНОЕ СГЛАЖИВАНИЕ ПО ТРЕМ ТОЧКАМ"
  20 INPUT "N=": N: IF N<2 THEN PRINT "N>=2": GOTO 20
  30 DIM Y (N)
  40 FOR I=0 TO N: PRINT "Y ("I;: INPUT ")=": Y (I): NEXT
 100 A=Y (0):C=Y (N-2):Y (0)=(5*A+2*Y (1)-Y (2))/6
 110 FOR I=1 TO N-1
 120 B=Y (I):Y (I)=(A+B+Y (I+1))/3:A=B
 130 NEXT
 140  Y (N)=(-C+2*B+5*Y (N))/6
1000 PRINT "ПОСЛЕ СГЛАЖИВАНИЯ:"
1010 FOR I=0 TO N: PRINT "Y("I")="Y (I): NEXT
1020 END

JRUN
ЛИНЕЙНОЕ СГЛАЖИВАНИЕ ПО ТРЕМ ТОЧКАМ
N=4
Y (0)=0
Y (1)=1
Y (2)=4
Y (3)=9
Y (4)=16
ПОСЛЕ СГЛАЖИВАНИЯ:
Y (0)=-.333333333
Y (1)=1.66666667
Y (2)=4.66666667
Y (3)=9.66666667
Y (4)=15.6666667
```

В следующей программе производится сглаживание по пяти точкам на основании формул

$$\tilde{y}_0=(69y_0+4y_1-6y_2+4y_3-y_4)/70,$$
$$\tilde{y}_1=(2y_0+27y_1+12y_2-8y_3+2y_4)/35,$$
$$\tilde{y}_i=(-3y_{i-2}+12y_{i-1}+17y_i+12y_{i+1}-3y_{i+2})/35,\ i=\overline{2,\ n-2},$$
$$\tilde{y}_{n-1}=(2y_{n-4}-8y_{n-3}+12y_{n-2}+27y_{n-1}+2y_n)/35,$$
$$y_n=(-y_{n-4}+4y_{n-3}-6y_{n-2}+4y_{n-1}+69y_n)/70. \qquad (7)$$

В программе используются переменные: N — количество интервалов n; $Y(N)$ — ординаты точек, затем сглаженные ординаты $\tilde{y}_i$; I — переменная цикла; A, B, S, T — рабочие переменные.

Структура программы такая же, как в предыдущей программе. Программа получена переводом на Бейсик Алгол-программы 189б [7].

В контрольном примере проведено сглаживание функции $y=x^2+1$ при $x=(0, 1, 2, 3, 4)$. Результаты ручного счета совпадают с вычислениями $\tilde{y}_i=(1, 2, 5, 10, 17)$.

```
JLIST
  10 PRINT "ЛИНЕЙНОЕ СГЛАЖИВАНИЕ ПО ПЯТИ ТОЧКАМ"
  20 INPUT "N=";N: IF N<4 THEN PRINT "N>=4": GOTO 20
  30 DIM Y (N)
  40 FOR I=0 TO N: PRINT "Y("I;: INPUT ")=";Y (I): NEXT
 100 A=Y (0):B=Y (1):S=Y (N-4):T=Y (N-1)
 110 Y (0)=(69*A+4*B-6*Y (2)+4*Y (3)-Y (4))/70
 120 Y (1)=(2*A+27*B+12*Y (2)-8*Y (3)+2*Y (4))/35
 130 FOR I=2 TO N-2
 140 C=Y (I):Y (I)=(-3*A+12*B+17*C+12*Y (I+1)-3*Y (I+
     +2))/35:A=B:B=C
 150 NEXT
 160 Y (N-1)=(-8*A+12*B+2*S+27*T+2*Y (N))/35
 170 Y (N)=(4*A-6*B-S+4*T+69*Y (N))/70
1000 PRINT "ПОСЛЕ СГЛАЖИВАНИЯ:"
1010 FOR I=0 TO N: PRINT "Y("I")="Y (I): NEXT
1020 END

JRUN
ЛИНЕЙНОЕ СГЛАЖИВАНИЕ ПО ПЯТИ ТОЧКАМ
N=4
Y (0)=1
Y (1)=2
Y (2)=5
Y (3)=10
Y (4)=17
ПОСЛЕ СГЛАЖИВАНИЯ:
Y (0)=1
Y (1)=2
Y (2)=5
Y (3)=10
Y (4)=17
```

Метод наименьших квадратов. Пусть некоторая функция $y=y(x)$ задана n парами значений абсцисс x_i и ординат y_i. Задача линейного регрессионного анализа состоит в нахождении такой линии регрессии, чтоб сумма квадратов отклонений вдоль оси OY были минимальной. В приведенной ниже программе уравнение регрессии имеет вид:

$$y=b_0\varphi_0(x)+b_1\varphi_1(x), \tag{8}$$

где $\varphi_0(x)$ и $\varphi_1(x)$ — произвольные функции.

Требование минимальности выражения

$$\sum_{i=1}^{n} [y_i - b_0\varphi_0(x_i) - b_1\varphi_1(x_i)]^2$$

приводит к следующим значениям для коэффициентов b_0 и b_1:

$$b_0 = \frac{\overline{y\varphi_0}\,\overline{\varphi_1^2} - \overline{y\varphi_1}\,\overline{\varphi_0\varphi_1}}{\overline{\varphi_0^2}\,\overline{\varphi_1^2} - (\overline{\varphi_0\varphi_1})^2}, \quad b_1 = \frac{\overline{\varphi_0^2}\,\overline{y\varphi_1} - \overline{\varphi_0\varphi_1}\,\overline{y\varphi_0}}{\overline{\varphi_0^2}\,\overline{\varphi_1^2} - (\overline{\varphi_0\varphi_1})^2}, \tag{9}$$

где $\overline{y\varphi_0} = \left(\sum_{i=1}^{n} y_i\varphi_0(x_i)\right)/n$, $\overline{\varphi_1^2} = \left(\sum_{i=1}^{n} \varphi_1^2(x_i)\right)/n$ и т. д.

В программе используются переменные: N, X, Y — количество точек n, значения x_i, y_i, $F0$, $F1$ — значения функций $\varphi_0(x_i)$ и $\varphi_1(x_i)$; I — переменная цикла; $B0$, $B1$ — коэффициенты b_0, b_1; $A1 = \overline{y\varphi_0}$, $A2 = \overline{\varphi_1^2}$, $A3 = \overline{y\varphi_1}$, $A4 = \overline{\varphi_0\varphi_1}$, $A5 = \overline{\varphi_0^2}$, $A6 = \overline{\varphi_1^2}$; Y — знаменатель в (9).

Структура программы:

10—20 — ввод n, начальные значения переменных;

40 — ввод x_i, y_i;

50 — вычисление $\varphi_0(x_i)$ и $\varphi_1(x_i)$;

60—70 — вычисление сумм в равенствах (9);

90 — вычисление знаменателя в равенствах (9) и коэффициентов b_0, b_1;

500—510 — подпрограмма вычислений $\varphi_0(x)$ и $\varphi_1(x)$;

1000—1010 — вывод b_0 и b_1.

В контрольном примере найдены коэффициенты уравнения прямой $y = b_0 + b_1 x$ для следующей последовательности пар x_i, y_i: $x_i = (1.5; 4; 5; 7; 8.5; 10; 11; 12.5)$, $y_i = (5, 4.5; 7; 6.5; 9.5; 9; 11; 5.9)$. В приведенном примере ручной счет дает $b_0 = 4.342$, $b_1 = 0.401$.

```
JLIST
  10 PRINT "ЛИНЕЙНАЯ РЕГРЕССИЯ"
  20 INPUT "N="; N: A1=0: A2=0: A3=0: A4=0: A5=0: A6=0
  30 FOR I=1 TO N
  40 INPUT "(X, Y)="; X, Y
  50 GOSUB 500
  60 A1=A1+Y*F0/N: A2=A2+F1*F1/N: A3=A3+Y*F1/N
  70 A4=A4+F0*F1/N: A5=A5+F0*F0/N: A6=A6+F1*F1/N
  80 NEXT
  90 Y=A5*A6-A4*A4:  B0=(A1*A2-A3*A4)/Y:  B1=(A5*A3-A4*
     A1)/Y
 100 GOTO 1000
 500 F0=1
 510 F1=X: RETURN
1000 PRINT "B0=" B0, "B1=" B1
1010 END

JRUN
ЛИНЕЙНАЯ РЕГРЕССИЯ
N=8
(X, Y)=1.5, 5
```

```
(X, Y) = 4, 4.5
(X, Y) = 5, 7
(X, Y) = 7, 6.5
(X, Y) = 8.5, 9.5
(X, Y) = 10, 9
(X, Y) = 11, 11
(X, Y) = 12, 5.9
B0 = 4.34218543 B1 = .401059603
```

Иногда удобно после нахождения коэффициентов регрессии b_0 и b_1 построить график регрессии $y = y(x)$. В таком случае программу надо изменить начиная со строки 1010:

```
1010 INPUT "ВВЕДИТЕ НАЧ. И КОН. ЗНАЧ. X: (X1, X2) ="; X1, X2
1020 INPUT "КОЛ. ТОЧЕК M ="M
1030 H = (X2 - X1) / (M - 1)
1040 FOR J = 1 TO M
1050 X = X1 + H * (J - 1): GOSUB 500: Y = B0 * F0 + B1 * F1
1060 PRINT "X ="X, "Y ="Y
1070 NEXT: END
ВВЕДИТЕ НАЧ. И КОН. ЗНАЧ. X: (X1, X2) = 1,12
КОЛ. ТОЧЕК M = 5
X = 1          Y = 4.74324503
X = 3.75       Y = 5.84615894
X = 6.5        Y = 6.94907285
X = 9.25       Y = 8.05198675
X = 12         Y = 9.15490067
```

Глава VIII

РАЗНЫЕ АЛГОРИТМЫ И ПРОГРАММЫ

8.1. Площадь многоугольника

Если координаты вершин n-угольника равны (x_i, y_i) $(i=\overline{1, n})$, то площадь многоугольника может быть вычислена по формуле

$$S=\frac{1}{2}\left|\sum_{i=0}^{n-1}(x_i+x_{i+1})(y_i-y_{i+1})\right|, \qquad (1)$$

где $(x_0, y_0)=(x_n, y_n)$.

Структуру формулы (1) на примере треугольника можно понять по рисунку 1.

Действительно, $S_{A_1A_2A_3}=S_{y_1A_1A_2y_2}+S_{y_2A_2A_3y_3}-S_{y_1A_1A_3y_3}$, а, например, площадь трапеции $y_1A_1A_2y_2$ равна $S=\frac{1}{2}|(y_2-y_1)(x_1+x_2)|$.

Отметим, что площадь фигуры, заданной точками A_1, ..., A_n $(n\geqslant 4)$, существенно зависит от порядка, в котором заданы вершины. Так, если вершины будут размещены на плоскости в том порядке, как изображено на рисунке 2, то получим фигуру, называемую четырехвершинником (его площадь заштрихована).

В первой программе расчет по формуле (1) производится после ввода координат вершин (x_i, y_i). Во второй программе такой расчет ведется в процессе ввода.

В программе используются переменные: N — количество вершин многоугольника; S — площадь многоугольника (многовершинника); $X(N)$, $Y(N)$ — координаты вершин (в первой программе); X, $Y(A, B)$ — координаты $i+1$ (i)-й верши-

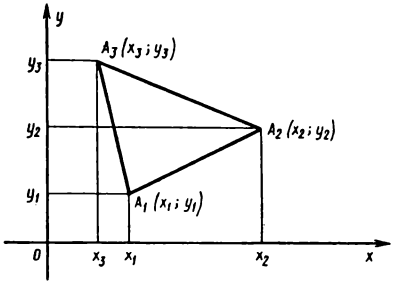

Рис. 1

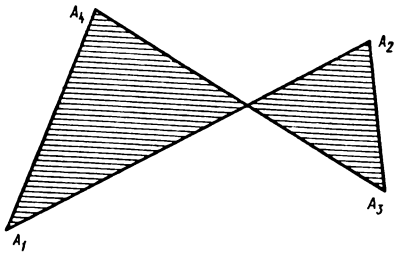

Рис. 2

ны (во второй программе); $X1$, $Y1$ — координаты первой вершины во второй и $X(0)$, $Y(0)$ — в первой программе; I — переменная цикла.

В контрольном примере найдена площадь четырехвершинника $A_1(1, 2)$, $A_2(5, 4)$, $A_3(3, 5)$, $A_4(-2, 3)$ и четырехвершинника $A_1A_3A_2A_4$. Ручные вычисления приводят к такому же результату: $S_1=9.5$, $S_2=1$.

```
JLIST
  10 PRINT "ПРОГРАММА ВЫЧИСЛЯЕТ ПЛОЩАДЬ N-УГОЛЬНИКА"
  20 INPUT "КОЛИЧЕСТВО ВЕРШИН N=": N
  40 DIM X (N), Y (N)
  50 PRINT "ВВЕДИТЕ КООРДИНАТЫ (X, Y) ВЕРШИН:"
  60 FOR I=1 TO N: PRINT "(X ("I"), Y ("I";: INPUT ")) =";X (I), Y (I): NEXT
 100 X (0)=X (N):Y (0)=Y (N):S=0
 110 FOR I=0 TO N-1
 120 S=S+(X (I)+X (I+1)) * (Y (I)-Y (I+1))
 130 NEXT
 140 S=ABS (S)/2
1000 PRINT "ПЛОЩАДЬ S="S
1010 END

JRUN
ПРОГРАММА ВЫЧИСЛЯЕТ ПЛОЩАДЬ N-УГОЛЬНИКА
КОЛИЧЕСТВО ВЕРШИН N=4
ВВЕДИТЕ КООРДИНАТЫ (X, Y) ВЕРШИН.
(X (1), Y (1)) =1, 2
(X (2), Y (2)) =5, 4
(X (3), Y (3)) =3, 5
(X (4), Y (4)) =-2, 3
ПЛОЩАДЬ S=9.5
JRUN
ПРОГРАММА ВЫЧИСЛЯЕТ ПЛОЩАДЬ N-УГОЛЬНИКА
КОЛИЧЕСТВО ВЕРШИН N=4
ВВЕДИТЕ КООРДИНАТЫ (X, Y) ВЕРШИН:
(X (1), Y (1)) =1, 2
(X (2), Y (2)) =3, 5
(X (3), Y (3)) =5, 4
(X (4), Y (4)) =-2, 3
ПЛОЩАДЬ S=1
JLIST
  10 PRINT "ПЛОЩАДЬ МНОГОУГОЛЬНИКА."
  20 INPUT "КОЛИЧЕСТВО ВЕРШИН N=": N
  30 S=0:I=1
  40 PRINT "ТОЧКА N%"I;: INPUT " (X, Y) =": X, Y: X1=X: Y1=Y
  50 I=I+1: IF I=N+1 THEN 90
  60 PRINT "ТОЧКА N%"I;: INPUT " (X, Y) =": A, B
  70 GOSUB 500:X=A:Y=B
  80 GOTO 50
  90 A=X1:B=Y1: GOSUB 500
```

```
 100 S=ABS (S)/2: GOTO 1000
 500 S=S+(X-A) * (Y+B): RETURN
1000 PRINT "S="S
1010 END
JRUN
ПЛОЩАДЬ МНОГОУГОЛЬНИКА.
КОЛИЧЕСТВО ВЕРШИН N=4
ТОЧКА N%1 (X, Y)=1, 2
ТОЧКА N%2 (X, Y)=5, 4
ТОЧКА N%3 (X, Y)=3, 5
ТОЧКА N%4 (X, Y)=-2, 3
S=9.5
```

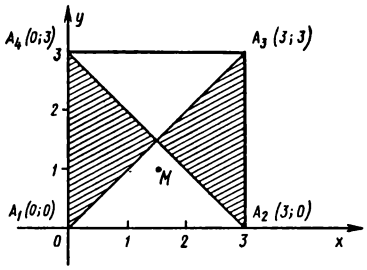

Рис. 3

8.2. Положение точки относительно n-угольника

Программа определяет, лежит ли точка с координатами (x_0, y_0) внутри n-угольника (точнее, n-вершинника), заданного координатами вершин (x_i, y_i), $i=\overline{1, n}$.

В программе используются переменные: N — количество вершин (n); $X(N+1)$, $Y(N+1)$ — массивы, содержащие координаты точки (x_0, y_0) и координаты вершины (x_i, y_i), где $i=\overline{1, n}$; $X(N+1)=X(1)$, $Y(N+1)=Y(1)$; B — если точка лежит внутри n-вершинника, то $B=1$, в противном случае $B=0$. Программа получена переводом на язык Бейсик Алгол-программы 112б [6].

В контрольном примере показано, что точка $M(1.5; 1)$ лежит внутри квадрата $A_1A_2A_3A_4$ и не лежит внутри четырехвершинника $A_1A_3A_2A_4$ (см. рис. 3). Координаты точек равны: $A_1(0; 0)$, $A_2(3; 0)$, $A_3(0; 3)$, $A_4(3; 3)$.

```
JLIST
  10 PRINT "ПОЛОЖЕНИЕ ТОЧКИ ОТНОСИТЕЛЬНО N-УГОЛЬНИКА"
  20 INPUT "N="; N: DIM X (N+1), Y (N+1)
  30 INPUT "КООРДИНАТЫ ТОЧКИ X, Y="; X (0), Y (0)
  40 PRINT "КООРДИНАТЫ ВЕРШИН "N"-УГОЛЬНИКА:"
  50 FOR I=1 TO N
  60 PRINT "X ("I"), Y ("I";: INPUT ")="; X (I), Y (I)
  70 NEXT
 100 X (N+1)=X (1):Y (N+1)=Y (1):B=1
 110 FOR I=1 TO N
 120 IF (Y (0)>Y (I))<>(Y (0)<=Y (I+1)) THEN 140
 130 IF X (0)-X (I)<(Y (0)-Y (I)) * (X (I+1)-X (I))/(Y (I+1)-Y (I))
     THEN GOSUB 500
 140 NEXT
 150 GOSUB 500: GOTO 1000
```

```
 500 IF B=0 THEN B=1: GOTO 520
 510 B=0
520 RETURN
1000 PRINT "ТОЧКА ";
1010 IF B=0 THEN PRINT "НЕ",
1020 PRINT "ЛЕЖИТ ВНУТРИ "N"-УГОЛЬНИКА."
1030 END

JRUN
ПОЛОЖЕНИЕ ТОЧКИ ОТНОСИТЕЛЬНО N-УГОЛЬНИКА
N=4
КООРДИНАТЫ ТОЧКИ X, Y=1.5, 1
КООРДИНАТЫ ВЕРШИН 4-УГОЛЬНИКА:
X (1), Y (1)=0, 0
X (2), Y (2)=3, 0
X (3), Y (3)=3, 3
X (4), Y (4)=0, 3
ТОЧКА ЛЕЖИТ ВНУТРИ 4-УГОЛЬНИКА.
JRUN
ПОЛОЖЕНИЕ ТОЧКИ ОТНОСИТЕЛЬНО N-УГОЛЬНИКА
N=4
КООРДИНАТЫ ТОЧКИ X, Y=1.5, 1
КООРДИНАТЫ ВЕРШИН 4-УГОЛЬНИКА:
X (1), Y (1)=0, 0
X (2), Y (2)=3, 3
X (3), Y (3)=3, 0
X (4), Y (4)=0, 3
ТОЧКА НЕ ЛЕЖИТ ВНУТРИ 4-УГОЛЬНИКА.
```

8.3. Кратчайший путь

Программа находит кратчайший путь от точки i сети до точки j. Если прямой связи между точками не существует, то необходимо на входе задать $M(I, J)=1E+20$. Если такой связи не существует вообще, то и на выходе $M(I, J)=1E+20$.

Программа получена переводом на язык Бейсик Алгол-программы 97б [5].

В программе используются переменные: N — размерность матрицы расстояний (количество точек); $M(N, N)$ — массив, содержащий вначале исходные расстояния между пунктами, на выходе — кратчайшие расстояния; I, J, K — переменные циклов.

В контрольном примере было задано ($N=4$)

$$M=\begin{pmatrix} 0 & 20 & 50 & 10 \\ 20 & 0 & 15 & 10^{20} \\ 50 & 15 & 0 & 30 \\ 10 & 10^{20} & 30 & 0 \end{pmatrix}$$

и получен правильный результат:

$$M=\begin{pmatrix} 0 & 20 & 35 & 10 \\ 20 & 0 & 15 & 30 \\ 35 & 15 & 0 & 30 \\ 10 & 30 & 30 & 0 \end{pmatrix}$$

```
JLIST
  10 PRINT "КРАТЧАЙШИЙ ПУТЬ!"
  20 INPUT "ВВЕДИТЕ КОЛИЧЕСТВО ТОЧЕК N=";N: DIM M (N, N)
  30 PRINT "ВВЕДИТЕ РАССТОЯНИЯ МЕЖДУ ТОЧКАМИ!": PRINT "ЕСЛИ
     ПУТИ НЕТ ВВЕДИТЕ 1E+20"
  40 FOR I=1 TO N: FOR J=1 TO N: IF I=J THEN 60
  50 PRINT "M("I", "J;: INPUT ")=";M (I, J)
  60 NEXT J: NEXT I
 100 FOR I=1 TO N: FOR J=1 TO N: IF M (J, I)>1E+19 THEN 160
 110 FOR K=1 TO N
 120 IF M (I, K)>1E+19 THEN 150
 130 S=M (J, I)+M (I, K)
 140 IF S<M (J, K) THEN M (J, K)=S
 150 NEXT
 160 NEXT J,I
1000 PRINT "МАТРИЦА КРАТЧАЙШИХ РАССТОЯНИЙ:"
1010 FOR J=1 TO N: FOR K=1 TO N
1020 PRINT "M("J", "K")="M (J, K)" ";
1030 NEXT: PRINT: NEXT
1040 END

JRUN
КРАТЧАЙШИЙ ПУТЬ!
ВВЕДИТЕ КОЛИЧЕСТВО ТОЧЕК N=4
ВВЕДИТЕ РАССТОЯНИЯ МЕЖДУ ТОЧКАМИ!
ЕСЛИ ПУТИ НЕТ ВВЕДИТЕ 1E+20
M (1, 2)=20
M (1, 3)=50
M (1, 4)=10
M (2, 1)=20
M (2, 3)=15
M (2, 4)=1E+20
M (3, 1)=50
M (3, 2)=15
M (3, 4)=30
M (4, 1)=10
M (4, 2)=1E+20
M (4, 3)=30
МАТРИЦА КРАТЧАЙШИХ РАССТОЯНИЙ:
M (1, 1)=0 M (1, 2)=20 M (1, 3)=35 M (1, 4)=10
M (2, 1)=20 M (2, 2)=0 M (2, 3)=15 M (2, 4)=30
M (3, 1)=35 M (3, 2)=15 M (3, 3)=0 M (3, 4)=30
M (4, 1)=10 M (4, 2)=30 M (4, 3)=30 M (4, 4)=0
```

8.4. Сортировка по величине

Если задана произвольная последовательность чисел $x_1, x_2, \ldots, x_n$, то программа преобразует эту последовательность в неубывающую последовательность, которая помещается на место старой:

$$x_1 \leqslant x_2 \leqslant x_3 \leqslant \ldots \leqslant x_n.$$

В программе используются переменные: N — количество чисел (n); $X(N)$ — массив чисел x_i; T — рабочая переменная.

Программа получена переводом на язык Бейсик Алгол-программы 175б [7].

В контрольном примере последовательность (3, 7, 5, 3, —2, 0) преобразована в неубывающую последовательность (—2, 0, 3, 3, 5, 7).

```
JLIST
  10 PRINT "СОРТИРОВКА N ЧИСЕЛ ПО ВЕЛИЧИНЕ."
  20 INPUT "N=";N: DIM X (N)
  30 FOR I=1 TO N: PRINT "X("I;: INPUT ")=";: X (I): NEXT
 100 FOR I=2 TO N
 110 K=I:T=X (K)
 120 FOR J=I-1 TO 1 STEP-1
 130 IF X (J)<=T THEN 160
 140 K=J:X (J+1)=X (J)
 150 NEXT J
 160 X (K)=T
 170 NEXT I
1000 PRINT "НЕУБЫВАЮЩАЯ ПОСЛЕДОВАТЕЛЬНОСТЬ ЧИСЕЛ:"
1010 FOR I=1 TO N: PRINT X (I)" ";: NEXT
1020 END

JRUN
СОРТИРОВКА N ЧИСЕЛ ПО ВЕЛИЧИНЕ.
N=6
X (1)=3
X (2)=7
X (3)=5
X (4)=3
X (5)=-2
X (6)=0
НЕУБЫВАЮЩАЯ ПОСЛЕДОВАТЕЛЬНОСТЬ ЧИСЕЛ:
-2 0 3 3 5 7
```

8.5. Сумма знакопеременного ряда

Пусть задан набор n чисел $a_1, \ldots, a_n$. Составим суммы из всевозможных произведений этих чисел по одному, по два, по три и т. д.:

$$\Sigma_1=\sum_{i=1}^{n} a_i,\ \Sigma_2=\sum_{i>j} a_ia_j,\ \Sigma_3=\sum_{i>j>k} a_ia_ja_k,\ \ldots . \qquad (1)$$

Тогда

$$S=\Sigma_1-\Sigma_2+\Sigma_3+\ldots+(-1)^{n+1}a_1a_2\ \ldots\ a_n=1-\prod_{i=1}^{n}(1-a_i). \qquad (2)$$

Программа вычисляет сумму S знакопеременного ряда произведений из элементов сочетаний.

В программе используются переменные: N — количество чисел n; $A(N)$ — массив, содержащий числа a_i; S — сумма и рабочая переменная; I — переменная цикла.

Программа получена переводом на язык Бейсик Алгол-программы 156б [7].

В контрольном примере найдено, что $S=7$ для $a_1=2$, $a_2=3$, $a_3=4$ ($n=3$).

```
JLIST
10 PRINT "СУММА ЗНАКОПЕРЕМЕННОГО РЯДА..."
20 INPUT "N=";N: DIM A (N)
30 FOR I=1 TO N: PRINT "A ("I;: INPUT ") =";A (I): NEXT
100 S=1
110 FOR I=1 TO N:S=S * (1-A (I)): NEXT
120 S=1-S
1000 PRINT "СУММА S="S
1010 END

JRUN
СУММА ЗНАКОПЕРЕМЕННОГО РЯДА...
N=3
A (1) =2
A (2) =3
A (3) =4
СУММА S=7
```

8.6. День недели по дате

Программа определяет день недели по известной дате (по современному европейскому календарю).

Для вычислений надо ввести через запятую день, месяц и год (соответственно C, B, A). Переменные $A1$, $B1$ хранят исходный месяц и год, N — номер дня недели. Структура программы понятна из текста листинга.

В контрольном примере найдено, что 31 декабря 1987 г.— четверг, а 1 января 1990 г.— понедельник.

```
JLIST
10 PRINT "ДЕНЬ, МЕСЯЦ, ГОД (НАПРИМЕР 12, 8, 1988)"
20 INPUT "(Д, М, Г) ="; C, B, A
100 A1=A: B1=B: IF B>2 THEN B=B+1: GOTO 120
```

```
110 B=B+13:A=A-1
120 N=INT (365.25*A)+INT (30.6*B)+C-621050
130 N=N-INT (N/7)*7+1
140 PRINT C;"." B1"." A1"-";
150 ON N GOTO 1000, 1010, 1020, 1030, 1040, 1050, 1060
1000 PRINT "ПН": END
1010 PRINT "ВТ": END
1020 PRINT "СР": END
1030 PRINT "ЧТ": END
1040 PRINT "ПТ": END
1050 PRINT "СБ": END
1060 PRINT "ВС": END
```

```
JRUN
ДЕНЬ, МЕСЯЦ, ГОД (НАПРИМЕР 12, 8, 1988)
(Д, М, Г)=31, 12, 1987
31.12.1987 — ЧТ
JRUN
ДЕНЬ, МЕСЯЦ, ГОД (НАПРИМЕР 12, 8, 1988)
(Д, М, Г)=1, 1, 1990
1.1.1990 — ПН
```

8.7. Количество дней между двумя датами

Программа определяет количество дней между двумя датами, вычисляя вначале количество дней от начала летоисчисления по григорианскому календарю.

В программе используются переменные: *C*, *B*, *A* — день, месяц и год начальной (затем конечной) даты; *N* — количество дней от начала летоисчисления и на выходе количество дней между датами; *N*1 — количество дней от начала летоисчисления для первой даты.

```
JLIST
 10 PRINT "КОЛИЧЕСТВО ДНЕЙ МЕЖДУ ДВУМЯ ДАТАМИ."
 20 INPUT "ДЕНЬ, МЕСЯЦ, ГОД ПЕРВОЙ ДАТЫ="; D, M, Y
 30 GOSUB 500: N1=N
 40 INPUT  "ДЕНЬ, МЕСЯЦ, ГОД ВТОРОЙ ДАТЫ="; D, M, Y
 50 GOSUB 500:N=ABS (N1-N)
 60 PRINT "N="N: END
500 IF M>2 THEN M=M+1: GOTO 520
510 M=M+13:Y=Y-1
520 N=INT (365.25*Y)+INT (30.6*M)+D: RETURN
530 END
```

```
JRUN
КОЛИЧЕСТВО ДНЕЙ МЕЖДУ ДВУМЯ ДАТАМИ.
ДЕНЬ, МЕСЯЦ, ГОД ПЕРВОЙ ДАТЫ=3, 6, 1987
ДЕНЬ, МЕСЯЦ, ГОД ВТОРОЙ ДАТЫ=19, 8, 1965
N=7958
```

Приведенные в книге программы были реализованы в системе Бейсик-Агат. Для справок приведем здесь основные конструкции этой системы.

1. Символы арифметических операций, отношений и разделителей языка Бейсик-Агат

а) ^, *, /, +, — — соответственно возведение в степень, умножение, деление, сложение и вычитание. Приведены в порядке приоритета выполнения.

б) =, <, <=, >, >=, < > — соответственно равно (=), меньше (<), меньше или равно (≤), больше (>), больше или равно (≥), не равно (≠). NOT, AND, OR — логические НЕ, И, ИЛИ.

в) : (двоеточие), , (запятая), ; (точка с запятой).

Двоеточие разделяет операторы в одной строке. Запятая задает печать по зонам в операторе PRINT, разделяет данные в операторе DATA, разделяет переменные в списках переменных (в операторах INPUT, READ, DIM).

Точка с запятой задает печать вплотную в операторе PRINT.

"..." — строковые переменные заключаются в кавычки.

2. Типы переменных языка Бейсик-Агат

AB — вещественная переменная, AB% — целая переменная, AB $ — текстовая переменная (0 — 255 символов). Причем A — произвольная буква, B — последовательность букв или цифр.

3. Математические функции интерпретатора языка

SIN (X) — функция синус, x в радианах;
COS (X) — функция косинус, x в радианах;

TAN (X) — функция тангенс, x в радианах;
ATN (X) — значение арктангенса в радианах;
INT (X) — целая часть x;
RND (X) — случайное число в интервале 0,1 ($x \neq 0$);
SGN (X) — сигнум-функция (1 при $x > 0$, 0 при $x = 0$ и -1 при $x < 0$);
ABS (X) — модуль x;
SQR (X) — квадратный корень из x;
EXP (X) — экспонента e^x;
LOG (X) — натуральный логарифм x;
FNL (X) — функция, заданная пользователем.

4. Основные операторы интерпретатора языка Бейсик-Агат

= — оператор присваивания;
INPUT — оператор ввода;
PRINT — оператор печати;
DATA — задание значений, которые могут быть считаны оператором READ;
READ — оператор чтения из блока данных DATA;
DIM — задание массива (DIM (X, Y) — двухмерный массив);
GOTO — безусловный переход;
IF ... THEN ... (IF ... GOTO ...) — условный переход;
FOR ... TO ... STEP ... NEXT — оператор цикла;
GOSUB — обращение к подпрограмме;
RETURN — возврат из подпрограммы;
REM — комментарий (при выполнении игнорируется до конца строки);
STOP — приостанов выполнения программы;
END — конец программы;
GET — ввод одного символа с клавиатуры;
RESTORE — восстановление списка данных с начала;
ON ... GOTO ... — переключатель;
ON ... GOSUB ... — переключатель — вызов подпрограмм;
ON ERR GOTO 1000 — подключение программы обработки ошибок, диагностируемых интерпретатором. Начальный номер строки обработчика 1000. На экране диагностика не выдается;
RESUME — выполняется возврат на оператор, при выполнении которого обнаружена ошибка;
OPEN (CLOSE) — открыть (закрыть) файл;
VTAB X (HTAB X) — устанавливает курсор в строку X (позицию X слева);
TAB (X) — элемент оператора PRINT; то же, что HTAB;
POS (O) — номер позиции от левого края текстового окна;

SPC (X) — сдвиг курсора на X позиций вправо, элемент оператора PRINT;
LEN (A$) — число символов в текстовой переменной A$;
STR$ (X) — текстовое представление числового аргумента X;
VAL (A$) — число, текстовое представление которого расположено в начале A$ (до первого нечислового символа);
CHR$ (X) — значение, символ КОИ-8, код которого равен X;
ASC (A$) — код КОИ-8 первого символа текстовой переменной A$;
LEFT$ (A$, X) (RIGHT$ (A$, X)) — первые (последние) X символов A$;
MID$ (A$, X, Y) — строка из Y символов A$, начиная с X.

Как операторы могут использоваться также директивы языка: RUN, LIST, DEL, INVERSE, NORMAL, FLASH, SPEED, HOME, CLEAR.

5. Служебные операторы (директивы) языка Бейсик-Агат

LOAD — загрузка файла из диска;
SAVE — запись файла на диск;
NEW — уничтожение текущей программы и очистка переменных;
CATALOG — читает перечень файлов на диске;
RUN — выполнить программу;
CONT — продолжает выполнение программы (после STOP);
TRACE (NOTRACE) — включить (выключить) трассировку программы;
LIST — вывод текста всей программы;
LIST 10 — вывод строки 10;
LIST 10, 20 — вывод строк от 10 до 20 включительно;
LIST, 30 — вывод строк до 30 включительно;
DEL 10, 20 — стирание строк от 10 до 20 включительно;
DELETE — уничтожить программу на диске;
HOME — очистка экрана;
CLEAR — обнуление всех переменных;
SPEED=X — задает скорость вывода текста (от 0 до 255);
INVERSE — включение инверсного режима вывода текста;
FLASH — включить мерцающий режим вывода текста;
NORMAL — отменяет INVERSE и FLASH;
RENAME AA, BB — переименовать программу AA на BB (на диске);
LOCK (UNLOCK) — установить (снять) защиту для записи в файл;
INIT AA — инициализация магнитного диска и запись файла AA.

6. Сообщения об ошибках

NO FOR ERROR — программа не содержит соответствующего FOR для NEXT;
SYNTAX ERROR — синтаксическая ошибка;
NO GOSUB ERROR — оператор GOSUB не найден для оператора RETURN;
NO DATA ERROR — прочитаны все операторы DATA, а программа пытается прочитать еще данные;
ILLEGAL VALUE ERROR — параметры, передаваемые оператором, не соответствуют их пределам;
OVER FLOW ERROR — переменная, величина которой $>9.999999999E+37$;
OUT OF MEMORY ERROR — программа слишком большая, содержит очень много циклов, подпрограмм или переменных;
UNDEF STATEMENT ERROR — не существует строки, указанной в ссылке;
SUBSCRIPT ERROR — обращение к переменной массива, которая вне описанной в размерности массива, или в ссылке неправильное число размерности;
REDIM ARRAY ERROR — повторное описание массива;
DIVISION BY ZERO ERROR — деление на ноль;
TYPE ERROR — для оператора был задан неправильный тип данных;
DOUBLE DEF NAME ERROR — повторное описание функции;
LANGUAGE NOT AVAILABLE — программа написана не на том языке;
WRITE PROTECTED — диск защищен на запись;
I/O ERROR — ошибка ввода-вывода на диске;
DISK FULL — все пространство на диске заполнено;
FILE LOCKED — файл защищен на запись;
FILE NOT FOUND — файл не найден.

ЛИТЕРАТУРА

1. Уорт Т. Программирование на языке Бейсик.— М.: Машиностроение, 1981.

2. Кетков Ю. Л. Программирование на Бейсике.— М.: Статистика, 1978.

3. Корн Г., Корн Т. Справочник по математике.— М.: Наука, 1984.

4. Библиотека алгоритмов. 1б — 50б: Справочное пособие.— М.: Советское радио, 1975.— Вып. 1.

5. Агеев М. И., Алик В. П., Марков Ю. И. Библиотека алгоритмов. 51б — 100б: Справочное пособие.— М.: Советское радио, 1976.— Вып. 2.

6. Агеев М. И., Алик В. П., Марков Ю. И. Библиотека алгоритмов. 101б — 150б.— М.: Советское радио, 1978.— Вып. 3.

7. Библиотека алгоритмов 151б — 200б: Справочное пособие.— М.: Радио и связь, 1981.— Вып. 4.

8. Агеев М. И., Марков Ю. И., Швакова Г. М. Алгоритмы (201—250).— М.: ИПУ ВЦ АН СССР, 1971.

9. Демидович Б. П., Марон И. А. Основы вычислительной математики.— М.: Физматгиз, 1963.

10. Фильчаков П. Ф. Численные и графические методы прикладной математики: Справочник.— Киев: Наукова думка, 1970.

11. Иванов В. В. Методы вычислений на ЭВМ: Справочное пособие.— Киев: Наукова думка, 1986.

12. Демидович Б. П., Марон И. А., Шувалова Э. З. Численные методы анализа.— М.: Наука, 1967.

13. Толстов Г. П. Ряды Фурье.— М.: Физматгиз, 1960.

14. Лаврентьев М. А., Шабат Б. В. Методы функций комплексного переменного.— М.: Наука, 1965.

15. Гальперин Г. А., Корлюков А. В. Бинарный алгоритм // Квант.— 1986.— № 12.

16. Ляпин Е. С., Евсеев А. Е. Алгебра и теория чисел.— М.: Просвещение, 1974.

17. Бронштейн И. Н., Семендлев К. А. Справочник по математике.— М.: Наука, 1986.

18. Справочник по специальным функциям / Под ред. М. Абрамовича, И. Стиган.— М.: Наука, 1979.

19. Янке Е., Эмде Ф., Леш Ф. Специальные функции.— М.: Наука, 1968.

20. Верлань А. Ф., Сизиков В. С. Методы решения интегральных уравнений с программами для ЭВМ.— Киев: Наукова думка, 1978.

21. Уилкинсон, Райнш. Справочник алгоритмов на языке АЛГОЛ: Линейная алгебра.— М.: Машиностроение, 1976.

22. Трэктон К. Программы на Бейсике для инженерно-технических расчетов.— М.: Радио и связь, 1985.

СОДЕРЖАНИЕ

Введение 3

Глава I. **Теория чисел и линейная алгебра** 8

Теория чисел . . . . . —

1. 1. Алгоритм Евклида для нахождения наибольшего общего делителя двух целых чисел . . . . . —
1. 2. Наименьшее общее кратное двух целых чисел 10
1. 3. Решето Эратосфена для нахождения простых чисел 11
1. 4. Разложение числа на простые множители 12
1. 5. Нахождение числа сочетаний C_n^m 13
1. 6. Коэффициенты полинома $(1+x)^n$ 14
1. 7. Диофантово уравнение $ax+by=c$ . . . . . —

Линейная алгебра 16

1. 8. Обращение матрицы с помощью расширенной матрицы . . . —
1. 9. Вычисление определителя методом триангуляции 18
1.10. Решение системы линейных алгебраических уравнений методом Гаусса 20
1.11. Обращение матрицы методом Гаусса 22

Глава II. **Уравнения. Интерполяция. Ряды. Экстремумы** 25

Полиномиальные и трансцендентные уравнения . . . . . —

2. 1. Корень уравнения $x=F(x)$ . . . . . —
2. 2. Корень уравнения $x=F(x)$ — модифицированный метод итераций 26
2. 3. Корень уравнения — метод половинного деления 27
2. 4. Решение уравнения методом Ньютона (метод касательных) . . 28
2. 5. Нахождение корней уравнения модифицированным методом Ньютона (метод хорд) 29
2. 6. Решение уравнения методом секущих-хорд 30
2. 7. Решение системы уравнений 31
2. 8. Разложение полинома на рациональные линейные множители 32

Интерполяция функций 35

2. 9. Полином Лагранжа по Эйтекену . . . . . —
2.10. Рациональная интерполяция с помощью непрерывных дробей 36
2.11. Интерполяция по Ньютону 38

Операции над полиномами и степенными рядами 39

2.12. Умножение рядов . . . . . —
2.13. Деление рядов 41
2.14. Возведение ряда в степень 42
2.15. Обращение ряда 44
2.16. Коэффициенты полинома при линейном преобразовании аргумента 45

Суммирование и вычисление коэффициентов ряда 46
2.17. Сумма ряда Фурье . . . . . —
2.18. Коэффициенты тригонометрического полинома 48
2. 19. Сумма ряда по Эйлеру 50
Экстремумы функций 51
2.20. Метод наискорейшего спуска . . . . . —
2.21. Минимизация функции многих переменных методом конфигураций 53

Глава III. **Комплексный анализ** 56
3. 1. Умножение комплексных чисел . . . . . —
3. 2. Деление комплексных чисел . . . . . —
3. 3. Корни n-й степени комплексного числа 57
3. 4. Действительная степень комплексного числа 59
3. 5. Комплексная степень комплексного числа 60

Глава IV. **Интегралы** 61
4. 1. Интегрирование методом Симпсона с оценкой точности . . . —
4. 2. Вычисление интеграла методом Симпсона от функции, заданной таблично 62
4. 3. Вычисление интеграла методом Ромберга 63
4. 4. Вычисление криволинейного интеграла в комплексной области 64
4. 5. Вычисление интеграла методом Гаусса 66

Глава V. **Обыкновенные дифференциальные уравнения** 68
5. 1. Модифицированные методы Эйлера решения уравнения первого порядка . . . . . —
5. 2. Метод Рунге-Кутта четвертого порядка для решения уравнения первого порядка 69
5. 3. Обыкновенные дифференциальные уравнения высших порядков и системы дифференциальных уравнений 70
5. 4. Метод Рунге-Кутта с автоматическим выбором шага . . . 72

Глава VI. **Интегральные уравнения** 75
6. 1. Линейное уравнение Вольтерра второго рода 76
6. 2. Уравнение Вольтерра первого рода 78
6. 3. Уравнение Фредгольма второго рода 80

Глава VII. **Специальные функции** 83
7. 1. Гамма-функция и связанные с ней функции . . . . . —
7. 2. Некоторые интегральные функции 90
7. 3. Ортогональные полиномы 95
7. 4. Эллиптические интегралы 104
7. 5. Функции Бесселя целого порядка 107
7. 6. Модифицированные функции Бесселя 115
7. 7. Сферические и модифицированные сферические функции Бесселя. Функции Эйри 123
7. 8. Интеграл вероятностей и интегралы Френеля 129
7. 9. Гипергеометрические функции 132
7.10. Функция распределения вероятностей 135
7.11. Статистические расчеты 138

Глава VIII. **Разные алгоритмы и программы** 145
8. 1. Площадь многоугольника . . . . . —
8. 2. Положение точки относительно n-угольника 147
8. 3. Кратчайший путь 148
8. 4. Сортировка по величине 150
8. 5. Сумма знакопеременного ряда . . . . . —
8. 6. День недели по дате 151
8. 7. Количество дней между двумя датами 152

Приложение 153
Литература 157

Учебное издание

Гринчишин Ярослав Тадеевич
Ефимов Владимир Иванович
Ломакович Афанасий Николаевич

АЛГОРИТМЫ И ПРОГРАММЫ НА БЕЙСИКЕ

Зав. редакцией *Р. А. Хабиб*
Редактор *Н. А. Песина*
Младшие редакторы *Е. А. Сафронова, Е. В. Коркина*
Художественный редактор *Е. Р. Дашук*
Технический редактор *Г. Е. Петровская*
Корректор *М. Ю. Сергеева*

ИБ № 11956

Сдано в набор 29.02.88. Подписано к печати 26.10.88. Формат $60\times90^1/_{16}$. Бум. кн.-журн. офсет. отеч. Гарнитура литературная. Печать офсетная. Усл. печ. л. 10,0. Усл. кр.-отт. 10,25. Уч.-изд. л. 7,91. Тираж 112000 экз. Заказ № 324. Цена 25 к.

Ордена Трудового Красного Знамени издательство «Просвещение» Государственного комитета РСФСР по делам издательств, полиграфии и книжной торговли. 129846, Москва, 3-й проезд Марьиной рощи, 41.

Саратовский ордена Трудового Красного Знамени полиграфический комбинат Росглавполиграфпрома Государственного комитета РСФСР по делам издательств, полиграфии и книжной торговли. 410004, Саратов, ул. Чернышевского, 59.