Министерство общего и профессионального
образования Российской Федерации

Хабаровский государственный технический университет

М Е Т О Д И Ч Е С К И Е У К А З А Н И Я

и задания к контрольной работе
по дисциплине "Информатика" для студентов
II курса заочного факультета специальности
17.09.00 "Подъемно-транспортные, строительные,
дорожные машины и оборудование"

Хабаровск 1998 г.

УДК

Методические указания и задания к контрольной и курсовой работам по дисциплине "Информатика" для студентов II курса заочного факультета специальности 17.09.00 "Подъемно-транспортные, строительные, дорожные машины и оборудование".

Составил Г.М. Вербицкий. – Хабар. гос. техн. ун-т., 1998. – 40 с.

В методических указаниях приведены план курса "Информатика", основные сведения и элементы алгоритмического языка Фортран, даны варианты контрольной и курсовой работ и методика выполнения контрольной работы.

Указания предназначены для студентов II курса заочного факультета специальности 17.09.00 "Подъемно-транспортные, строительные, дорожные машины и оборудование"

Печатается в соответствии с решением кафедры "Строительные и дорожные машины" и методического совета заочного факультета.

СОДЕРЖАНИЕ

Дисциплина "Информатика" имеет целью научить студентов методике постановки, подготовки и решения задач на современных ЭВМ.

Дисциплина предусматривает решене задач общего характера и специально подобранных из практики инженерных расчетов и проектирования элементов строительных и дорожных машин, их технической эксплуатации и ремонта.

1. Содержание дисциплины

Перечень основных тем и вопросов, изучаемых в дисциплине, приведен ниже. После каждой темы указаны литературные источники и номера глав (разделов) и параграфов, где можно найти ответы на вопросы темы.

Тема 1. Характеристика средств вычислительной техники.

1. История и тенденции развития средств вычислительной техники.
2. Структура цифровой ЭВМ. Функциональные возможности ЭВМ и ее подсистем. Особенности современных ЭВМ.
3. Представление информации в ЭВМ. Программы для ЭВМ.
4. Появление и характеристики персональных компьютеров.
5. Персональные компьютеры IBM PC. Принцип открытой архитектуры. Развитие компьютеров IBM PC.

Литература

1. Фигурнов В.Э. IBM PC для пользователя. Часть 1 глава 1 с.11-22. – М.: ИНФРА – М, 1995.

Тема 2. Основы алгоритмизации вычислительных процессов

6. Понятие алгоритма. Основные свойства алгоритмов.
7. Способы описания алгоритмов: словесная запись, операторная запись, описание в виде блок-схем.

8. Правила составления блок-схем алгоритмов. Назначение блоков. Связи между ними.
9. Пример построения блок-схемы алгоритма вычислительного процесса.

Литература

1. Коричнев Л.П., Чистякова В.И. Фортран: Учеб. пособие для сред. спец. учеб. заведений и инж.–техн. работников. – М.: Высш. шк., 1989. Глава 1 п.1.1 с. 8-13.

Тема 3. Основные элементы алгоритмического языка Фортран

10. Алфавит языка Фортран.
11. Константы языка Фортран.
12. Переменные языка Фортран.

13. Функции языка Фортран.
14. Арифметические выражения. Примеры записи на языке Фортран.
15. Выбор имен и типов переменных.

Литература

1. Бухтияров А.М., Маликова Ю.П., Фролов Г.Д. Практикум по программированию на Фортране. Учеб. пособие для вузов. – М.: Наука. Гл. ред.физ.–мат. лит., 1988. Глава 1 п.1.2 – 1.8 с. 11-34.
2. Фортран 77 ЕС ЭВМ / З.С. Брич, О.Н. Гулецкая, Д.В. Капилевич и др. – М.: Финансы и статистика, 1989. Глава 1 п.1.1 1.2 с. 5-7, Глава 2 п. 2.1 – 2.3 с. 14-19, п. 2.6 с. 22-24.
3. Коричнев Л.П., Чистякова В.И. Фортран: Учеб. пособие для сред. спец. учеб.заведений и инж.–техн. работников. – М.: Высш. шк., 1989. Глава 2 п. 2.6 с. 26-38.

Тема 4. Программирование линейных алгоритмов

16. Понятие о линейных алгоритмах. Схема алгоритма.
17. Программа. Структура программы. Запись программы на Фортране на бланке.
18. Арифметический оператор присваивания.
19. Присвоение переменным начальных значений.
20. Печать результатов.
21. Пример программы линейного вычислительного процесса.

Литература

1. Коричнев Л.П., Чистякова В.И. Фортран: Учеб. пособие для сред. спец. учеб.заведений и инж.–техн. работников. – М.: Высш. шк., 1989. Глава 3 п.
,

3.1– 3.6.
2. Бухтияров А.М., Маликов Ю.П., Фролов Г.Д. Практикум по программированию на Фортране. Учеб. пособие для вузов. – М.: Наука. Гл. ред.физ.–мат. лит., 1988, Глава 2 п.2.1, 2.2, с. 40-44.

Тема 5. Программирование разветвляющихся алгоритмов

22. Понятие о разветвляющихся алгоритмах.
23. Операторы безусловного перехода.
24. Логические выражения.
25. Операторы условного перехода арифметические и логические.
26. Правила программирования разветвлений.

Литература

1. Коричнев Л.П., Чистякова В.И. Фортран: Учеб. пособие для сред. спец. учеб. заведений и инж.–техн. работников. – М.: Высш. шк., 1989. Глава 1, п. 1.3

с. 14-15, Глава 3 п.5.1–5.4 с. 73-86.
2. Бухтияров А.М., Маликов Ю.П., Фролов Г.Д. Практикум по программированию на Фортране. Учеб. пособие для вузов. – М.: Наука. Гл. ред.физ.–мат. лит., 1988. Глава 2 п.2.3, 2.4 с. 44-51, Глава 1 п.1.9,1.10 с. 34-40.
3. Фортран 77 ЕС ЭВМ / З.С. Брич, О.Н. Гулецкая, Д.В. Капилевич и др. – М.: Финансы и статистика, 1989. Глава 2 п. 2.6.3–2.6.5 с. 24-27.

Тема 6. Программирование циклических алгоритмов

27. Понятие о циклических алгоритмах с заданным числом повторений и с заранее неизвестным числом повторений.
28. Алгоритмы с вложенными циклами.
29. Оператор DO. Программирование циклов с помощью оператора DO.
30. Вычисление конечных сумм и произведений.
31. Программирование итерационных циклов.
32. Программирование вложенных циклов.

Литература

1. Коричнев Л.П., Чистякова В.И. Фортран: Учеб. пособие для сред. спец. учеб
заведений и инж.–техн. работников. – М.: Высш. шк., 1989. Глава 1, п. 1.4, 1.5 с. 15-23, Глава 6 п.6.1–6.6 с. 89-98.
2. Фортран 77 ЕС ЭВМ / З.С. Брич, О.Н. Гулецкая, Д.В. Капилевич и др. – М.: Финансы и статистика, 1989. Глава 3, п. 3.2.4, 3.2.5 с. 40-48.
3. Боглаев Ю.П. Вычислительная математика и программирование: Учеб. пособие для студентов вузов. – М.: Высш. шк., 1990. Глава 3 п.3.2–8 с. 77-79.
4. Самохин А.Б., Самохина А.С. Фортран и вычислительные методы. – М.:

Русина, 1994. п.5 с. 27-30.

Тема 7. Массивы

33. Основные сведения о массивах. Описание массивов.
34. Присвоение начальных значений элементами массивов.
35. Ввод-вывод значений элементов массивов. Ввод-вывод отдельных элементов массивов. Ввод-вывод значений элементов массивов по имени массива. Ввод-вывод значений элементов массивов с помощью неявной формы DO.
36. Программирование нахождения минимального или максимального значения элемента массива и его расположения в массиве.
37. Программирование определения сумм положительных и отрицательных элементов массива и их количества.

Литература

1. Самохин А.Б., Самохина А.С. Фортран и вычислительные методы. – М.:

Русина, 1994. п.5 с.30-33.
2. Коричнев Л.П., Чистякова В.И. Фортран: Учеб. пособие для сред. спец. учеб. заведений и инж.–техн. работников. – М.: Высш. шк., 1989. Глава 7 п.7.1–7.5 с. 101-116.

Тема 8. Организация ввода-вывода данных

38. Операторы ввода-вывода.
39. Оператор FORMAT. Спецификации преобразования.
40. Оператор FORMAT. Форматы редакции. Разделители форматов. Повторители форматов. Группы форматов.
41. Взаимодействие оператора FORMAT со списком ввода-вывода. Преобразование данных при вводе. Выбор форматов ввода. Запись исходных данных на бланке.
42. Преобразование данных при выводе. Выбор форматов вывода.

Литература

1. Боглаев Ю.П. Вычислительная математика и программирование: Учеб. пособие для студентов втузов. – М.: Высш. шк., 1990. п. 3.2.7 с. 71-76.
2. Самохин А.В., Самохина А.С. Фортран и вычислительные методы. – М.: Русина, 1994. п. 6 с. 33-41.
3. Коричнев Л.П., Чистякова В.И. Фортран: Учеб. пособие для сред. учеб. заведений и инж.–техн. работников. – М.: Высш. шк., 1989. Глава 4 п. 4.1–4.5 с. 56-70

Тема 9. Подпрограммы

43. Использование подпрограмм при программировании на языке Фортран. Оператор-функция. Обращение к оператору-функции.
44. Подпрограмма FUNCTION. Структура подпрограммы FUNCTION. Оператор FUNCTION. Обращение к подпрограмме FUNCTION.
45. Подпрограмма SUBROUTINE. Структура подпрограммы SUBROUTINE. Оператор SUBROUTINE. Выполнение подпрограммы.
46. Правила установления соответствия между формальными и фактическими аргументами при обращении к подпрограммам.
47. Использование общих областей памяти для передачи данных между подпрограммами.

Литература

1. Боглаев Ю.П. Вычислительная математика и программирование: Учеб. пособие для студентов втузов. – М.: Высш. шк., 1990. п. 3.2.10 с. 82-87.
2. Самохин А.Б., Самохина А.С. Фортран и вычислительные методы. – М.: Русина, 1994. п. 7 с. 42-46.
3. Фортран 77 ЕС ЭВМ / З.С. Брич, О.Н. Гулецкая, Д.В. Капилевич и др. – М.: Финансы и статистика, 1989, с. 54-57.

4. Коричнев Л.П., Чистякова В.И. Фортран: Учеб. пособие для сред. спец. учеб. заведений и инж.–техн. работников. – М.: Высш. шк., 1989. Глава 8 п. 8.1–8.6 с. 118-131.

Тема 10. Организация внешних файлов данных

48. Работа с внешними файлами. Файлы последовательного доступа. Файлы прямого доступа.
49. Операторы для работы с файлами. Оператор OPEN. Оператор CLOSE.
50. Операторы передачи данных.
51. Операторы установки текущей позиции в файле.

Литература

1. Фортран 77 ЕС ЭВМ / З.С. Брич, О.М. Гулецкая, Д.В. Капилевич и др. – М.: Финансы и статистика, 1989. Глава 4 п. 4.4–4.6 с. 127-151.
2. Боглаев Ю.П. Вычислительная математика и программирование: Учеб. пособие для студентов втузов. – М.: Высш шк., 1990. Глава 3 п. 3.5.3, 3.5.4 с. 128-131.

Тема 11. Элементы линейной алгебры

52. Классификация матриц. Операции с матрицами.
53. Решение систем линейных алгебраических уравнений. Метод исключения неизвестных (метод Гаусса): решение способом преобразования уравне-

ний, решение системы линейных уравнений с помощью эквивалентных преобразований матриц системы.

54. Решение системы линейных уравнений по формулам Крамера.

Литература

1. Справочник по математике для научных работников и инженеров. Корн Г., Корн Т. – М.: Наука, 1984. Глава 13 п. 13.2-1, 13.2-2, 13.2-3 с 390-393.

Тема 12. Численные интегрирование и дифференцирование

55. Численное интегрирование методами прямоугольников, трапеций, Симпсона.
56. Простейшие формулы численного дифференцирования разностным методом.
57. Применение интерполяционных многочленов Лагранжа для численного дифференцирования.
58. Применение интерполяционных многочленов Ньютона для численного дифференцирования.

Литература

1. Самохин А.Б., Самохина А.С. Фортран и вычислительные методы. – М.: Русина, 1994. п. 18 с. 89-93
2. Численные методы. Волков Е.А.: Учебное пособие. – М.: Наука, 1982. Глава 1 п. 10 с. 56-66, Глава 2 п.15 с. 105-111.

2. Краткие сведения об алгоритмическом языке Фортран

2.1. Понятие алгоритма

Понятие алгоритма широко используется как в математике, так и в программировании. Алгоритм представляет собой описание вычислительного процесса, ведущего от варьируемых начальных данных к искомому результату.

Правильно разработанный алгоритм должен обладать следующими свойствами:

определенностью или детерминированностью – применение алгоритма к одним и тем же исходным данным должно приводить к одному и тому же результату;

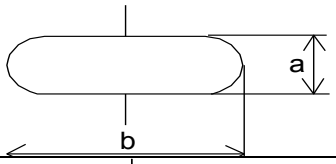
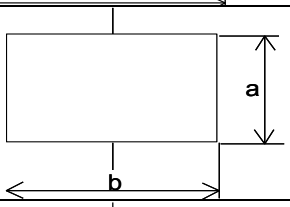
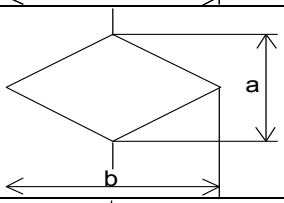
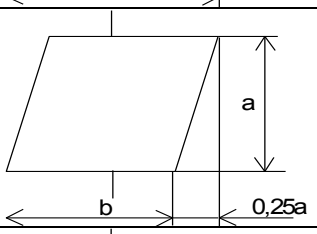
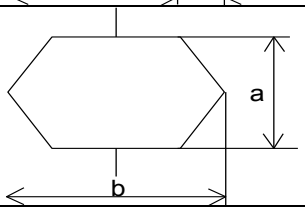
массовостью – алгоритм разрабатывается в общем виде так, чтобы его можно было применить для класса задач, различающихся лишь исходными данными;

результативностью, т.е. возможностью достижения результата за конечное число достаточно простых шагов.

Существуют различные способы описания алгоритмов – словесный, операторный, в виде блок-схемы. Наиболее наглядным является алгоритм в виде блок-схемы, которая содержит последовательность блоков, предписывающих выполнение определенных операций, и связей между ними. В табл. 1 приведены некоторые наиболее часто употребляемые блоки и пояснения выполняемых функций.

Таблица 1

Некоторые блоки, применяемые при составлении блок-схем алгоритмов

Название блока	Обозначение (ГОСТ 19.003 – 80)	Выполняемая функция
Пуск, останов		Начало, конец, останов, вход и выход в отдельно описанных алгоритмах и подпрограммах
Процесс		Вычислительное действие или последовательность действий
Решение		Проверка условия и выбор направления хода вычислительного процесса
Ввод-вывод		Ввод или вывод данных
Модификация		Начало цикла

2.2. Простейшие конструкции языка

К простейшим конструкциям языка относятся константы, переменные, функции и выражения. Для записи простейших конструкций и операторов используются следующие символы:

– прописные и строчные буквы латинского алфавита от A до Z;

– цифры от 0 до 9;

– специальные символы: = - равно; + - плюс; – - минус; * - звездочка; / - дробная черта; (- левая скобка;) – правая скобка; , - запятая; . – точка; ' – апостроф; - пробел (отсутствие символа); $ - знак денежной единицы; & - коммерческое "и".

Буквы русского алфавита и другие символы могут использоваться только в комментариях и текстовых константах фортран-программы.

Константы. В Фортране используются четыре типа констант – целые, вещественные, комплексные, логические.

Целые константы представляют собой последовательность цифр без десятичной точки со знаком (-) или без знака. Например: 92; -7619; 0; 371; +956748

В памяти компьютера (типа IBM PC) целая константа занимает 4 байта памяти и может иметь значение от –2147483648 до 2147483647.

Вещественная константа может быть представлена одним из следующих способов:

– вещественная константа без порядка, т.е. число, записанное с десятичной точкой, например: 74.95; -649.571; 0.0; +4.; 7.93; 0.0041

– вещественная константа с порядком, которая записывается в виде nEm.

Здесь мантисса n – вещественная константа, порядок m представляет собой как однозначное или двузначное целое число со знаком или без знака. Примеры: -0.02341E2 (=$-0.02341 \cdot 10^{2}$); 7.34E+11 (=$7.34 \cdot 10^{11}$); 0.03E-7 (=$0.03 \cdot 10^{-7}$).

Диапазон изменения – от – 8.43E-37 до 3.37E+38, количество цифр в мантиссе – не более семи.

Вещественная константа двойной точности записывается только в виде вещественной константы с порядком. Для указания порядка вместо буквы E используется буква D. Примеры: 7.24D2; -6.194D-14. В памяти компьютера константа занимает 8 байт памяти. Диапазон изменения – от – 10D-64 до +9.99D+62, количество цифр в мантиссе не более 14.

Комплексная константа записывается в виде(X_r, X_i),
где X_r и X_i – вещественные константы; X_r – действительная часть комплексного числа, X_i – мнимая часть. Примеры: (12.35, -1.129); (-3.94, 2.0E-13), которые в обычной математической записи имеют вид 12.35 – 1.129i; $-3.94+2.0 \cdot 10^{-13}$i. В памяти компьютера комплексная константа занимает 8 байт памяти.

Комплексная константа двойной точности состоит из пары вещественных констант двойной точности. Пример: (2.1D0, -3.12D-2). В памяти компьютера константа занимает 16 байт памяти.

Логическая константа может принимать только два значения: .TRUE. или .FALSE.. Точки в записи являются обязательными. В памяти компьютера константа занимает 4 байта.

Переменные – это величины, значения которых изменяются в процессе выполнения программы. Имя переменной состоит из не более чем 8 алфавитно-цифровых символов. Первым символом имени должна быть буква. Если никаких указаний о типе переменной в программе нет, то переменная относится к одному из двух типов (целому, вещественному) по следующему правилу: если имя переменной начинается с букв I, J, K, L, M, N, то данная переменная считается переменной целого типа, а если имя переменной начинается с любой другой буквы латинского алфавита, то она считается переменной вещественного типа. Такое описание типа называется описанием типа по умолчанию.

Описание по умолчанию перестает действовать, если тип переменной задать с помощью операторов явного описания типа: INTEGER <список переменных>; REAL <список переменных>; REAL*8 <список переменных>; COMPLEX <список переменных>; COMPLEX*16 < список переменных>; LOGICAL < список переменных>. Перечисленные операторы указывают следующие типы переменных: первый – целый; второй – вещественный; третий – вещественный двойной точности; четвертый – комплексный; пятый – комплексный двойной точности; шестой – логический.

Примеры: INTEGER X0, Y, NM, CLMPQ
COMPLEX S12, CZ1
LOGICAL LL

Операторы описания являются невыполняемыми операторами, поскольку никаких команд не порождают.

Функции. Для обращения к функции необходимо указать ее имя и в круглых скобках аргументы. Функции бывают целого, вещественного, вещественного двойной точности и комплексного типа. Наиболее часто употребляемые функции вещественного типа приведены в табл. 2.

Таблица 2

Функции вещественного типа

Название функции	Математическое обозначение функции	Запись функции на языке Фортран
Синус	sin x	SIN(X)
Косинус	cos x	COS(X)
Тангенс	tg x	TAN(X)
Арктангенс	arctg x	ATAN(X)
Логарифм натуральный	ln x	ALOG(X)
Логарифм десятичный	lg x	ALOG10(X)
Экспонента	e^x	EXP(X)
Корень квадратный	$\sqrt{x}$	SQRT(X)
Абсолютное значение	| x |	ABS(X)
Целая часть числа	] x [	AINT(X)
Остаток от деления x_1 на x_2	–	AMOD(X1,X2)

Тип аргумента должен соответствовать типу используемой функции. Аргумент прямых тригонометрических функций должен быть в радианах.

Имена большинства функций целого, вещественного двойной точности, комплексного и комплексного двойной точности типов начинаются соответственно с символов I, D, C, CD. Аргументы должны быть тех же типов.

Арифметические выражения – это запись математической формулы с использованием констант, переменных, функций, знаков арифметических операций и круглых скобок. Результатом вычисления значения арифметического выражения является числовая константа. Порядок выполнения операций в выражении задается установленным приоритетом операций: вычисление функций; возведение в степень; умножение и деление; сложение и вычитание. Операции одного приоритета выполняются слева направо, за исключением операции возведения в степень, которая выполняется справа налево.

При написании арифметических выражений во избежание ошибок, а также для удобства прочтения целесообразно использовать скобки.

Пример:

X*Y/(X**2+7.5)+COS(X).

В математической записи это выглядит как

$$\frac{xy}{(x^2 + 7{,}5)} + \cos(x)$$

Величины разных типов (целые, вещественные и т.д.) в памяти компьютера кодируются по разному, и, следовательно, арифметические операции над ними выполняются также по разному.

Результатом операции с целыми величинами будет целая величина. Например, результатом деления двух целых величин будет целая часть полученного частного. Если J=9, I=4, то J/I=2.

Любая вещественная величина может быть возведена в целую степень. Операция возведения вещественных величин в вещественную степень вида A**B выполняется только в том случае, если $A \geq 0$.

Когда в арифметическом выражении все операнды (переменные и константы) одного типа, то величина, являющаяся результатом выполнения этого выражения, того же типа. Когда операнды имеют разный тип, то тип результата определяется типом операнда максимального ранга.

Ранг операнда зависит от его типа в соответствии со списком

1. INTEGER
2. REAL
3. REAL*8
4. COMPLEX
5. COMPLEX*16

Например результатом вычисления арифметического выражения с операндами INTEGER и REAL будет величина, относящаяся к типу REAL.

Особый случай: операции над операндами REAL*8 и COMPLEX породят величину COMPLEX*16, а не COMPLEX.

В любой Фортран-программе должен присутствовать, по крайней мере, один выполняемый оператор.

Логические выражения состоят из отношений, логических переменных, логических констант и логических операций.

Отношение – это простейшее логическое выражение вида

A # B

Здесь A и B – арифметические выражения целого или вещественного типа; # – знак отношения.

Знаки отношения

.LT. меньше <
.LE. меньше или равно ≤
.EQ. равно =
.NE. не равно ≠
.GE. больше или равно ≥
.GT. больше >

Точки в записи знаков отношений обязательны.

Пример: X**2+5*X.GT.Y

В математической записи это выглядит как $x^2 + 5x \geq y$.

Значением отношения может быть либо "истина", т.е. .TRUE., либо "ложь", т.е. .FALSE. В приведенном примере, если X=1, Y=2, то результатом отношения будет .TRUE., а если X=2, Y=20, то результатом будет .FALSE.

Из отношений, логических переменных и констант можно составить логическое выражения вида L1@L2@L3… Здесь L1, L2, …– логические элементы, т.е.

отношения, логические переменные и константы; @–знак логической операции (табл. 3)

Таблица 3

Логические операции

Знак операции	Запись	Значение
.AND.	L1.AND.L2	Выражение истинно, когда L1 и L2 истинны. Во всех других случаях оно ложно.
.OR.	L1.OR.L2	Выражение истинно, если L1 или L2 истинны. Во всех других случаях оно ложно.
.NOT.	.NOT.L	Выражение истинно, когда L ложно, а в противном случае оно ложно.

Точки в записи логических операций обязательны.

Пример: X.LT.Y.AND.Y.LT.Z

Логическое выражение истинно, если Y удовлетворяет условию X<Y<Z и ложно в других случаях.

Два знака логических операций не должны стоять рядом, а, если это необходимо, то они должны быть разделены скобками. Так, например, запись вида L1.AND..NOT.L2 неверна, а запись L1.AND.(.NOT.L2) верна.

При отсутствии скобок в логическом выражении порядок выполнения операций идет в следующей последовательности: 1) вычисление функций; 2) возведение в степень; 3) умножение и деление; 4) сложение и вычитание; 5) операции отношения; 6) операция .NOT.; 7) операция .AND.; 8) операция .OR. Операции одного ранга выполняются последовательно слева направо.

2.3 Основные операторы языка

2.3.1. Структура программы

Любая программа на языке Фортран состоит из операторов и комментариев. С помощью операторов программист описывает действия, которые необходимо выполнить при реализации алгоритма решения задачи. Комментарии служат для облегчения понимания программы, обеспечивают словесное описание логики работы программы.

Операторы и комментарии располагаются в строках длиной в 80 позиций. Положение символа в строке нумеруется слева направо, начиная с 1-ой позиции. Операторы Фортрана могут находиться не ранее чем с 7 и по 72 позиции строки. Любой оператор может быть помечен меткой – целым числом, метка располагается в

позициях 1-5 строки. В строке не должно быть более одного оператора. Если оператор не помещается в позициях 7-72 или желателен его перенос на следующую строку, то в каждой строке продолжения в 6-ой позиции печатается символ "*" либо любой другой символ, отличный от нуля. Каждая строка комментария начинается с буквы С (лат.) в 1-ой позиции. Такие строки транслятором игнорируются. Для записи комментария могут использоваться и русские буквы, а сам текст комментария располагается в позициях 2-80 строки.

Все операторы Фортрана разделяются на выполняемые и невыполняемые. Выполняемые операторы определяют действия, которые должны быть выполнены. К выполняемым операторам относятся операторы присваивания. управления и ввода-вывода. Невыполняемые операторы позволяют описывать свойства данных, к ним относятся: операторы описания типов данных и размерностей массивов, оператор присвоения начальных значений данных (DATA), оператор указания формы преобразования вводимой и выводимой информации (FORMAT) и некоторые другие.

При составлении программы на Фортране операторы должны записываться в такой последовательности: 1) операторы описания; 2) определения операторов-функций; 3) выполняемые операторы; 4) оператор конца текста программы.

Для завершения программы и указания конца её текста используются операторы STOP и END.

2.3.2. Оператор присваивания

Простейшим выполняемым оператором является арифметический оператор присваивания, который имеет следующий вид

V=A

Здесь V – имя переменной, а A – арифметическое выражение.

Правило выполнения: переменной с именем V присваивается значение арифметического выражения A.

Если переменная V – целая, а тип A – вещественный, то V будет присвоено значение целой части результата A.

Примеры: Y=(X**2+5.2)/(X+1)
I=I+1
A=7.5

Все переменные, находящиеся справа от знака равенства перед выполнением оператора присваивания должны быть определены в предыдущей части программы

либо с помощью других операторов присваивания, либо при задании исходных данных с помощью оператора ввода.

2.3.3. Операторы ввода-вывода

Операторы ввода-вывода по своей структуре близки и имеют следующий вид

READ (n,m) <список переменных>
WRITE (n,m) <список переменных>

Здесь первый оператор – оператор ввода, а второй оператор – оператор вывода; n – номер устройства ввода или вывода; m – метка оператора FORMAT, который опре-

деляет порядок расположения данных в строке при вводе или выводе; <список переменных> – вводимые или выводимые переменные.

Простейшая запись этих операторов имеет следующий вид

READ (*,*) <список переменных>

WRITE (*,*) <список переменных>

В этом случае исходные данные вводятся в компьютер путем набора их на экране дисплея, а результаты после выполнения программы также появляются на экране дисплея.

Примеры:

READ (*,*) N,A1,B01,C

WRITE (*,*) Z,FX

2.3.4. Оператор безусловного перехода

Оператор безусловного перехода используется для перехода к оператору, помеченному меткой. Оператор имеет вид

GO TO n

Здесь n – метка оператора. Правило выполнения: следующим выполняется оператор с меткой n.

Пример: GO TO 15

2.3.5. Операторы условного перехода

Используют два вида операторов условного перехода – арифметический и логический.

Условный арифметический оператор имеет следующий вид

IF(A) n_1,n_2,n_3

Здесь A – арифметическое выражение целого или вещественного типа; n_1,n_2,n_3 – метки операторов. Правило выполнения: если значение арифметического выражения меньше нуля, то следующим выполняется оператор с меткой n_1; если значение выражения равно нулю, то выполняется оператор с меткой n_2; если значение выражения больше нуля, то выполняется оператор с меткой n_3. Две метки из трех могут совпадать.

Примеры: IF(X+Y-7.5) 5,16,3

IF(I – N) 2,2,1

Условный логический оператор имеет вид

IF (L) S

Здесь L – логическое выражение, а S – оператор. Правило выполнения: если логическое выражение L истинно, то выполняется оператор S, а если оно ложно, то выполняется следующий за IF оператор, а S не выполняется.

Примеры: IF(I.LE.N) GO TO 10

IF(X**2+Y**2.LE.R**2) A=1

Структурный оператор IF имеет вид

IF(L) THEN

S1

S2

…
END IF

Здесь L – логическое выражение; S1,S2,… – операторы, которых может быть любое количество. Правило выполнения: если логическое выражение L истинно, то выполняются операторы S1,S2,…, а если оно ложно, то выполняется следующий после END IF оператор.

Пример:
```
IF(L) THEN
F=SIN(X)
F1=X**2+5*X
END IF
```

Блочный оператор IF (оператор IF THEN ELSE) имеет вид

```
IF(L) THEN
<операторы 1>
ELSE
<операторы 2>
END IF
```

Правило выполнения: если логическое выражение L истинно, то выполняется <операторы 1>, а если оно ложно, то выполняются <операторы 2>.

2.3.6. Операторы цикла

Операторы цикла предназначены для многократного повторения вычислений по одним и тем же зависимостям при различных значениях входящих в них переменных. Такой вычислительный процесс называется циклическим, многократно повторяющийся участок этого процесса – телом цикла.

Оператор цикла DO имеет вид

```
Do n  i=m1,m2,m3
```

Здесь n – метка последнего оператора тела цикла (или оператора CONTINUE – пустого оператора); i – переменная целого (вещественного или вещественного двойной точности – по версии 5.0 Фортрана) типа, которая называется параметром цикла; m1, m2, m3 – константы, переменные или арифметические выражения целого (вещественного или вещественного двойной точности) типа. Величина m1 определяет начальное значение параметра цикла, m2 – конечное значение, а m3 – шаг или приращение.

Оператор DO автоматически исполняет все операторы по оператор с меткой n (или оператор CONTINUE с меткой n) для значений i от m1 по m2 с шагом m3. Если в записи цикла значение шага m3 отсутствует, то оно принимается равным единице.

После окончания цикла выполняется следующий за оператором с меткой n (оператором CONTINUE с меткой n) оператор.

Пример: пусть задана функция f(x) в виде ряда по степеням основания x

$$f(x) = x + \frac{x^2}{2} + \frac{x^3}{3} + \ldots + \frac{x^n}{n},$$

где n – параметр.

Программа, которая для любых X и N вычисляет соответствующее значение функции F может иметь следующий вид

```
      READ (*,*) X,N
      F=0.
      DO 5 I=1,N
      F=F+X**I/I
    5 CONTINUE
      WRITE (*,*) F
      STOP
      END
```

В качестве оператора, завершающего тело цикла, можно использовать оператор END DO. В этом случае в заголовке цикла не указывается метка, а вместо оператора CONTINUE записывается END DO. Приведенная программа для этой формы записи будет иметь следующий вид

```
      …
      DO I=1,N
      …
      END DO
      …
```

Оператор цикла DO WHILE имеет вид

```
      DO n WHILE (L)
```

Здесь n – метка последнего оператора цикла (оператора CONTINUE); L – логическое выражение.

Правило выполнения: вычисляется логическое выражение L. Если оно принимает значение .TRUE., то выполняются все операторы тела цикла до оператора CONTINUE с меткой n. Этот процесс продолжается до тех пор, пока логическое выражение не примет значение .FALSE.. Тогда ни один оператор тела цикла не исполняется и выполняется оператор, следующий за оператором с меткой n.

Примеры заголовка цикла

```
      DO 25 WHILE (X**2+Y**2.LE.R**2)
      DO 10 WHILE (Y.GE.A.OR.X.GE.B)
```

Пример использования оператора DO WHILE. Пусть задана функция f(x) в виде ряда по степеням основания x.

$$f(x) = 1 - x + \frac{x^2}{2} - \frac{x^3}{3} + \ldots + (-1)^n \frac{x^n}{n} + \ldots$$

Суммирование членов ряда в программе должно проводиться до тех пор, пока модуль члена ряда не станет меньше заданной величины E. Программа может иметь следующий вид

```
      READ (*,*) X,E
```



```
      F=1.0
      A=-X
      I=1
      DO 10 WHILE (ABS(A).GE.E)
```

```
      F=F+A
      I=I+1
      A=-X*A/I
   10 CONTINUE
      WRITE (*,*) F
      STOP
      END
```

В качестве оператора, завершающего тело цикла, можно использовать оператор END DO. В этом случае в заголовке цикла не указывается метка. Последняя программа для этой формы записи цикла будет иметь вид

```
      …
      DO WHILE (ABS(A).GE.E)
      …
      END DO
      …
```

При использовании операторов цикла необходимо придерживаться следующих правил:

– если используется оператор цикла внутри другого цикла, то их тела не должны пересекаться, т.е. тело внутреннего цикла полностью должно находиться внутри тела внешнего цикла;

– можно входить в цикл только через оператор DO;

– два или более цикла DO или DO WHILE могут заканчиваться одним оператором (CONTINUE). Однако оператор END DO может быть концом только одного цикла;

– при выходе из цикла сохраняется текущее значение параметра цикла I для цикла DO или значение логического выражения L для цикла DO WHILE.

2.3.7 Массивы

Массивом называется совокупность элементов одного типа, обозначаемая именем с индексами в круглых скобках. Имена массивов образуются по тем же правилам, что и имена простых переменных. Например, вектор с компонентами u_1, u_2, …, u_n можно представить одномерным массивом

U(I), $1 \le I \le N$,

а матрицу с компонентами A_{ij} можно представить двумерным массивом

A(I,J), $1 \le I \le N$, $1 \le J \le M$

Количество индексов в массиве определяет его мерность, и массивы могут быть, соответственно, одномерными, двумерными, трехмерными и т.д.

Каждый массив должен быть описан в начале программы с помощью оператора размерности DIMENSION с указанием предельных значений каждого индекса, которые задаются целыми константами. Это необходимо для того, чтобы зарезервировать соответствующий объем памяти для хранения элементов массива.

Пример:


```
      DIMENSION A(10,30),BK(500),IU(40)
```

Здесь первый массив – двумерный. Если представить его в виде матрицы, то количество строк – 10, в каждой строке – по 30 элементов.

Описание размерности массива может быть совмещено с описанием типа переменных. Поэтому запись

```
REAL KLM, XF
INTEGER FR, SS
DIMENSION KLM(15,15), FR(100), XF(10), SS(5,5)
```

эквивалентна записи

```
INTEGER FR(100), SS(5,5)
REAL KLM(15,15), XF(10)
```

Описание массива в программе может быть произведено только один раз.

Массив размещается в последовательно расположенных ячейках памяти. Если массив одномерный, то его элементы хранятся в памяти друг за другом, например A(1), A(2), A(3), A(4), …. Если массив двумерный, например,

```
DIMENSION B(3,4),
```

то его элементы хранятся в памяти в таком порядке B(1,1), B(2,1), B(3,1), B(1,2), B(2,2), B(3,2), B(1,3), B(2,3), B(3,3), B(1,4), B(2,4), B(3,4). Тот же принцип сохраняется и для массива большей размерности: быстрее всего изменяется первый индекс, затем второй и т.д.

При обращении к элементу массива необходимо указывать его имя и индексы, которые могут быть либо константами, либо арифметическими выражениями целого типа, например

```
V=A(I,J)+B(3,2*L+7)
```

В этом случае целые переменные I, J, L должны быть заданы в предыдущей части программы.

При вводе и выводе массивов используются два основных способа – ввод-вывод массива по имени или с помощью цикла.

Первый способ иллюстрируется примером:

```
DIMENSION A(10),B(3,4)
READ (*,*) A,B
```

Для второго способа пример следующий:

```
DIMENSION A(10), B(3,4)
READ (*,*) N,L,M,(A(I),I=1,N),((B(I,J),J=1,L),I=1,M)
```

В этом случае числа N, M, L подчиняются ограничениям $N \le 10$, $L \le 4$, $M \le 3$. Этот способ удобнее, когда в программе используется меньшее количество элементов массивов, нежели то, которое задано в операторах размерности. В этом примере сначала вводятся три целых числа N, L, M, и затем последовательно N элементов массива A, а затем L*M элементов массива B в последовательности

```
B(1,1), B(1,2), … , B(2,1), B(2,2), …
```

В этом случае матрица B вводится построчно.

Вывод массивов отличается от их ввода только заменой оператора READ на оператор WRITE.

Пример: Найти сумму элементов одномерного массива с количеством элементов $N \le 100$. Программа может иметь следующий вид:

```
DIMENSION A(100)
READ(*,*) N,(A(I),I=1,N)
```



```
S=0.0
DO 5 I=1,N
```

```
      S=S+A(I)
    5 CONTINUE
      WRITE(*,*) S
      STOP
      END
```

2.3.8. Ввод-вывод на Фортране

Операторы ввода-вывода по своей структуре близки и имеют следующий вид

READ (n,m) <список переменных>

WRITE (n,m) <список переменных>

Здесь n – номер канала ввода или вывода; m – метка оператора FORMAT, который определяет порядок расположения данных в строке при вводе или выводе; <список переменных> – имена вводимых или выводимых переменных, разделенных запятыми.

Для ввода данных с клавиатуры в качестве номера канала используется n=5 (наряду с символом "*" – см. выше), а для вывода результатов на принтер – n=6. Вывод результатов или информации на экран может быть осуществлен при n=5 (наряду с символом "*").

Оператор FORMAT обеспечивает форматный ввод-вывод. В этом случае в явном виде указывают местоположение и форму вводимых или выводимых переменных в записи (строке). Общий вид оператора

m FORMAT (S),

где m – обязательная метка оператора, а S – список спецификаций формата, разделенных запятыми.

Для каждого типа данных существует своя спецификация: для целых величин – I, для вещественных – F или E, для вещественных двойной точности – D, для логических величин – L.

Общая форма спецификации I

Iw,

где w – беззнаковая целая константа, показывающая количество символов (длину поля), отводимых для ввода или вывода целых величин. Например, в спецификации I5 целые величины -5712 и 359 будут представлены

-5712 359

а в спецификации I4 первая величина не может быть представлена, а вторая будет иметь вид 359. Поэтому при задании спецификации I необходимо учитывать примерный порядок вводимых или выводимых величин.

Если число меньше длины поля w, то при вводе и выводе левая часть поля заполняется пробелами.

Если при вводе число занимает больше w символов, то считываются только w левых символов. Например, запись –1352764 по спецификации I5 будет считана как –1352.

Если при выводе число занимает больше w символов, то в отводимом для записи числа поле будут напечатаны w звездочек "*".

Общая форма спецификации F

Fw.d,

где w – общая длина поля, включая знак числа и десятичную точку, а d – число цифр после десятичной точки.

Во всех случаях w > d для положительных чисел и w > d+1 – для отрицательных.

Например, в спецификации F7.3 числа –72.591 и 24.52 будут представлены

-72.591 24.520

Если при выводе целая часть числа (для отрицательного числа – со знаком) больше w–(d+1) символов, то в отводимом для записи числа поле будут напечатаны w звездочек "*".

Вещественные числа с порядком представляются с помощью спецификации E, общий вид которой

Ew.d,

где w – общая длина поля; d – число цифр после десятичной точки в мантиссе.

Величины w и d связаны соотношением w>d+6, где 6 – число обязательных позиций, предназначенных для знака числа, десятичной точки мантиссы символа E, порядка числа и его знака. Например, при выводе в спецификации E11.4 числа $-0.256\ 10^{-11}$ и $2.3\ 10^{2}$ будут представлены

-0.2560E-11 0.2300E 03

В спецификации E могут быть записаны любые вещественные числа, поэтому эту спецификацию целесообразно использовать особенно при выводе, когда неизвестен порядок величины. При выводе вещественное число округляется до значения, которое определяется количеством значащих в мантиссе цифр при заданной спецификации.

Спецификация вещественных величин двойной точности аналогична предыдущей и отличается наличием буквы D

Dw.d

Для записи между вводимыми или выводимыми данными w пробелов применяется спецификация типа

wX

Например, спецификация пробелов 3X обеспечит пропуск 3 позиций при вводе или выводе.

Для записи текста между выводимыми данными используется спецификация, имеющая вид

' T ',

где T – текст, состоящий из любых символов Фортрана и русских букв.

Для вывода данных на печать на принтере используется оператор, имеющий вид

PRINT m, <список переменных>,

где m – метка оператора FORMAT, по которому производится преобразование данных для печати.

Пример: при следующем вводе данных

```
      READ (5,5) A,X,M
    5 FORMAT (2F5.2, I3)
```

с клавиатуры первые переменные – вещественного типа, третья переменная – целого типа. Пусть A=1.45, B=-0.56, M=45. Тогда в строке экрана, начиная с крайней позиции, нужно набрать

1.45–0.56 45

При выводе на печать согласно операторам

```
PRINT 5,GX,K,P
5 FORMAT (2X, 'GX=', F4.1, 1X, 'K=', I4, 1X, 'P=', F5.2)
```

для переменных со значениями GX=-7.4, K=29, P=-359.6
будет выведена следующая строка

GX=-7.4 K= 29 P=* * * * *

В данном случае переменная P не могла быть размещена в поле спецификации F5.2 и вместо нее были напечатаны символы "*".

Пусть в результате выполнения программы для аргументов X1=1.5 и X2=3.0 были найдены соответствующие значения функции F1=0.5632945 10^{-2}, F2=-0.1429123$\circ 10^{-14}$. Тогда результаты можно вывести с помощью следующих операторов вывода

```
WRITE (6,10) X1,F1,X2,F2
10 FORMAT(2X, 'X=', F3.1, 1X, 'F=',E10.3)
```

При этом на печать будут выведены следующие две строки

X=1.5 F= 0.563E-02
X=3.0 F=–0.143E-14

При бесформатном выводе допустимы комментарии, которые заключаются в апострофы и печатаются на экране без изменения. Примеры:

1) WRITE(*,*) 'A=', A, 'B=',B
2) WRITE(*,*) 'ЗАДАТЬ ЗНАЧЕНИЕ M'
   READ(*,*) M

В первом примере на экране появится текст A=, затем числовое значение A, потом текст B= и числовое значение B. Во втором примере на экране появится текст.

ЗАДАТЬ ЗНАЧЕНИЕ M

после чего на клавиатуре надо набрать число M.

Пример ввода одномерного целочисленного массива A с количеством элементов не более 100:

```
INTEGER A(100)
WRITE(*,*) 'K='
READ(*,*) K
READ(5,3) (A(I),I=1,K)
3 FORMAT(4(I2,1X))
```

Пусть значение K=9, а элементы массива следующие: A(1)=10, A(2)=24, A(3)=5, A(4)=7, A(5)=2, A(6)=29, A(7)=0, A(8)=44, A(9)=15. Тогда после появления на экране текста K= необходимо набрать на клавиатуре число 9, а затем на клавиатуре следует набрать данные из следующих строк:

```
10 24  5   7
 2 29  0 44
15
```

являющихся элементами массива.

При форматном выводе на экран или принтер следует иметь в виду, что первый символ строки не выводится, а используется для управления печатью. Например, пробел используется для перехода на следующую строку, 0 – для перехода через две строки и т.д. Например, m FORMAT(1X,S),
где S – список спецификаций.

3. Пример выполнения контрольной работы

Контрольная работа содержит четыре задания. Номер варианта для выбора исходных данных задания выбирается по последней цифре шифра зачетной книжки студента.

При оформлении каждого задания контрольной работы необходимо: сформулировать условие задачи, произвести идентификацию и назначить типы констант, переменных и массивов, разработать блок-схему алгоритма, решить контрольный пример, привести текст программы и результаты расчета в виде распечатки, сделать выводы.

Рассмотрим примеры решения заданий контрольной работы.

3.1. Задание 1. Программирование вычислений функции с одной переменной.

3.1.1. Условие задачи

Составить таблицу значений функции по исходным данным, принятым в соответствии с вариантом №: из табл. 4

№ вар.	Вид функции	Аргумент	Пределы изменения аргумента	Шаг	Значения констант
	$S = e^{-ax}\sqrt[3]{ax+b\sin(2x)}$	X	10 – 90 град.	10 град.	a=1.5 в=-1.2

3.1.2. Идентификация и назначение типов констант и переменных

В процессе реализации алгоритма будут использованы следующие переменные и константы соответствующих типов:

x – независимая переменная (аргумент) целого типа (X);

s – зависимая переменная (функция) вещественного типа (S);

x_r – независимая переменная (аргумент) вещественного типа (в радианной мере) (XR);

Δx – шаг изменения аргумента, константа целого типа (DX);

x_o – минимальное значение аргумента, константа целого типа (X0);

x_k – максимальное значение аргумента, константа целого типа (XK);

π – константа вещественного типа, π =3,141593 (PI);

a=1,5 – константа вещественного типа (A);

в= -1,2 – константа вещественного типа (B).

3.1.3. Блок-схема алгоритма

Блок-схема алгоритма представлена на рис. 1.

Примечания к блок-схеме. Так как выражение $\sqrt[3]{ax = b\sin(2x)}$ вычисляется как $e^{0,3333\ln[ax+b\sin(2x)]}$, то следует проверить $[ax + b\sin(2x)]$ на отрицательность.

Вследствие этого вводится блок 7.

3.1.4. Контрольный пример

Для отладки программы необходимо выполнить контрольный пример, При этом из заданного диапазона изменения аргумента выбираются такие два значения, при которых, при возможности, можно было бы подсчитать функцию просто. В данном варианте таких значений аргумента нет. Приходится принимать два любых: X_1=50 град, X_2=90 град.

$$S_1 = e^{-1,5\cdot\frac{50\cdot 3.141593}{180}} \cdot \sqrt[3]{1,5\cdot\frac{50\cdot 3,141593}{180} - 1,2\sin\left(2\cdot\frac{50\cdot 3,141593}{180}\right)} = 0,27\cdot\sqrt[3]{1,309-1,182} =$$

$$= 0,27\cdot\sqrt[3]{0,127} = 0,27\cdot 0,503 = 0,136$$

$$S_2 = e^{-1,5\cdot\frac{90\cdot 3,141593}{180}} \cdot \sqrt[3]{1,5\cdot\frac{90\cdot 3,141593}{180} - 1,2\sin\left(2\cdot\frac{90\cdot 3,141593}{180}\right)} = 0,0948\cdot\sqrt[3]{2,356-0} =$$

$$= 0,0948\cdot\sqrt[3]{2,356} = 0,0948\cdot 1,3306 = 1,261$$

Поскольку натуральный логарифм отрицательного подкоренного выражения (см. выше) отсутствует, то первое выбранное значение аргумента X_1=50 град.

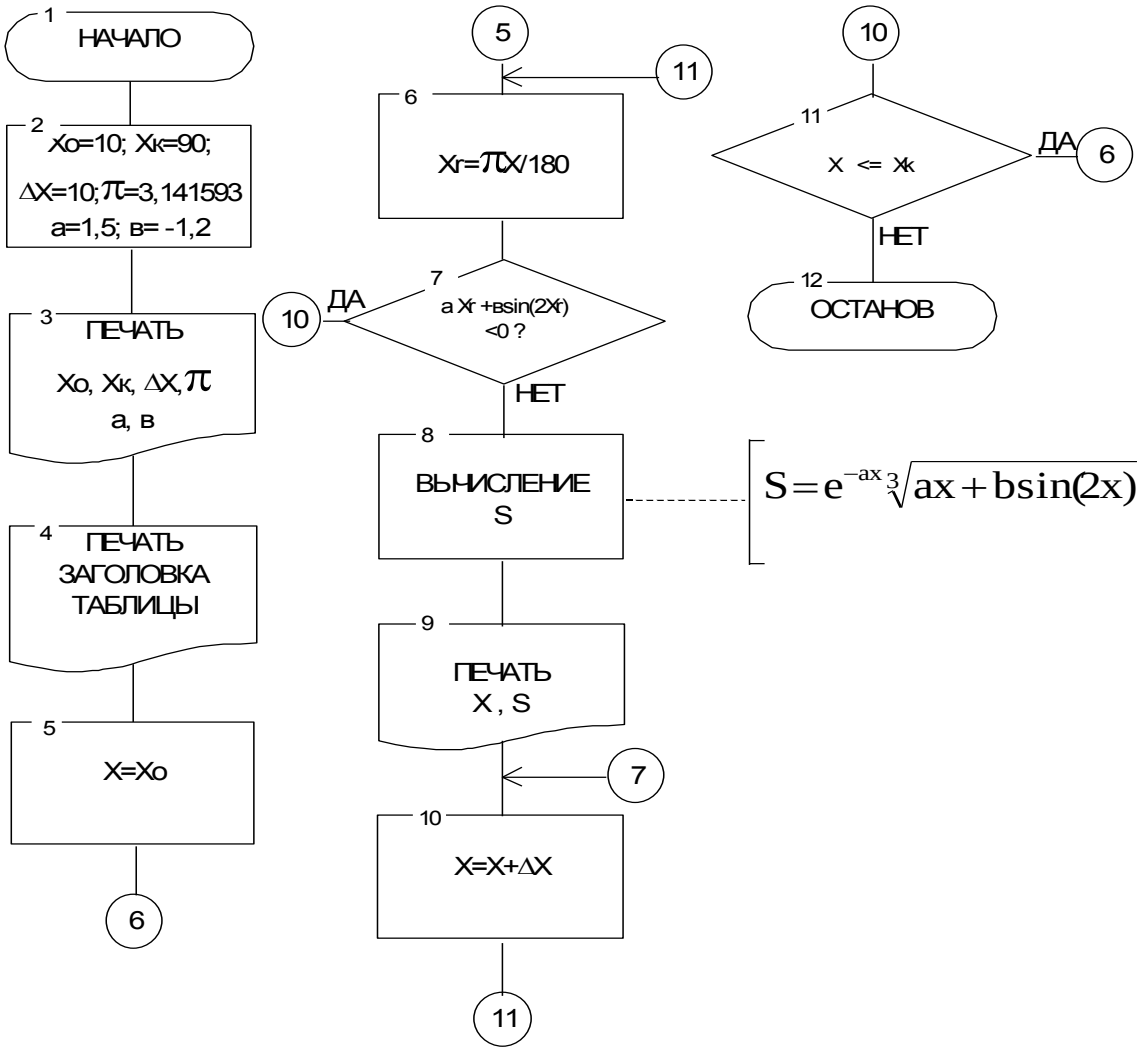


Рис.1. Блок-схема алгоритма

3.1.5 Программа

Программа на Фортране имеет следующий вид:

```
C       ПРОГРАММА ТАБУЛИРОВАНИЯ  ФУНКЦИИ
C       ВЫПОЛНИЛ СТУДЕНТ ПЕТРОВ П.П.
C
        INTEGER X,DX,X0,XK
C       ПРИСВОЕНИЕ НАЧАЛЬНЫХ ЗНАЧЕНИЙ ДАННЫМ
C
        X0=10
        XK=90
        DX=10
        PI=3.141593
        A=1.5
```

```
        B=-1.2
C
C       ВЫВОД НАЧАЛЬНЫХ ЗНАЧЕНИЙ
C
        PRINT 1,X0,XK,DX,PI,A,B
      1 FORMAT (5X,'XO=',I2,' XK=',I2,' DX=',I2,' PI=',F8.6,
       *' A=',F3.1,' B=',F4.1)
```



```
C
C       ПЕЧАТЬ ЗАГОЛОВКА ТАБЛИЦЫ
C
        PRINT 2
      2 FORMAT (5X,15('-')/6X,'X',6X,'S'/5X,('-'))
        X=X0
C
C       ПЕРЕВОД АРГУМЕНТА ИЗ ГРАДУСНОЙ МЕРЫ В РАДИАННУЮ
C
      6 XR=PI*X/180
C
C       ПРОВЕРКА ПОДКОРЕННОГО ВЫРАЖЕНИЯ НА ОТРИЦАТЕЛЬНОСТЬ
C
        IF(A*XR+B*SIN(2*XR).LT.0)GO TO 10
C
C       ВЫЧИСЛЕНИЕ ФУНКЦИИ
C
        S=EXP(-A*XR)*(A*XR)+B*SIN(2*XR))**0.3333
C
C       ПЕЧАТЬ АРГУМЕНТА И ФУНКЦИИ
C
        PRINT 3,X,S
      3 FORMAT(5X,I2,10X,F7.3)
C
C       ПОЛУЧЕНИЕ ОЧЕРЕДНОГО ЗНАЧЕНИЯ АРГУМЕНТА
C
        X=X+DX
C
C       ПРОВЕРКА АРГУМЕНТА НА ПРЕВЫШЕНИЕ МАКСИМАЛЬНОГО ЗНАЧЕНИЯ
C
        IF(X.LE.XK) GO TO 6
C
C       ПРЕКРАЩЕНИЕ ВЫЧИСЛЕНИЙ
C
        STOP
        END
```

3.1.6. Результаты расчета

Примечание. В качестве результата расчета привести распечатку с ЭВМ.

3.1.7. Выводы

Примечание. Если результаты расчета сходятся с контрольным примером, то вывод однозначен – программа работает верно. В противном случае – программа не отлажена, следует выявить ошибки, устранить их, что должно привести только к выше приведенному выводу.

3.2. Задание 2. Программирование вычислительного процесса с выбором расчетной формулы

– 27 –

3.2.1. Условие задачи

Составить таблицу значений функции по исходным данным, принятым в соответствии с вариантом № из табл. 5.

№ вар.	Вид функции	Ограничения	Аргумент	Пределы изменения аргумента	Шаг	Константы
	$Z=\begin{cases} at^2\cdot\ln t \\ 1 \\ e^{at}\cos(bt) \end{cases}$	$t<1$ $1\le t\le 2$ $t>2$	t	0 – 3 рад.	0,3 рад.	a=-0,5 b=2

3.2.2. Идентификация и назначение типов констант и переменных

В процессе реализации алгоритма будут использованы следующие переменные и константы соответствующих типов:

t – независимая переменная(аргумент0 вещественного типа (T);

z – зависимая переменная (функция) вещественного типа (Z);

Δt – шаг изменения аргумента, константа вещественного типа (DT);

t_0 – минимальное значение аргумента, константа вещественного типа (T0);

t_k – максимальное значение аргумента, константа вещественного типа (TK);

a=-0,5 – константа вещественного типа (A);

b=2,0 – константа вещественного типа (B).

3.2.3 Блок-схема алгоритма

Блок-схема алгоритма представлена на рис. 2.

Примечание к блок-схеме (рис.2): блок 6 введен для того, чтобы исключить невыполняемую операцию вычисления логарифма числа, равного нулю.

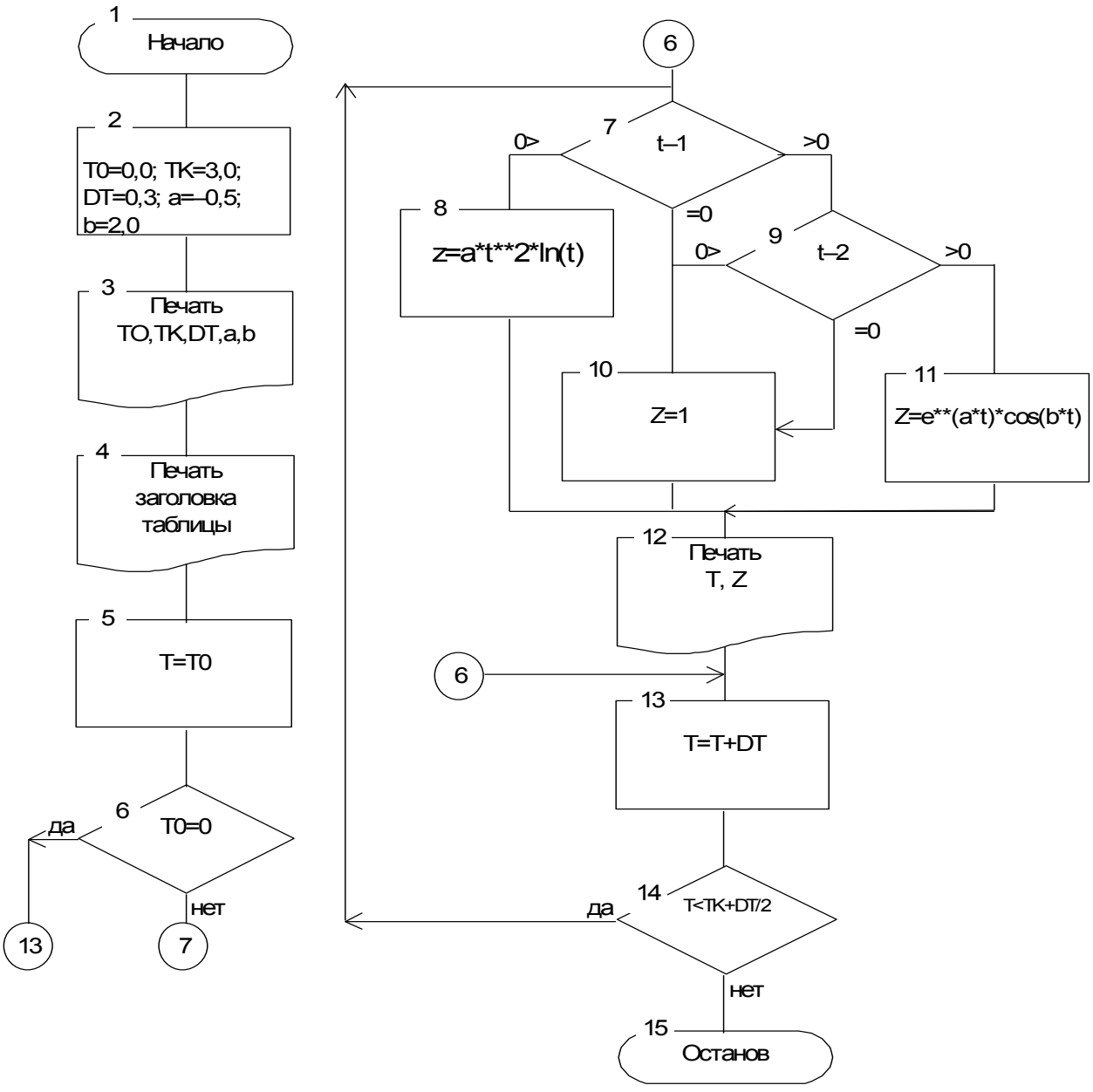


Рис 2. Блок-схема алгоритма.

3.2.4 Контрольный пример

Для отладки программы необходимо проверить значения функции по двум формулам (1-ой и 3-ей). Вторая формула проста, и значение функции очевидно. Значения функции можно посчитать для t_1=0,3 и t_2=2,1.

При t=0,3

$$Z = at^2 \ln t = -0{,}5 \cdot 0{,}3^2 \ln 0{,}3 = -0{,}045 \cdot (-1{,}204) = 0{,}0542;$$

при t=2,1

$$Z = e^{at}\cos(bt) = e^{-0{,}5\cdot 2{,}1}\cos(2 \cdot 2{,}1) = \frac{1}{e^{1{,}05}}\cos 4{,}2 = 0{,}350 \cdot (-0{,}4903) = -0{,}172$$

3.2.5. Программа

Программа на Фортране имеет следующий вид:

```
C       ПРОГРАММА ТАБУЛИРОВАНИЯ ФУНКЦИИ С ВЫБОРОМ
C       РАСЧЕТНОЙ ФОРМУЛЫ
C       ВЫПОЛНИЛ СТУДЕНТ ПЕТРОВ П.П.
C
C       ПРИСВОЕНИЕ НАЧАЛЬНЫХ ЗНАЧЕНИЙ ДАННЫМ
C
        T0=0.
        TK=3.
        DT=.3
        A=-0.5
        B=2.
C
C       ВЫВОД НАЧАЛЬНЫХ ЗНАЧЕНИЙ
C
        PRINT 1,T0,TK,DT,A,B
      1 FORMAT (5X,'T0=',F4.1,' TK=',F4.1,' DT=',F4.1,
       *' A=',F4.1,' B=',F4.1)
```

```
C
C       ПЕЧАТЬ ЗАГОЛОВКА ТАБЛИЦЫ
C
        PRINT 2
      2 FORMAT (5X,12('-')/6X,'T',6X,'Z'/5X,12('-'))
        T=T0
C
C       ИСКЛЮЧЕНИЕ НЕВЫПОЛНИМОЙ ОПЕРАЦИИ ВЫЧИСЛЕНИЯ ЛОГАРИФМА НУЛЯ
C
        IF(T.EQ.0) GO TO 13
C
C       ВЫБОР РАСЧЕТНОЙ ФОРМУЛЫ ОПРЕДЕЛЕНИЯ ЗНАЧЕНИЯ ФУНКЦИИ
C
      7 IF(T-1) 8,10,9
      9 IF(T-2) 10,10,11
      8 Z=A*T**2*ALOG(T)
        GO TO 12
```



```
     10 Z=1
        GO TO 12
     11 Z=EXP(A*T)*COS(B*T)
C
C       ПЕЧАТЬ АРГУМЕНТА И ФУНКЦИИ
C
        PRINT 3,T,Z
      3 FORMAT(5X,F3.1,2X,F7.3)
C
C       ПОЛУЧЕНИЕ ОЧЕРЕДНОГО ЗНАЧЕНИЯ АРГУМЕНТА
C
        T=T+DT
C
C       ПРОВЕРКА АРГУМЕНТА НА ПРЕВЫШЕНИЕ МАКСИМАЛЬНОГО ЗНАЧЕНИЯ
C
        IF(T.LE.TK+DT/2) GO TO 7
C
C       ПРЕКРАЩЕНИЕ ВЫЧИСЛЕНИЙ
C
        STOP
        END
```

3.2.6. Результаты расчета

Примечание. В качестве результатов расчета приводится распечатка с ЭВМ.

2.3.7. Выводы

Примечание. Если результаты расчетов сходятся с контрольным примером, то программа работает верно. В противном случае выявить ошибки программы, исправить программу и распечатать на ЭВМ.

3.3. Задание 3. Программирование вычислений функции с двумя переменными

3.3.1. Условие задачи

Составить таблицу значений функции по исходным данным, принятым в соответствии с вариантом № из табл. 6. При программировании воспользоваться вложенными циклами.

№ вар.	Вид функции	Аргументы	Пределы изменения аргументов	Шаг	Константы
	$y = \frac{ax + \sin(at)}{\sqrt{2t + e^{-0,5x}}}$	x t	1–2 1–2	0,3 0,3	a=0,7

3.3.2. Идентификация и назначение типов констант и переменных

В процессе реализации алгоритма будут использованы следующие переменные и константы соответствующих типов:

x – независимая переменная (аргумент) вещественного типа (X);

t – независимая переменная (аргумент) вещественного типа (T);

y – зависимая переменная (функция) вещественного типа (Y);

Δx – константа (шаг изменения аргумента x) вещественного типа (DX);

Δt – константа (шаг изменения аргумента t) вещественного типа (DT);

x_0 – константа (минимальное значение аргумента x) вещественного типа (X0);

$x_к$ –константа (максимальное значение аргумента x) вещественного типа (XK);

t_o – константа (минимальное значение аргумента t) вещественного типа (T0);

t_k – константа (максимальное значение аргумента t) вещественного типа (TK);

n_x – константа (число повторений аргумента x) целого типа (NX);

n_t – константа (число повторений аргумента t) целого типа (NT);

i – счетчик внешних циклов (повторений аргумента x), переменная целого типа (I);

j – счетчик вложенных циклов (повторений аргумента t), переменная целого типа (J).

Число повторений: аргумента x определяется по формуле

$$n_x = \left[\frac{x_k - x_0}{\Delta x} \right] + 1$$

аргумента t

$$n_t = \left[\frac{t_k - t_0}{\Delta t} \right] + 1,$$

где квадратными скобками обозначается операция выделения целой части числа.

3.3.3. Блок-схема алгоритма

Блок-схема алгоритма представлена на рис. 3.

3.3.4. Контрольный пример

Для отладки программы необходимо проверить значения функции в двух точках при x=1, t=1 и x=2, t=2.

x=1, t=1

$$y = \frac{0,7 \cdot 1 + \sin(0,7 \cdot 1)}{\sqrt{2 \cdot 1 + e^{-0,5 \cdot 1}}} = \frac{0,7 + \sin(0,7)}{\sqrt{2 + \frac{1}{\sqrt{e}}}} = \frac{1,344}{1,614} = 0,832$$

x=2, t=2

$$y = \frac{0,7 \cdot 2 + \sin(0,7 \cdot 2)}{\sqrt{2 \cdot 2 + e^{-0,5 \cdot 2}}} = \frac{1,4 + \sin(1,4)}{\sqrt{4 + \frac{1}{e}}} = \frac{2,385}{2,090} = 1,141$$

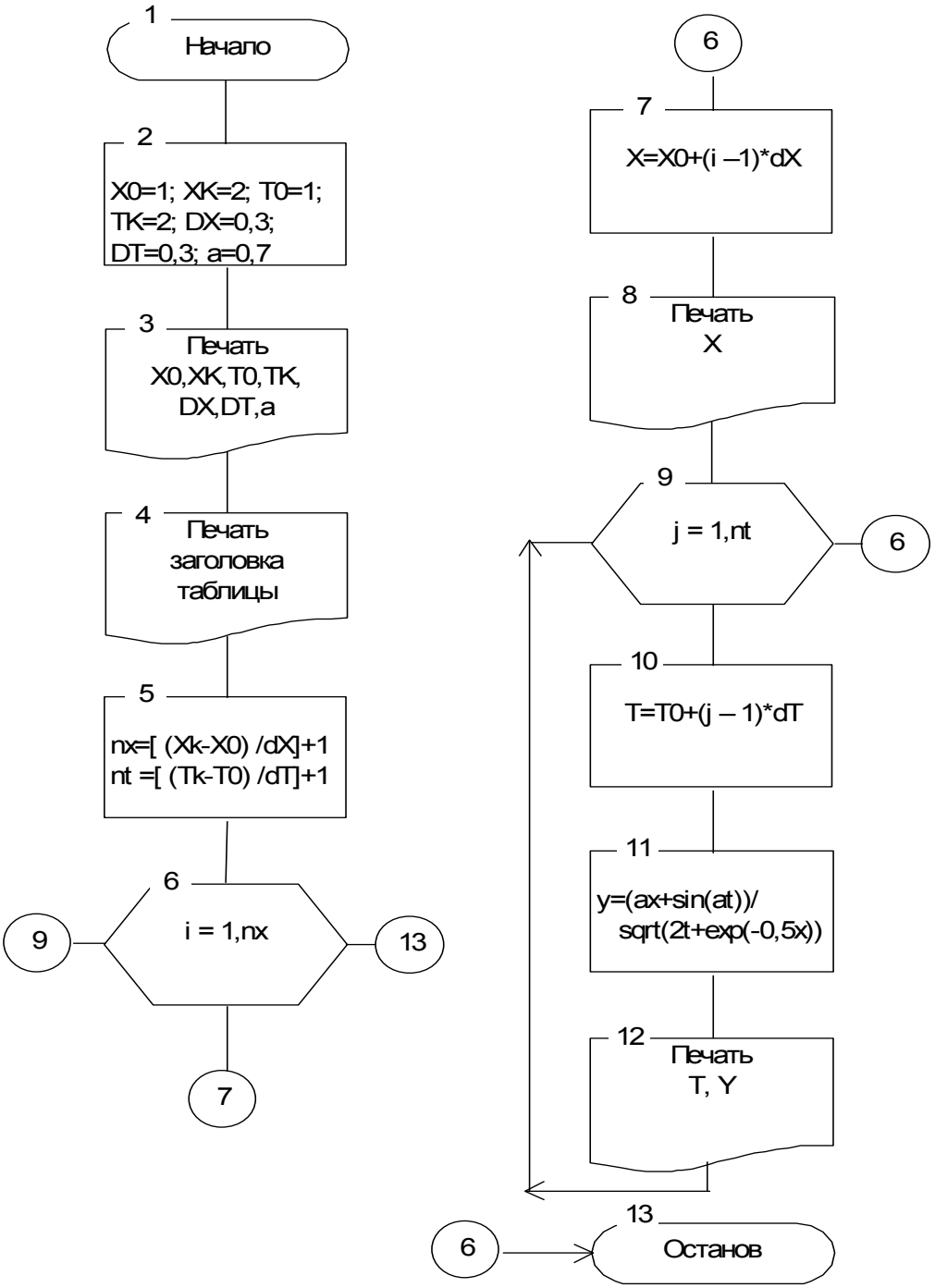


Рис. 3. Блок-схема алгоритма.

3.3.5. Программа

Программа на Фортране выглядит следующим образом:

```
C
C       ПРОГРАММА ТАБУЛИРОВАНИЯ ФУНКЦИИ ДВУХ ПЕРЕМЕННЫХ
C       ВЫПОЛНИЛ СТУДЕНТ ПЕТРОВ П.П.
C
C       ПРИСВОЕНИЕ НАЧАЛЬНЫХ ЗНАЧЕНИЙ ДАННЫМ
C
        DATA DX,DT,X0,XK,T0,TK/.3,.3,1.,2.,1.,2./
C
C       ВЫВОД НАЧАЛЬНЫХ ЗНАЧЕНИЙ
C
        PRINT 1,DX,DT,X0,XK,T0,TK
      1 FORMAT(5X,'DX=',F3.1,' DT=',F3.1,' X0=',F3.1,
       *' XK=',F3.1,' T0=',F3.1,' TK=',F3.1)
C
C       ПЕЧАТЬ ЗАГОЛОВКА ТАБЛИЦЫ
C
        PRINT 2
      2 FORMAT(5X,17('-')/6X,'X',4X,'T',6X,'Y'/5X,17('-'))
C
C       РАСЧЕТ ЧИСЛА ПОВТОРЕНИЙ ЦИКЛОВ
C
        NX=(XK-X0)/DX+1
        NT=(TK-T0)/DT+1
C
C       ЗАГОЛОВОК ВНЕШНЕГО ЦИКЛА
C
        DO 12 I=1,NX
C
C       ТЕКУЩЕЕ ЗНАЧЕНИЕ АРГУМЕНТА X
C
        X=X0+(I-1)*DX
C
C       ВЫВОД АРГУМЕНТА X
C
        PRINT 3,X
C
C       ЗАГОЛОВОК ВЛОЖЕННОГО ЦИКЛА
C
        DO 12 J=1,NT
C
C       ТЕКУЩЕЕ ЗНАЧЕНИЕ АРГУМЕНТА T
C
        T=T0+(J-1)*DT
C
C       ОПРЕДЕЛЕНИЕ ЗНАЧЕНИЯ ФУНКЦИИ
C
        Y=(A*X+SIN(A*T))/SQRT(2*T+EXP(-.5*X))
C
C       ВЫВОД АРГУМЕНТА T И ФУНКЦИИ Y
```

```
C
    12 PRINT 4,T,Y
     3 FORMAT(3X,F3.1)
     4 FORMAT(10X,F3.1,2X,F7.3)
C
C      ПРЕКРАЩЕНИЕ ВЫЧИСЛЕНИЙ
C
       STOP
C
C      КОНЕЦ ПРОГРАММЫ
C
       END
```

3.3.6. Результаты расчета

Примечание. В качестве результатов расчета приводится распечатка с ЭВМ.

3.3.7. Выводы

Примечание. Приводится вывод о сходимости результатов контрольного примера и расчета на ЭВМ. Если результаты не сходятся, то выявляются ошибки программы, программа исправляется и снова реализуется на ЭВМ.

3.4. Задание 4. Программирование процесса обработки одномерных массивов

3.4.1. Условие задачи

Вычислить вектор $C=(C_1,C_2, \dots , C_n)$, каждый элемент которого определяется по формуле

$$C_i = A_i + i \cdot B_i,$$

где: $B_i = \sqrt{|\operatorname{arctg}(0{,}5i - 1{,}5|}$, i=1, 6;

A=(1,0; 0,0; -1,0; 0,1; 0,2; -0,4)

3.4.2. Идентификация и назначение типов констант, переменных и массивов

В процессе реализации алгоритма будут использованы следующие константы, переменные и массивы:

n – константа целого типа (размерность одномерных массивов, максимальное значение счетчика циклов) (N);

i – переменная целого типа (и счетчик циклов) (I);

A – одномерный массив (вектор) вещественного типа размерностью n (A);

B – одномерный массив (вектор) вещественного типа размерностью n (B);

C – одномерный массив (вектор) вещественного типа размерностью n (C).

3.4.3. Блок-схема алгоритма

Блок-схема алгоритма представлена на рис. 4.

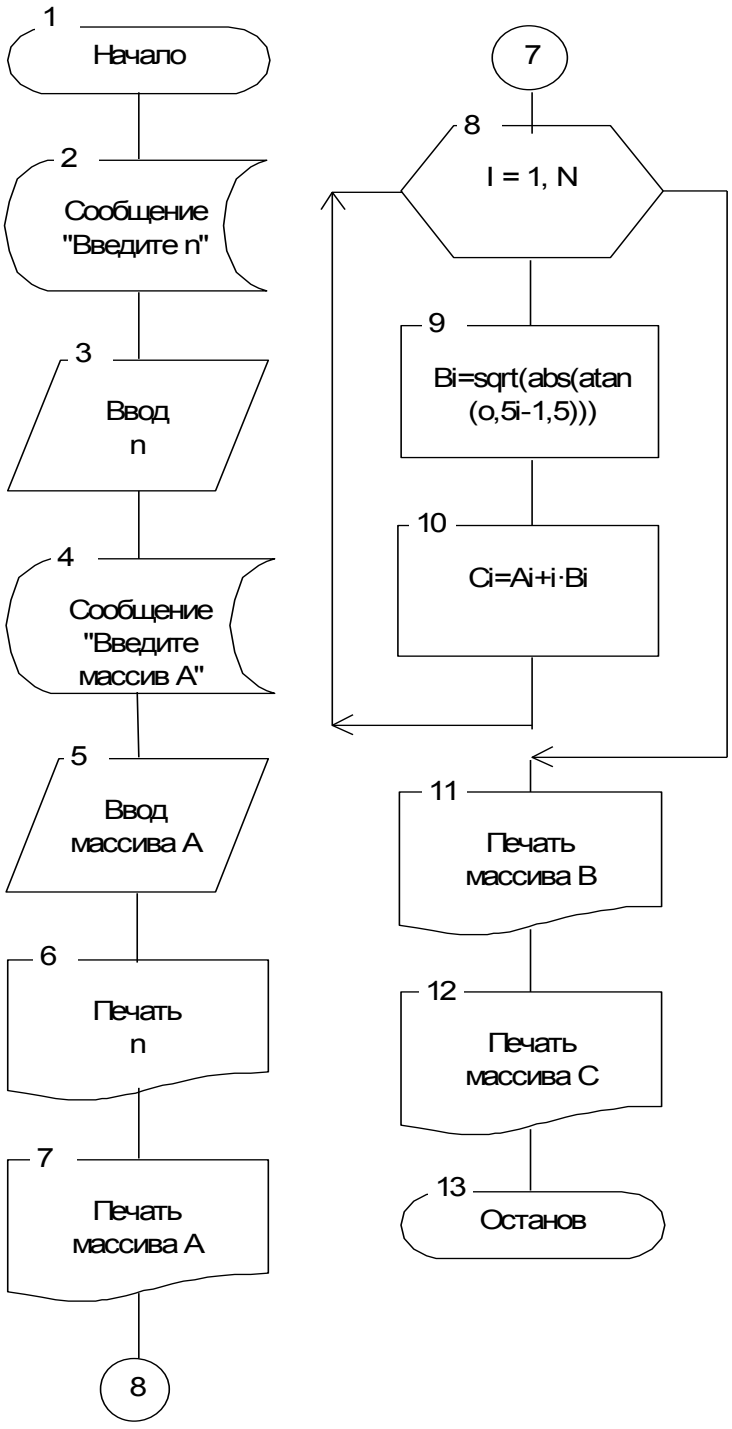


Рис. 4. Блок-схема алгоритма

3.4.4. Контрольный пример

Для отладки программы следует рассчитать два элемента массива C, например, для i=1 и i=6.

i=1

$$B_1 = \sqrt{|\text{arctg}(0{,}5 \cdot 1 - 1{,}5)|} = \sqrt{|\text{arctg}(-1{,}0)|} = \sqrt{|-0{,}785|} = \sqrt{0{,}785} = 0{,}886$$

$$C_1 = A_1 + 1 \cdot B_1 = 1{,}0 + 1 \cdot 0{,}886 = 1{,}886$$

i=6

$$B_6 = \sqrt{|\text{arctg}(0{,}5 \cdot 6 - 1{,}5)|} = \sqrt{|\text{arctg}1{,}5|} = \sqrt{0{,}983} = 0{,}991$$

$$C_6 = A_6 + 6 \cdot B_6 = -0{,}4 + 6 \cdot 0{,}991 = 5{,}548$$

3.4.5 Программа

Программа на Фортране выглядит следующим образом:

```
C
C       ПРОГРАММА ПРОЦЕССА ОБРАБОТКИ ОДНОМЕРНЫХ МАССИВОВ
C       ВЫПОЛНИЛ СТУДЕНТ ПЕТРОВ П.П.
C
C       ОПИСАНИЕ МАССИВОВ
C
        DIMENSION A(6),B(6),C(6)
C
C       ВВОД ИСХОДНЫХ ДАННЫХ
C
        WRITE(5,1)
      1 FORMAT(10X,'ВВЕДИТЕ N')
        READ(5,*) N
        WRITE(5,2)
      2 FORMAT(10X,'ВВЕДИТЕ МАССИВ A')
        READ(5,*) (A(I),I=1,N)
C
C       ВЫВОД ИСХОДНЫХ ДАННЫХ
C
        PRINT 3,N
      3 FORMAT (10X,'N=',I2)
        PRINT4,(A(I),I=1,N)
      4 FORMAT(10X,'МАССИВ A:'/3X,6(2X,F4.1))
C
C       ФОРМИРОВАНИЕ МАССИВОВ B И C
C
        DO 5 I=1,N
        B(I)=SQRT(ABS(ATAN(.5*I-1.5)))
        C(I)=A(I)+I*B(I)
```

```
      5 CONTINUE
C
C       ВЫВОД МАССИВОВ B И C
C
        PRINT 6,(B(I),I=1,N),(C(I),I=1,N)
      6 FORMAT(10X,'МАССИВ B:'/3X,6(1X,F7.3)/
       *10X,'МАССИВ C:'/3X,6(1X,F7.3))
```



```
C
C       ПРЕКРАЩЕНИЕ ВЫЧИСЛЕНИЙ. КОНЕЦ ПРОГРАММЫ
C
        STOP
        END
```

3.4.6. Результаты расчета

Примечание. В качестве результатов расчета приводится распечатка с ЭВМ.

3.4.7. Выводы

Примечание. Приводится вывод о сходимости результатов контрольного примера и расчета на ЭВМ. Если результаты не сходятся, то выявляются ошибки программы, программа исправляется и снова реализуется на ЭВМ.

4. Задания к контрольной работе

Исходные данные к задачам заданий 1–4 по вариантам приведены в табл. 4–7.

Таблица 4

Исходные данные к заданию 1

№ вар.	Вид функции	Аргумент	Пределы изменения аргумента	Шаг	Значения констант
1	$y = ae^{-\sqrt{x}}\cos bx + c$	x	0–90 град.	10 град.	a=1,5 b=2,0 c=-0,75
2	$z = a \cdot \cos(bt \cdot \sin t) + c$	t	0–90 град.	10 град.	a=2,0 b=0,7 c=0,5
3	$z = \sqrt{a + b^{\sin x} + 1}$	x	0–1,5 рад.	0,1 рад.	a=2,0 b=1,2
4	$t = \sqrt[3]{m \cdot tge + c \cdot \sin e}$	e	0–90 град.	10 град.	m=2 c=-1,0
5	$y = \frac{bx^2 - a}{e^{ax} - 1}$	x	-1–1	0,3	a=-0,5 b=2,3
6	$z = bte^{at^2} + a\sqrt{t + 1{,}5}$	t	-1–1	0,2	a=-0,5 b=1,5
7	$S = e^a\sqrt{x+1} + e^b\sqrt{x+1{,}5}$	x	0–2	0,2	a=0,5 b=1,0

8	$y = b^x \operatorname{arctg}\left(\frac{x}{a}\right) - \sqrt[5]{x+a}$	x	3,1–4 рад.	0,1 рад.	a=3,7 b=0,5
9	$z = \frac{x + a\cos(2x)}{x + \sqrt{a + b\sin(3x)}}$	x	1–2 рад.	0,1 рад.	a=4,1 b=-2,3
0	$y = \arcsin\left(\frac{x}{a}\right) - e^{-bx}\sqrt{x+1}$	x	-90–90 град.	30 град.	a=2,3 b=0,75

Таблица 5

Исходные данные к заданию 2

№ вар.	Вид функции	Ограничения	Аргумент	Пределы изменения аргумента	Шаг	Константы
1	$S = \begin{cases} e^{-\sqrt{x}}\cos(ax) \\ \sin(ax) \\ e^{-\sqrt{x}}\sin(ax) \end{cases}$	x<a x=a x>a	x	2–3 рад.	0,1 рад.	a=2,7
2	$y = \begin{cases} e^{-bx}\sin(bx) \\ \cos(ax) \\ e^{-ax}\cos(bx) \end{cases}$	x<a $a \le x \le b$ x>b	x	0–4 рад.	0,5 рад.	a=1,0 b=3,0
3	$S = \begin{cases} t \cdot \sqrt[3]{t-a} \\ t \cdot \sin(at) \\ e^{-at}\cos(at) \end{cases}$	t>a t=a t<a	t	1–5 рад.	0,5 рад.	a=2,5
4	$Z = \begin{cases} ai + b/i \\ ai^2 + b \\ i^3 \end{cases}$	i<5 $5 \le i \le 10$ i>10	i	3-15	1	a=2,0 b=-0,5
5	$y = \begin{cases} \pi x^2 + 7/x^2 \\ ax^3 + 7\sqrt{x} \\ \lg(x + 7\sqrt{x}) \end{cases}$	x<1,3 x=1,3 x>1,3	x	0,9–2	0,2	a=1,5
6	$Z = \begin{cases} (\ln^3 x + a^2)/\sqrt{x+t} \\ \sqrt{x+t} + 1/x \\ \cos x + t\sin^2 x \end{cases}$	x<0,5 x=0,5 x>0,5	x	0,1–2	0,2	a=1,5 t=2,2
7	$S = \begin{cases} e\cos(ax) \\ \sin(ax) \\ e\sin(ax) \end{cases}$	x<a x=a x>a	x	10–90 град.	10 град.	a=1,1
8	$y = \begin{cases} a/i + bi^2 + c \\ i \\ ai + bi^3 \end{cases}$	i<3 $3 \le i \le 7$ i>7	i	2–10	1	a=1,3 b=1,0 c=0,5

9	$y=\begin{cases} ae^{\sin x}+2{,}5 \\ e^{\cos x}+a \\ \sin(x)/(a+e^{x}) \end{cases}$	x<0,3 x=0,3 x>0,3	x	0–1 рад.	0,2 рад.	a=1,5
0	$Z=\begin{cases} \sqrt{at^2+b\sin t+1} \\ at+b \\ \sqrt{at^2+b\cos t+1} \end{cases}$	t<0,1 t=0,1 t>0,1	t	-1–1 рад.	0,1 рад.	a=2,1 b=0,37

Таблица 6

Исходные данные к заданию 3

№ вар.	Вид функции	Аргументы	Пределы изменения аргументов	Шаги	Константы
1	$y=ae^{2xt}\cos(\frac{\pi}{2}+t)$	x t	0–1 0–π рад.	0,1 0,3 рад.	a=-3,1
2	$Z=ae^{-x}\sin(ax)+\sqrt{ay}$	x y	-1–1 рад. 1–5	0,2 рад. 1,5	a=0,75
3	$S=x^{-0,75}\sin(x+a)\ln(y+a)$	x y	0–2 рад. 0–1	0,4 рад. 0,3	a=0,7
4	$y=(\pi t+1)e^{x}\cos(t-a)$	t x	2–3 рад. 1–2	0,3 рад. 0,1	a=-3,1
5	$y=b\cdot 2^{-x^2y}+\sqrt{b\cos(2x)}$	x y	0–π/2 рад. 0–1	0,2 рад. 0,25	b=1,2
6	$S=\sqrt[5]{axy^2+1{,}3}\sin(x-a)$	x y	2–5 рад. -1–1	0,5 рад. 0,5	a=1,9
7	$Z=b\cdot 2^{xy^2}+b\cos(2x)$	x y	2–4 рад. 0–1	0,2 рад. 0,1	b=1,2
8	$S=\sqrt{ax^2y+1{,}3}\sin(x-a)$	x y	2–5 рад. -1–1	0,5 рад. 0,5	a=1,9
9	$Z=b^{xy}\cdot 2+b\cos(2x-1)$	x y	2–4 рад. -1–1	0,2 рад. 0,5	b=2,2
0	$Z=a(xy)^{0,7}\cos(ax)$	x y	0–1 рад. 3–4	0,2 рад. 0,3	a=0,2

Таблица 7

Исходные данные к заданию 4

№ вар.	Условие задачи
1	Вычислить вектор $Z=(Z_1, Z_2, \dots, Z_{10})$, равный сумме векторов $X=(x_1, x_2, \dots, x_{10})$ и $Y=(y_1, y_2, \dots, y_{10})$ по формуле $Z_i = x_i + y_i$, i = 1,10 Вектор X=(0,1; 0,2; 0,3; 0,8; 1,6; -4,2; -0,4; 0,5; 0,8; 5,3)

	Вектор Y вычисляется по зависимости $y_i = 0{,}7\sin(0{,}5i) + 1{,}3\cos(i+1)$, i = 1,10
2	Вычислить $H = \sum_{i=1}^{n}(a_i - b_i)^2$, где: $a_i=i^2$, $b_i=i^2$, если i – четное; $a_i=i$, $b_i=i^3$, если i – нечетное. Расчет произвести при n=5, предусмотрев вывод на печать векторов $A=(a_1, a_2, \ldots, a_n)$ и $B=(b_1, b_2, \ldots, b_n)$

Продолжение табл. 7

№ вар.	Условие задачи
3	Вычислить координаты x_c, y_c, z_c центра тяжести системы материальных точек с координатами x_i, y_i, z_i для i=1,n и с массами m_i, i=1,n по зависимостям $x_c = \frac{\sum_{i=1}^{n} m_i x_i}{\sum_{i=1}^{n} m_i}$; $y_c = \frac{\sum_{i=1}^{n} m_i y_i}{\sum_{i=1}^{n} m_i}$; $z_c = \frac{\sum_{i=1}^{n} m_i z_i}{\sum_{i=1}^{n} m_i}$, где: n=5; x=(-2,; 1,; -2,; 4,; 6,) y=(5,; -3,; 1,; 4,; -1,) z=(7,; 6,; -4,; -2,; 1,) m=(1,; 2,; 2,; 2,; 1,)
4	Вычислить вектор $X=(x_1, x_2, \ldots, x_{10})$, где: $x_i = \begin{cases} \operatorname{arctg}\frac{\sqrt{i}+2}{n+3}, & \text{если } \operatorname{tg} i \le 3; \\ e^{i+\cos(n)}, & \text{если } \operatorname{tg} i > 3 \end{cases}$ i=1,10; n=10
5	Вычислить вектор $Y=(y_1, y_2, \ldots, y_8)$ по зависимости $y_i = 2\sin x_i + \frac{1}{\cos x_i}$, где: $x_i \in X=(x_1, x_2, \ldots, x_8)$ и $x_1=0$, а $x_{i+1} - x_i = 0{,}1$
6	Решить уравнение относительно x $a_i x + b_i = 0$, i=1, 5, где: A=(20,3; 0,05; 0,0; 13,1; 9,5) B=(13,2; 1,64; 20,6; 14,6; 2,2) Выводить на печать значения корней только при $a_i \neq 0$
7	Вычислить скалярное произведение векторов $X=(x_1, x_2, \ldots, x_6)$ и $Y=(y_1, y_2, \ldots, y_6)$ где: $y_i = 0{,}1\operatorname{tg}(0{,}1 \cdot i)$, i=1,6

	X=(0,1; 0,2; 0,3; -0,4; 0,2; 0,3)
8	Сформировать вектор Y=(y_1, y_2, … , y_6), значения элементов которого равны элементам вектора X=(x_1, x_2, … , x_6), упорядоченным по возрастанию. X=(29,; 13,; 64,; 72,; 2,; 99,)
9	Вычислить вектор A=(a_1, a_2, … , a_{10}), значения элементов которого определяются как: $a_i = \begin{cases} \sin\left(\frac{i^2+1}{n}\right), & \text{если } \sin\left(\frac{i^2+1}{n}\right) > 0 \\ \cos\left(i+\frac{1}{n}\right), & \text{если } \sin\left(\frac{i^2+1}{n}\right) \le 0 \end{cases}$, i=1, 10

Окончание табл. 7

№ вар.	Условие задачи
0	Вычислить математическое ожидание $\bar{x}$ и средний модуль отклонения m по зависимостям $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$; $m = \frac{1}{n}\sum_{i=1}^{n} \lvert x_i - \bar{x} \rvert$, где: n=6; X=(0,0002; 0,000153; 0,00062; 0,00096; 0,00087; 0,00101)