Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Магнитогорский государственный технический университет
им. Г. И. Носова»
Многопрофильный колледж

Методические указания
по выполнению практических (лабораторных)работ
ОП.08 ТЕОРИЯ АЛГОРИТМОВ
«Профессиональный цикл»
программы подготовки специалистов среднего звена
специальности 09.02.03 Программирование в компьютерных системах
(базовой подготовки)

Магнитогорск, 2017

ОДОБРЕНО:
Предметной/Предметно-цикловой комиссией
Информатика и вычислительная техника
Председатель И.Г.Зорина
Протокол № _7__ от _14_ марта 2017_

Методической комиссией МпК
Протокол №4 от «23» марта 2017г

Составитель (и):
преподаватель ФГБОУ ВО МГТУ МпК,
к.т.н., доцент В.Д.Тутарова

Методические указания по выполнению практических работ разработаны на основе рабочей программы учебной дисциплины «ТЕОРИЯ АЛГОРИТМОВ».

Содержание практических работ ориентировано на подготовку студентов к освоению профессионального модуля основной профессиональной образовательной программы по специальности 09.02.03 Программирование в компьютерных системах базового уровня и овладению профессиональными компетенциями.

Содержание

ВВЕДЕНИЕ

Важную часть теоретической и профессиональной практической подготовки студентов составляют практические занятия. Являясь частью изучения учебной дисциплины, они призваны, экспериментально подтвердить теоретические положения и формировать общие и профессиональные компетенции, практические умения.

Ведущей дидактической целью практических занятий является формирование практических умений - профессиональных (умений выполнять определенные действия, операции, необходимые в последующем в профессиональной деятельности) или учебных (умений решать задачи по математике, физике, химии, информатике и др.), необходимых в последующей учебной деятельности по математическим и естественно-научным, общепрофессиональным дисциплинам.

Состав и содержание практических работ направлены на реализацию действующих федеральных государственных образовательных стандартов среднего профессионального образования.

В соответствии с рабочей программой учебной дисциплины «ТЕОРИЯ АЛГОРИТМОВ» предусмотрено проведение практических работ.

В результате освоения дисциплины обучающийся должен

уметь:

- разрабатывать алгоритмы для конкретных задач;
- определять сложность работы алгоритмов;

Содержание дисциплины ориентировано на подготовку студентов к освоению профессиональных модулей ОПОП по специальности и овладению профессиональными компетенциями:

ПК 1.1. Выполнять разработку спецификаций отдельных компонент.

ПК 1.2. Осуществлять разработку кода программного продукта на основе готовых спецификаций на уровне модуля.

В процессе освоения дисциплины у студентов должны формироваться общие компетенции:

ОК 1. Понимать сущность и социальную значимость своей будущей профессии, проявлять к ней устойчивый интерес.

ОК 2. Организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.

ОК 3. Принимать решения в стандартных и нестандартных ситуациях и нести за них ответственность.

ОК 4. Осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.

ОК 5. Использовать информационно-коммуникационные технологии в профессиональной деятельности.

ОК 6. Работать в коллективе и в команде, эффективно общаться с коллегами, руководством, потребителями.

ОК 7. Брать на себя ответственность за работу членов команды (подчиненных), за результат выполнения заданий.

ОК 8. Самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.

ОК 9. Ориентироваться в условиях частой смены технологий в профессиональной деятельности.

Выполнение студентами практических работ по учебной дисциплине «ТЕОРИЯ АЛГОРИТМОВ» направлено на:

- обобщение, систематизацию, углубление, закрепление, развитие и детализацию полученных теоретических знаний по конкретным темам учебной дисциплины;
- формирование умений применять полученные знания на практике, реализацию единства интеллектуальной и практической деятельности;
- развитие интеллектуальных умений у будущих специалистов: аналитических, проектировочных, конструктивных и др.;
- выработку при решении поставленных задач профессионально значимых качеств, таких как самостоятельность, ответственность, точность, творческая инициатива.

Продолжительность выполнения практической работы составляет не менее двух академических часов и проводится после соответствующей темы, которая обеспечивает наличие знаний, необходимых для ее выполнения.

ПЕРЕЧЕНЬ ПРАКТИЧЕСКИХ РАБОТ

Наименование разделов и тем	**Содержание учебного материала, практические занятия, самостоятельная работа обучающихся**	**Объем часов**
Раздел 1. **Основные модели алгоритмов**	1. Составления схемы алгоритмов. Словесная запись алгоритмов	**2**
Раздел 2. **Методы построения алгоритмов**	2. Построение линейных алгоритмов	**2**
	3. Построение алгоритмов с ветвлением	**2**
	4. Построение циклических алгоритмов	**2**
Раздел 3. **Методы вычисления сложности работы алгоритмов**.	5. Построение алгоритмов при работе с массивами	**2**
	6. Построение алгоритмов поиска значений	**2**
	7. Построение алгоритмов сортировки	**2**
	8. Построение рекурсивных алгоритмов	**2**
	Итого	**16**

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

Практические работы

1. Составления схемы алгоритмов. Словесная запись алгоритмов
2. Построение линейных алгоритмов
3. Построение алгоритмов с ветвлением
4. Построение циклических алгоритмов
5. Построение алгоритмов при работе с массивами
6. Построение алгоритмов поиска значений
7. Построение алгоритмов сортировки
8. Построение рекурсивных алгоритмов

Цель работы:

1. Научиться применять базовые понятия теории приближенных методов решения задач на ЭВМ.
2. Сформировать представление об алгоритмах решения прикладных задач.
3. Выбрать и обосновать наиболее рациональный метод и алгоритм решения задачи;
4. Разработать алгоритм для решения прикладных задач.

Количество часов: _16_

Материальное обеспечение:

- ПЭВМ;
- рабочее место преподавателя.

Технические средства обучения:

- учебная аудитория, оснащенная мультимедийным оборудованием;
- компьютерный класс;
- программа Microsoft Visio.

Порядок выполнения работы

1. Составить математическую модель задачи.
2. Выбрать и обосновать наиболее рациональный метод решения задачи;
3. Разработать алгоритм для решения задачи.

Форма предоставления результата

1. Алгоритм программы

ПРАКТИЧЕСКАЯ РАБОТА №1. СОСТАВЛЕНИЯ СХЕМЫ АЛГОРИТМОВ. СЛОВЕСНАЯ ЗАПИСЬ АЛГОРИТМОВ

Алгоритмы классифицируются по форме представления (рис. 4.1).

Рис. 4.1. Классификация алгоритмов по форме представления

Рассмотрим различные формы представления алгоритмов на примере вычисления корней квадратного уравнения$ax^2 + bx + c = 0$.

Словесный способ записи алгоритмов представляет собой описание последовательных этапов обработки данных. Алгоритм задается в произвольном изложении на естественном языке.

Словесный алгоритм вычисления корней квадратного уравнения$ax^2 + bx + c = 0$ будет иметь следующий вид:

1. Задаем коэффициенты уравнения a, b, c;
2. Если значение коэффициента $a \neq 0$, то вычисляем дискриминант по формуле $D = b^2 - 4ac$;
3. Если выполняется условие $D > 0$, то корни квадратного уравнения $x_1 и x_2$ вычисляем по формуле $x_{1,2} = \frac{-b \pm \sqrt{D}}{2a}$;
4. Если выполняется условие $D = 0$, то вычисляем один корень квадратного уравнения x_1 по формуле $x_1 = \frac{-b}{2a}$;
5. Если выполняется условие $D < 0$, то выводим сообщение «Нет действительных корней».

Табличные алгоритмы оформляются в виде таблицы и используются для развития культуры оформления решения задач, для отображения связи с другими предметами и для формирования точности, аккуратности и пунктуальности В табл.4.1 приведена табличная форма вычисления корней квадратного уравнения$ax^2 + bx + c = 0$.

Таблица 4.1

Табличная форма вычисления корней квадратного уравнения$ax^2 + bx + c = 0$

Шаг	Алгоритм	Образец выполнения
1	Внимательно прочитайте текст задачи	Найдите корни квадратного уравнения вида $x^2 + 5x - 6 = 0$

Шаг	Алгоритм	Образец выполнения
2	Запишите в «Дано» буквенное обозначение и числовое значение известных по тексту коэффициентов	Дано: $a = 1, b = 5,$ $c = -6$
3	Под горизонтальной чертой запишите буквенное обозначение неизвестной величины со знаками «=» и «?»	x_1 = ? x_2 = ?
4	Если значение коэффициента $a \neq 0$, то по формуле $D = b^2 - 4ac$ вычислите дискриминант.	Так как $a \neq 0$ ($a = 1$), $D = 5^2 - 4 \cdot 1 \cdot (-6)$ $D = 49$
5	– если выполняется условие $D > 0$, то корни квадратного уравнения x_1 и x_2 вычисляем по формуле $x_{1,2} = \frac{-b \pm \sqrt{D}}{2a}$, – если выполняется условие $D = 0$, то вычисляем один корень квадратного уравнения x_1 по формуле $x_1 = \frac{-b}{2a}$, – если выполняется условие $D < 0$, то выводим сообщение «Нет действительных корней».	Так как $D > 0$ ($D = 49$), $x_1 = \frac{-5+\sqrt{49}}{2 \cdot 1} = 1$ $x_2 = \frac{-5-\sqrt{49}}{2 \cdot 1} = -6$
6	Запишите ответ	Ответ: $x_1 = 1$ и $x_2 = -6$.

Графический способ представления алгоритмов (в виде блок-схемы) наиболее распространенная форма записи алгоритмов. При этом представлении алгоритм изображается в виде последовательности связанных между собой функциональных блоков, каждый из которых соответствует выполнению одного или нескольких действий. На рис.4.2 представлена блок-схема вычисления корней квадратного уравнения $ax^2 + bx + c = 0$.

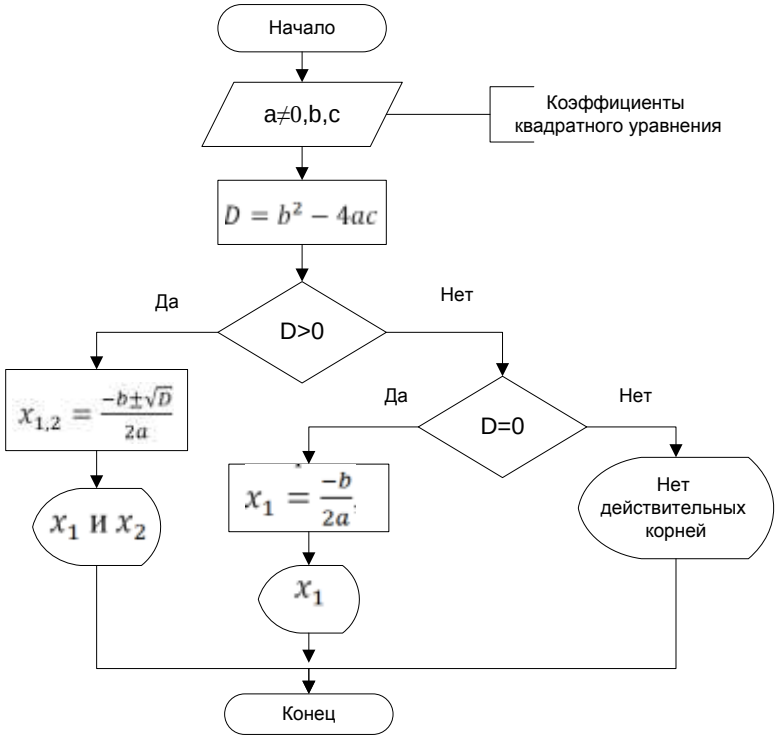


Рис.4.2. Блок-схема алгоритма решения квадратного уравнения

Форма записи алгоритма в виде псевдокодов представляет собой полуформализованные описания алгоритмов на условном алгоритмическом языке, включающие в себя как элементы языка программирования, так и фразы естественного языка, общепринятые математические обозначения и др.

Алгоритмический язык – формальный язык, используемый для записи, реализации или изучения алгоритмов.

Также понятием алгоритмический язык иногда называют [7]:

- семейство языков программирования Алгол;
- учебный алгоритмический язык (школьный алгоритмический язык, русский алгоритмический язык);
- ДРАКОН – Дружелюбный Русский Алгоритмический язык, Который Обеспечивает Наглядность.

Во второй половине 1980-х годов под руководством академика А.П.Ершова была разработана методика обучения программированию в средней и высшей школе и, непосредственно, сам школьный алгоритмический язык КуМир (Комплект Учебных МИРов). Это простой алголоподобный язык с русской лексикой и встроенными командами управления программными исполнителями (Робот, Чертёжник).

При вводе программы КуМир осуществляет постоянный полный контроль ее правильности, сообщая на полях программы обо всех обнаруженных ошибках.

При выполнении программы в пошаговом режиме КуМир выводит на поля результаты операций присваивания и значения логических выражений. Это позволяет ускорить процесс освоения азов программирования.

КуМир работает в операционных системах Windows или Linux.

Ниже представлен пример алгоритма решения квадратного уравнения с применением алгоритмического языка КуМир.

```
    алгКвад_кор (аргвещa,b,c, резвещx1,x2)
нач
| вводa,b,c
    вещD
    D:=b*b-4*a*c
    Выбор
    приD>0: x1:= (-b+sqrt(D))/2*a;x2:= (-b-sqrt(D))/2*a
    приD=0: x1:= -b/2*a
    иначевывод"Нет действительный корней"
    все
кон
```

Все ранее рассмотренные алгоритмы могут допускать неточности при изображении команд, что, в свою очередь, может привести к неполному пониманию процесса. Как было сказано выше, основным исполнителем алгоритмов в современном мире является компьютер, что требует записи алгоритма на понятном ему языке.

Следовательно, язык для записи алгоритмов должен быть формализован. Такой язык принято называть языком программирования, а запись алгоритма на этом языке - программой для компьютера.

Программа, создаваемая человеком-программистом, представляет собой текст, состоящий из знаков, как правило, букв, цифр и специальных знаков. Знаки в тексте программы часто объединены в последовательности – ключевые слова, слова объединены в предложения языка программирования – операторы. Каждый оператор, как правило, записывается в отдельную строку текста программы.

Таким образом, текстовое программирование представляет собой иерархическую последовательность знаков, слов, операторов, записываемых и читаемых последовательно, как обычный текст человеческой письменности. Ниже приведены примеры программного способа записи алгоритмов на языках программирования Паскаль и C++ (среда разработки MicrosoftVisualStudio).

Паскаль

```
var
  a,b,c,d,x1,x2: real;
begin
{Вводим значения a, b и c}
 write('Input a, b, c:');
  readln(a,b,c);
{Вычисляем дискриминант}
 d:= b*b - 4*a*c;
{Если дискриминант больше 0, то вычисляем корни и выводим на экран}
  if d > 0 then
begin
  x1:= (b - sqrt(d))/(2*a);
  x2:= (-b - sqrt(d))/(2*a);
  writeln('x1 = ',x1:6:1);
  writeln('x2 = ',x2:6:1);
end;
{Если дискриминант равен 0, то вычисляем один корень и выводим на экран}
 if d = 0 then
begin
  x1:= -(b/(2*a));
  writeln('Root = ',x1:6:1);
end;
{Если дискриминант меньше 0, то выводим сообщение}
  if d < 0 then
begin
  writeln('No valid roots');
end;
  readln;
end.
```

C++

```
#include "stdafx.h"
#include <stdio.h>
#include <math.h>
int _tmain(int argc, _TCHAR* argv[])
{
        float a, b, c,d;
        printf("Input a, b, c: ");
        scanf("%f%f%f",&a,&b,&c);
        if (a!=0)
        {
```

```
        //Вычисляем дискриминант}
    d = b*b - 4*a*c;
            printf("\nd =%5.2f",d);
                // Если дискриминант больше 0,
                //то вычисляем корни и выводим на экран
            if (d > 0)
            {
                float x1 = (-b - d) /2 /a;
                float x2 = (-b + d) /2 /a;
                printf("\nx1 =%5.2f",x1);
                printf("\tx2 =%5.2f",x2);
            }
                // Если дискриминант равен 0,
                //то вычисляем один корень и выводим на экран
            else if (d == 0)
            {
                float x = (-b) /2 /a;
            printf("\nx =%5.2f",x);
            }
                // Если дискриминант меньше 0,
                //то выводим сообщение
            elseif (d< 0)
                printf("\nNo valid roots");

            getchar();getchar();
            return 0;
    }
}
```

Алгоритмы классифицируются по структуре (рис. 4.3).

Линейный алгоритм – алгоритм, все этапы которого выполняются однократно и строго последовательно.

Разветвленный алгоритм – это алгоритм, включающий выбор тех или иных действий в зависимости от какого-либо условия.

Циклический алгоритм – это алгоритм, в котором предусмотрено многократное выполнение одной и той же последовательности действий.

Вспомогательный алгоритм – это алгоритм, созданный ранее и вызываемый из основного алгоритма как команда.

Комбинированный алгоритм – это алгоритм, в котором встречаются два вида алгоритма циклический и разветвляющий.

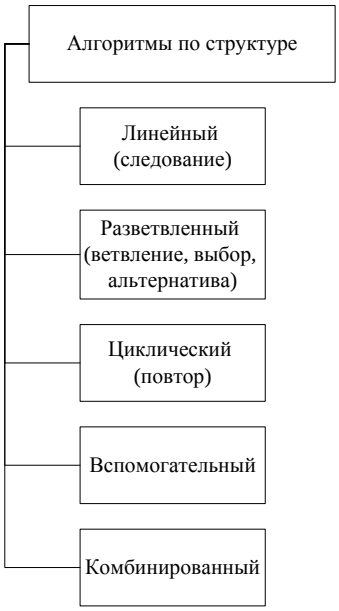


Рис.4.3. Классификация алгоритмов по структуре

Практическая работа №2.
Построение Линейных алгоритмов

Линейный алгоритм – это последовательность операций, выполняющихся последовательно друг за другом. Линейные алгоритмы используются при вычислении арифметических и алгебраических выражений и при решении ряда некоторых житейских задач, например, при копировании файлов с компьютера на USB флеш-накопитель:

1. Включить компьютер;
2. Установить флеш-накопитель в соответствующий USB-порт;
3. Выбрать файлы для копирования;
4. Скопировать файлы с компьютера на флеш-накопитель;
5. Извлечь безопасно флеш-накопитель из USB-порта.

Общий вид записи линейного алгоритма в виде блок-схемы представлен на рис. 7.1.

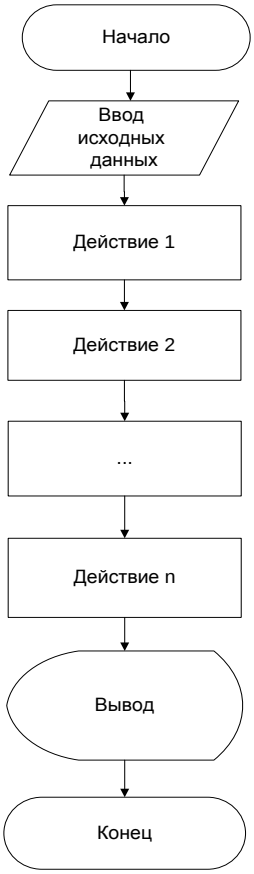


Рис.7.1 . Общий вид линейного алгоритма в виде блок-схемы

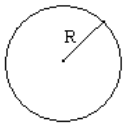
Рассмотрим ряд задач с использованием линейных алгоритмов.

Задача 7.1

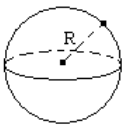
1. Формулировка задачи

Составить алгоритм вычисления длины окружности, площади круга, площади сферы и объема шара по заданному радиусу окружности.

2. Математическая постановка задачи



Для расчета перечисленных характеристик воспользуемся формулами:
длина окружности – $L = 2\pi R$;



площадь круга – $S_{кр} = \pi R^2$;
площадь сферы – $S_{сф} = 4\pi R^2$;
объем шара – $V = \frac{4}{3}\pi R^3$,
где π – число Пи, математическая константа, которая выражает отношение длины окружности к её диаметру $\pi \approx 3{,}14159265358979323846264 3\ldots$, R – радиус окружности.

3. Выбор переменных программы

Из приведенного выше решения определяем следующие переменные:

– исходные данные – радиус окружности (R);
– справочные данные – число π (Pi);
– результат – длина окружности (L), площадь круга ($S_{кр}$), площадь сферы ($S_{сф}$) и объем шара (V).

4. Блок-схема алгоритма

Составим алгоритм решения данной задачи в виде блок-схемы (рис. 7.2) и на алгоритмическом языке.

алг Линейн_алгоритм (**арг**вещR, Pi, **рез**вещL, Skr, Ssf, V)
дано | R> 0
нач
вводR; Pi:=3.14
L:=2*Pi*R
Skr:=Pi*R*R
Ssf:=4*Pi*R*R
V:=4/3*Pi*R*R*R
выводL, Skr, Ssf, V
Кон

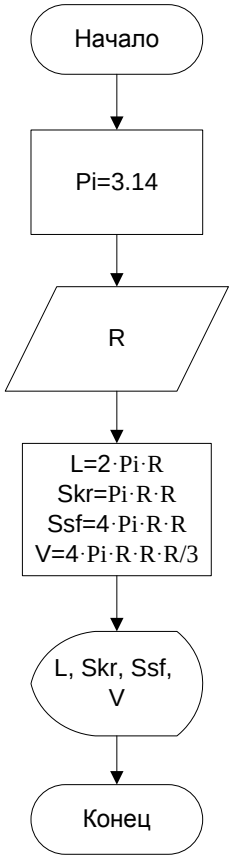
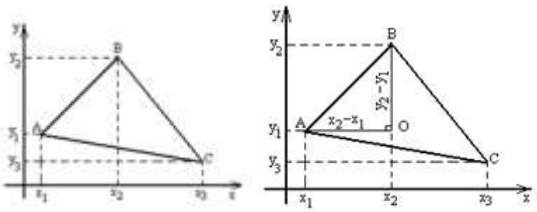


Рис. 7.2. Блок-схема к задаче 7.1

Задача 7.2

1. Формулировка задачи

Треугольник задается координатами своих вершин на плоскости: *A(x1,y1), B(x2,y2), C(x3,y3).* Требуется составить алгоритм программы, которая вычисляет площадь треугольника *ABC*.



2. Математическая постановка задачи

Для решения задачи можно использовать формулу Герона: $S = \sqrt{p(p-A)(p-B)(p-C)}$, где p – полупериметр. Для вычисления длины сторон по координатам вершин необходимо воспользоваться формулой $|AB| = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$, где $|AB|$ – длина сторона треугольника; (x_1,y_1), (x_2,y_2) – координаты вершин *A и B*.

3. Выбор переменных программы

Из приведенного выше решения определяем следующие переменные:

- исходные данные – координаты вершин на плоскости (x_1,y_1), (x_2,y_2), (x_3,y_3);
- промежуточные данные – длины сторон (*A, B, C*), *p* – полупериметр.
- результат – площадь треугольника (*S*).

4. Блок-схема алгоритма

Составим алгоритм решения данной задачи в виде блок-схемы (рис. 7.3) и на алгоритмическом языке.

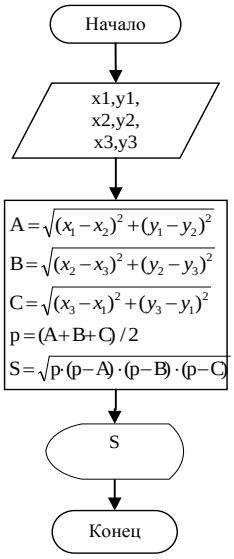


Рис. 7.3. Блок-схема алгоритма к задаче 7.2

алг Прощадь (**арг**цел x1, x2, x3, y1, y2, y3, **рез**вещ A, B, C, p, S)
нач
ввод x1, x2, x3, y1, y2, y3
A: = $sqrt((x_1 - x_2) ** 2 + (y_1 - y_2) ** 2)$
B: = $sqrt((x_2 - x_3) ** 2 + (y_2 - y_3) ** 2)$
C: = $sqrt((x_3 - x_1) ** 2 + (y_3 - y_1) ** 2)$
p: = $(A + B + C)/2$
S: = $sqrt(p * (p - A) * (p - B) * (p - C))$
вывод S
кон

Задача 7.3

1. Формулировка задачи

Требуется составить алгоритм программы, которая по введенному количеству секунд определяет количество суток, часов, минут и секунд. Например,

4000 секунд = 0 суток 1 час 6 минут 40 секунд

2. Математическая постановка задачи

Для того, чтобы вычислить количество суток, часов, минут и секунд необходимо использовать операторы div (возвращает целую часть от деления) и mod (возвращает остаток от деления).

1 минута – 60 секунд, 1 час – 60 минут, 1 сутки – 24 часа.

3. Выбор переменных программы

Из приведенного выше условия задачи определяем следующие переменные:

- исходные данные – количество секунд Seconds;
- результат – количество дней (Days), количество часов (Hours), количество минут (Minutes).

4. Блок-схема алгоритма

Составим алгоритм решения данной задачи в виде блок-схемы (рис. 7.4) и на алгоритмическом языке.

```
алгВремя (аргцел Seconds, резвещ Days, Hours, Minutes)
дано | Seconds > 0
нач
ввод  Seconds
      Days:= div (Seconds, 60*60*24)
      Seconds:= mod (Seconds, 60*60*24)
      Hours:= div (Seconds, 60*60)
      Seconds:= mod (Seconds, 60*60)
      Minutes:= div (Seconds, 60)
вывод  Days, Hours, Minutes, Seconds
кон
```

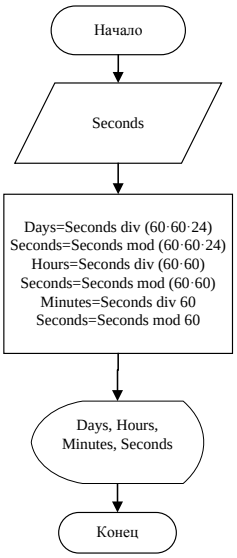


Рис. 7.4. Блок-схема алгоритма к задаче 7.3

Индивидуальные задания

Составить блок-схему алгоритма и программу на алгоритмическом языке для решения задач, согласно варианту.

1. Лошадиный барышник на ярмарке купил лошадь за m рублей, а продал за n. Через некоторое время он купил ту же лошадь за h рублей, а продал за q. Каков его барыш?
2. Скорый поезд вышел из Москвы в Санкт-Петербург и шел без остановок со скоростью v км/ч. Другой поезд вышел ему навстречу из Санкт-Петербурга и тоже шел без остановок, но со скоростью g км/ч. На каком расстоянии друг от друга будут поезда за t часов до встречи?
3. У n хозяев по n кошек, каждая кошка съедает по n мышей, каждая мышь съедает по n колосьев ячменя, из каждого колоса может вырасти по n мер зерна. Сколько мер зерна сохраняется благодаря этим кошкам?
4. Между городами A и BS километров. Их них выехали два велосипедиста и со скоростью v км/ч каждый помчались навстречу друг другу. Вместе с первым велосипедистом из города A стартовала муха со скоростью g км/ч. Встретившись с велосипедистом из города B, муха развернулась и полетела к первому, встретившись с ним, опять полетела ко второму Когда велосипедисты съехались и остановились, муха угомонилась и села одному из них на голову. Сколько километров пролетела муха?
5. Пасли ребята коней. Если пересчитать ноги коней и детей, получится n, а если головы, то m. Сколько было ребят и сколько коней?

6. В жаркий день n косцов выпили бочонок кваса за m часов. Сколько косцов выпьют такой же бочонок кваса за t часов?

7. Вол съел копну за t, конь – за m, коза – за n часов. За сколько времени вол, конь и коза – съедят ту копну вместе?

8. Веселый француз пришел в трактир с неизвестною суммой своего богатства, занял у хозяина столько денег, сколько у себя имел; из сей суммы издержал n рублей. С остатком пришел в другой трактир, где опять, занявши столько, сколько имел, издержал в оном также n рублей; то же учинил в третьем и четвертом трактирах. По выходе из четвертого не имел уже ничего. С какой суммой пришел он в первый трактир?

9. Селекционер вывел новый сорт зерновой культуры и снял с опытной делянки k кг семян. Посеяв 1 кг семян, можно за сезон собрать p кг семян. Через сколько лет селекционер сможет засеять новой культурой поле площадью s га, если норма высева n кг/га?

10. Из круга радиуса r вырезан прямоугольник, большая сторона которого равна a. Найти максимальный радиус круга, который можно вырезать из полученного прямоугольника?

11. Владелец автомобиля приобрел новый карбюратор, который экономит *50%* топлива, новую систему зажигания, которая экономит *30%* топлива, и поршневые кольца, экономящие *20%* топлива. Верно ли, что его автомобиль теперь сможет обходиться совсем без топлива? Найти фактическую экономию для произвольно заданных сэкономленных процентов.

12. Студент съедает торт за пять минут, преподаватель – за полчаса, а заведующий отделением – за час. За какое время они съедят торт вместе? (4 минуты)

13. Старинная задача. Летела стая гусей, а навстречу им летит один гусь и говорит: Здравствуйте, сто гусей!» А передний старый гусь ему и отвечает: «Нет, нас не сто гусей! Вот если б нас было еще столько, да еще полстолька, да еще четверть столько, да ты, гусь, то было бы сто гусей, а теперь … Вот и рассчитай-ка, сколько нас?» (36 гусей)

14. Старинная народная задача. Крестьянка пришла на базар продавать яйца. Первая покупательница купила у нее половину всех яиц и еще пол-яйца. Вторая покупательница приобрела половин оставшихся яиц и еще пол-яйца. Третья купила всего одно яйцо. После этого у крестьянки не осталось ничего. Сколько яиц она принесла на базар? (7)

15. В 5 мисках – 100 орехов. В первой и второй мисках вместе 52 ореха. Во второй и в третьей – 43, в третьей и четвертой – 34, в четвертой и пятой – 30. Сколько орехов в каждой миске? (27, 25, 18, 16 и 14)

Практическая работа №3.
Построение Алгоритмов с ветвлением

В линейной программе все операторы выполняются последовательно, один за другим. Для того, чтобы в зависимости от исходных данных обеспечить выполнение разных последовательностей операторов, применяются алгоритмы условного ветвления.

Если рассматривается передача управления на одну из двух ветвей вычислений, то применяется фрагмент блок-схемы алгоритма с ветвлением, приведенный на рис. 8.1.

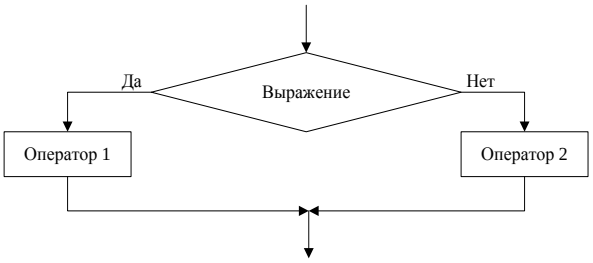


Рис. 8.1. Фрагмент блок-схемы алгоритма с ветвлением

Сначала вычисляется выражение, если оно принимает значение «истина», выполняется Оператор_1, иначе Оператор_2. После этого управление передается на оператор, следующий за условным.

Например, фрагмент блок-схемы алгоритма начисления стипендии студентам, закончившим семестр на оценки «хорошо» и «отлично» будет выглядеть следующим образом (рис. 8.2):

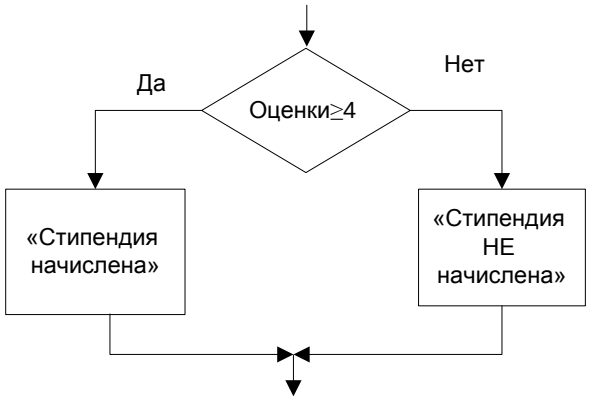


Рис. 8.2. Пример фрагмента блок-схемы алгоритма с ветвлением

Однако одна из ветвей алгоритма может отсутствовать. Например, фрагмент блок-схемы алгоритма вывода сообщения «Пример, мир», если программа запущена, приведен на рис. 8.3.

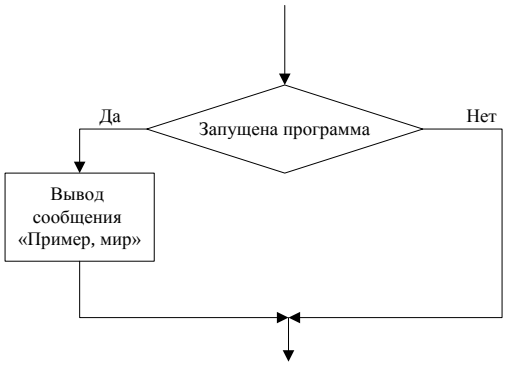


Рис. 8.3. Пример фрагмента блок-схемы алгоритма
при отсутствии одной ветви

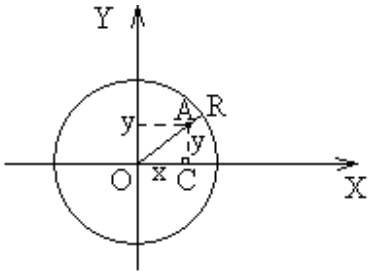
Рассмотрим ряд задач с использованием алгоритмов с ветвлением.

Задача 8.1

1. Формулировка задачи

Составить алгоритм решения задачи, который определяет по введенным координатам, попадает ли заданная точка в окружность с центром в точке O (0;0) и заданным радиусом R.

2. Математическая постановка задачи



Заданная точка имеет координаты x, y. Рассмотрим треугольник AOC. ∠OCA=90º, OC=x, AC=y, следовательно, по теореме Пифагора:
AO^2 (гипотенуза)= OC^2+AC^2 ($AO^2=x^2+y^2$).
Поэтому условие принадлежности точки окружности можно записать в виде:$x^2+y^2 \le R^2$.

3. Выбор переменных программы

Из приведенного выше решения определяем следующие переменные:

- исходные данные – радиус окружности (*R*) и координаты точки (*x* и *y*);
- результат – сообщение «В окружности» или «Вне окружности».

Так как радиус окружности и координаты точки могут принимать любые значения (2; 2,5; 3,75), все переменные для данной задачи определены как действительные числа.

4. Блок-схема алгоритма

Составим алгоритм решения данной задачи в виде блок-схемы (рис.8.4) и на алгоритмическом языке.

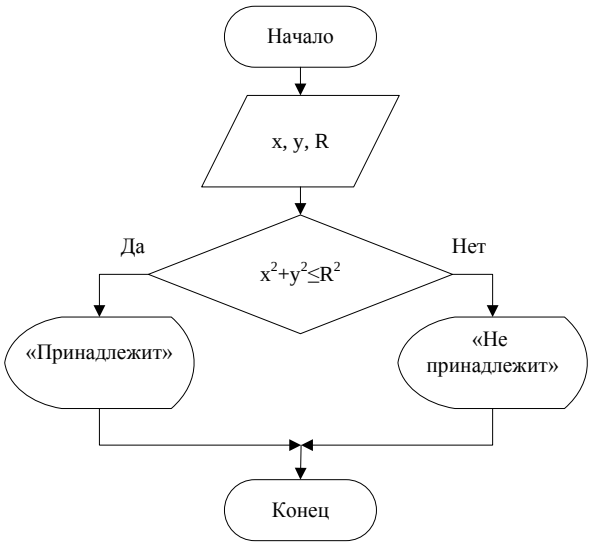


Рис. 8.4. Алгоритм в виде блок-схемы к задаче 8.1

алгОкружность (**арг**вещ*x, y, R*)
нач
ввод x, y, R
если*R*>= ***sqrt***(*x***x*+*y***y*)
товывод"Принадлежит"
иначевывод"Не принадлежит"
все
кон

Задача 8.2

1. Формулировка задачи

Составить алгоритм решения задачи, который по введенным координатам точки определяет, попадает ли эта точка в заштрихованную область (рис. 8.5).

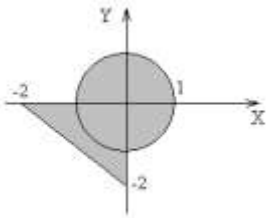


Рис. 8.5. Графическая интерпретация условия задачи 8.2

2. Математическая постановка задачи

Запишем условия попадания точки в область в виде формул. Область можно описать как круг, пересекающийся с треугольником. Точка может попадать либо в круг, либо в треугольник, либо в их общую часть. Условие попадание точки в круг запишем как $x^2 + y^2 \leq 1^2$, т.к. радиус окружности в соответствии с графической интерпретацией равен 1.

Так как треугольник расположен в третьей четверти координатной плоскости, следовательно, для него выполняется условие $\begin{cases} x \leq 0 \\ y \leq 0 \end{cases}$. Выведем уравнение прямой линии, ограничивающей треугольник снизу.

Известно, что прямая линия проходит через две точки: (-2;0) и (0;-2). Требуется составить уравнение прямой линии.

Общий вид уравнения прямой $y = kx + b$, где $k, x,$ – фиксированные числа.

Искомая прямая $y = kx + b$ с пока неизвестными коэффициентами $k, x,$ проходит через точки (-2;0) и (0;-2), а, значит, выполняются равенства $0 = k \cdot (-2) + b$ и $-2 = k \cdot 0 + b$, что можно записать в виде системы: $\begin{cases} 0 = k \cdot (-2) + b \\ -2 = k \cdot 0 + b \end{cases}$ или $\begin{cases} 0 = k \cdot (-2) + b \\ -2 = b \end{cases}$.

Решив систему относительно неизвестных *k и b* , мы найдем уравнение прямой.

Подставим значение $b = -2$ в уравнение $0 = k \cdot (-2) + b$:

$$-2k = -b$$
$$-2k = -(-2)$$
$$-2k = 2$$
$$k = -1$$

Таким образом, искомое уравнение прямой имеет вид $y = (-1) \cdot x - 2$. Так как заштрихованная область расположена выше прямой линии, ограничивающей треугольник снизу, то попадание точки в треугольник запишем как $\begin{cases} x \le 0 \\ y \le 0 \\ y \ge -x - 2 \end{cases}$. Следовательно, заштрихованная область определена следующими условиями $x^2 + y^2 \le 1^2$ или $\begin{cases} x \le 0 \\ y \le 0 \\ y \ge -x - 2 \end{cases}$.

3. Выбор переменных программы

Из приведенного выше решения определяем следующие переменные:

- исходные данные – координаты точки (*x* и *y*);
- результат – сообщение «Принадлежит» или «Не принадлежит».

Так как координаты точки могут принимать любые значения (2; 2,5; 3,75), все переменные определяем как действительные числа.

4. Блок-схема алгоритма

Составим алгоритм решения данной задачи в виде блок-схемы (рис. 8.6) и на алгоритмическом языке.

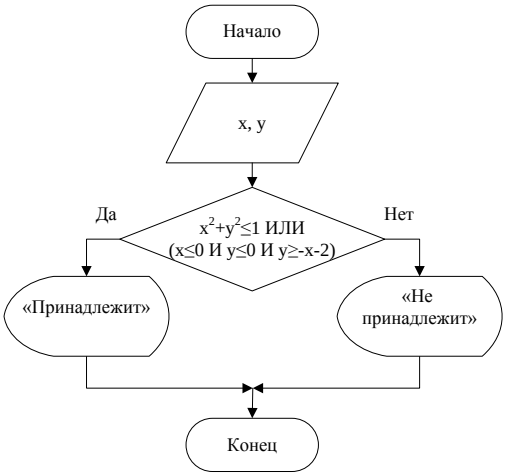


Рис. 8.6 Алгоритм в виде блок-схемы к задаче 8.2

алгОкружность(**аргвещ***x, y*)
нач

ввод x, y
если1>= *x***x*+*y***y***или** (*x*<=**0и***y*<=**0и***y*>=-*x*-**2**)
тоВывод"Принадлежит"
иначеВывод"Не принадлежит"
все
кон

Задача 8.3

1. Формулировка задачи

Составить алгоритм расчета значений кусочной функции для любого заданного *x* в виде блок-схемы и программу на алгоритмическом языке

$$y = \begin{cases} x - 5, \; x \le 0, \\ \dfrac{\ln x}{x^2 + x}, \; 0 < x < 3, \\ \sqrt{x+5}, \; x \ge 3. \end{cases}$$

2. Математическая постановка задачи

Для расчета значения кусочной функции *y(x)* необходимо задать значение *x*. Если *x* принимает значение в диапазоне (-∞, 0], то значение кусочной функции рассчитывается по формуле *y = x*-5. Если *x* принимает значение, лежащее в диапазоне]0, 3[, то значение кусочной функции рассчитывается по формуле $y = \frac{\ln x}{x^2+x}$. Если *x* принимает значение в диапазоне [3, +∞), то значение кусочной функции рассчитывается по формуле $y = \sqrt{x+5}$.

3. Выбор переменных программы

Из приведенного выше решения определяем следующие переменные:

- исходные данные – значение *x*;
- результат – значение кусочной функции *y*.

Выбор переменных программы: аргумент и значение функции могут принимать любые значения (2; 2,5; 3,75), все переменные определяем как действительные числа.

4. Блок-схема алгоритма

Составим алгоритм решения данной задачи в виде блок-схемы (рис. 8.7.) и на алгоритмическом языке.

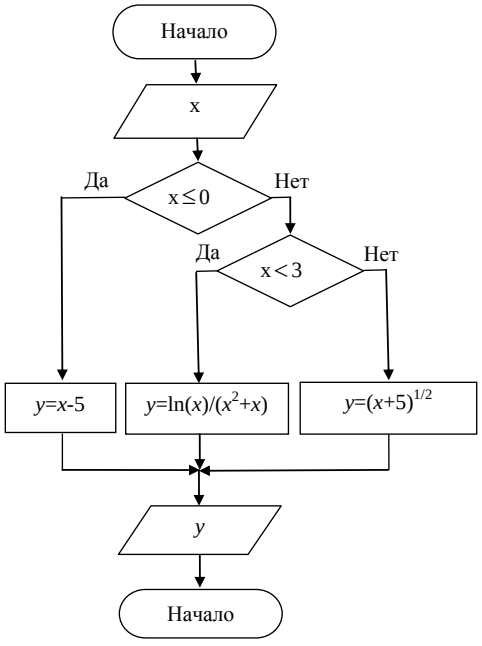


Рис. 8.7. Алгоритм в виде блок-схемы к задаче 8.3

```
алгКусочная_функция (аргцелx, резвещy)
нач
ввод  x
выбор
приx<= 0: y:=x-5
приx>0иx<3 : y:=ln(x)/(x*x+x)
иначеy:=sqrt(x+5)
все
выводy

    кон
```

Оператор выбора

Оператор *выбора* предназначен для разветвления процесса вычислений на несколько направлений. Управление передается первому оператору из списка, помеченного константным выражением, значение которого совпало с вычисленным, после этого последовательно выполняются все остальные ветви.

Задача 8.4. Составить программу, реализующую действия простейшего калькулятора.

Выбор переменных программы

Определяем следующие переменные:

- исходные данные – операнды (*a* и *b*) и знак операции (*op*);
- результат (*res*).

Операнды определяем как действительные числа, а знак операций – как символ.

```
алгКалькулятор (аргвещa,b, симвop,резвещres)
нач
вводa,b,op
выбор
приop='+': res:= a+b
приop='-': res:= a-b
приop='*': res:= a*b
приop='/': res:= a/b
все
выводres

    кон
```

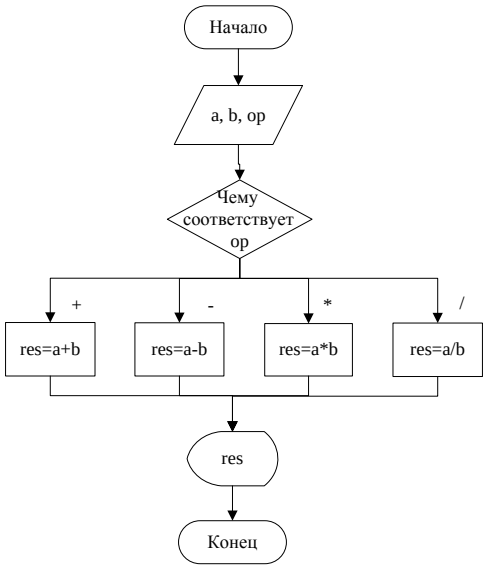


Рис. 8.8. Блок схема к задаче 8.4

Индивидуальные задания

1. Составить алгоритм расчета значений кусочной функции для любого заданного x в виде блок-схемы

1. $y=\begin{cases} x^2,\ x\le 0,\\ \cos x,\ 0<x<5,\\ \sqrt{x},\ x\ge 5. \end{cases}$

2. $y=\begin{cases} x^2-4,\ x\le 0,\\ \cos x+5,\ 0<x<5,\\ \sqrt{x}+x^2,\ x\ge 5. \end{cases}$

3. $y=\begin{cases} e^x+x^2,\ x\le 0,\\ \cos x,\ 0<x<5,\\ 3x-\sqrt{x},\ x\ge 5. \end{cases}$

4. $y=\begin{cases} \ln|x|,\ x<0,\\ \cos x,\ 0\le x<3,\\ x^{-5/6},\ x\ge 3. \end{cases}$

5. $y=\begin{cases} e^{-x},\ x\le 0,\\ \sin 2x,\ 0<x<5,\\ \sqrt{x},\ x\ge 5. \end{cases}$

6. $y=\begin{cases} \sin x+x^2,\ x\le 0,\\ \cos x,\ 0<x<5,\\ e^x+\sqrt{x},\ x\ge 5. \end{cases}$

7. $y=\begin{cases} \sqrt[3]{x^2},\ x<0,\\ \cos x,\ 0\le x<5,\\ x+3,\ x\ge 5. \end{cases}$

8. $y=\begin{cases} e^{-2x},\ x\le 0,\\ \cos(x-\frac{\pi}{6}),\ 0<x<5,\\ \sqrt{x},\ x\ge 5. \end{cases}$

9. $y=\begin{cases} x^4,\ x\le \pi,\\ \cos x,\ \pi<x<5,\\ \sqrt{x},\ x\ge 5. \end{cases}$

10. $y=\begin{cases} e^x,\ x\le -5,\\ 0,\ -5<x<5,\\ e^{-x+1},\ x\ge 5. \end{cases}$

11. $y=\begin{cases} \sin(x-5),\ x\le 0,\\ \frac{\cos 2x}{x^2+1},\ 0<x<3,\\ \sin(5+\sqrt{x}),\ x\ge 3. \end{cases}$

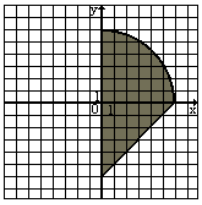
12. $y=\begin{cases} \sin 2x,\ x\le -2\pi,\\ e^x+e^{-x},\ -2\pi<x<5,\\ \cos 2x,\ x\ge 5. \end{cases}$

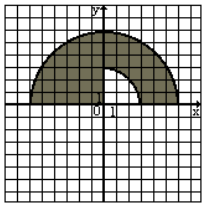
13. $y=\begin{cases} x-5,\ x\le 0,\\ 2\cos x,\ 0<x<10,\\ 5+\sqrt{x},\ x\ge 10. \end{cases}$

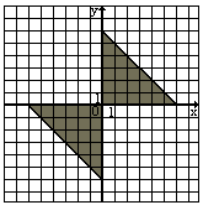
14. $y=\begin{cases} x^2,\ x\le 6,\\ e^{-x},\ 6<x<25,\\ 4,\ x\ge 25. \end{cases}$

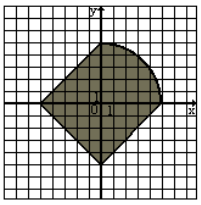
15. $y=\begin{cases} x^2,\ x\le -\frac{3}{2},\\ \cos(x),\ -\frac{3}{2}<x<5,\\ \sqrt{|\sin x|},\ x\ge 5. \end{cases}$

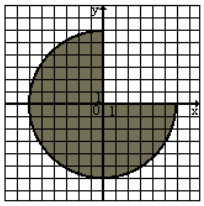
2. Составить алгоритм решения задачи, который по введенным координатам точки определяет, попадает ли эта точка в заштрихованную область

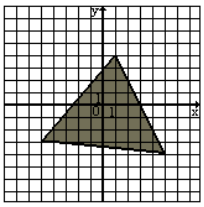
1. 

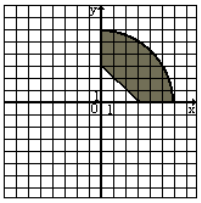
6. 

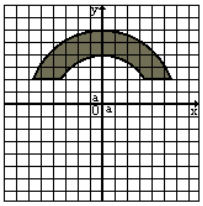
11. 

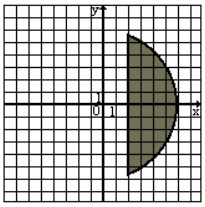
2. 

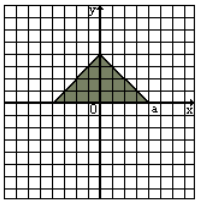
7. 

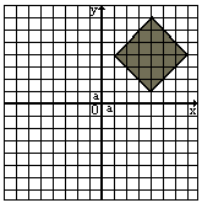
12. 

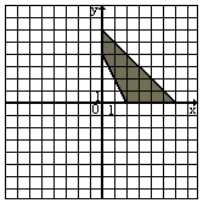
3. 

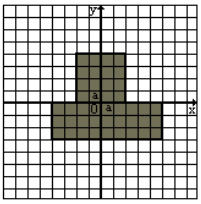
8. 

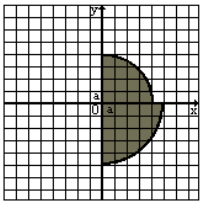
13. 

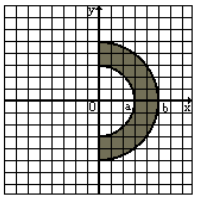
4. 

9. 

14. 

5. 

10. 

15. 

3. Составить алгоритм решения задачи:

1. Задан четырехугольник со сторонами *A, B, C, D*. Определить, является ли он параллелограммом или ромбом.
2. На плоскости задан отрезок координатами своих концов. Определить, имеет ли он общие точки с осями координат.
3. На плоскости задан отрезок координатами своих концов. Определить, принадлежит ли отрезок полностью первой четверти координатной плоскости.
4. На плоскости задан отрезок координатами своих концов. Определить, параллелен ли он осям координат.
5. На плоскости задана окружность с центром в точке (*X1, Y1*), радиусом *R*. Определить, принадлежит ли точка с координатами *X, Y* заданному кругу, лежит на ее границе или вне ее.
6. На плоскости задана точка с координатами *X, Y*. Определить, какой четверти принадлежит точка.
7. Даны три отрезка *A, B, C*. Определить, существует ли треугольник со сторонами, равными данным отрезкам.
8. Даны три отрезка *A, B, C*. Определить, является ли треугольник, со сторонами, равным отрезкам, остроугольным, прямоугольным или тупоугольным.
9. Задан треугольник координатами своих вершин. Определить, является он остроугольным, прямоугольным или тупоугольным.

10. Задана точка на плоскости и два прямоугольника со сторонами, параллельными осям координат. Определить, принадлежит точка только первому прямоугольнику, только второму или обоим вместе.

11. На плоскости заданы точка, круг радиусом *R* с центром в начале координат и описанный вокруг него квадрат со сторонами, параллельными осям координат. Определить, находится точка одновременно и в круге, и в квадрате, вне их или только в круге.

12. На плоскости задана точка, не находящаяся на осях координат. Если точка лежит в круге с центром в начале координат и радиусом *R*, определить, какой четверти координатной плоскости принадлежит точка.

13. Даны две точки *A(x1,y1)* и *B(x2,y2)*. Составить программу, определяющую, которая из точек находится ближе к началу координат.

14. На плоскости расположены три точки *a, b, c*. Определить, какая из точек *b* или *c* расположены ближе к *a*.

15. Заданы размеры *A, B* прямоугольного отверстия и размеры *x, y, z* кирпича. Определить, пройдет ли кирпич через это отверстие, параллельно сторонам отверстия.

4. Составить алгоритм с оператором выбора в виде блок-схемы и на алгоритмическом языке.

1. Вычислить значение функции, если *x* – натуральное число; *a, b, c* – действительные числа:

$$y=\begin{cases} a+bx+cx^2, & x=1; \\ a\cdot\sin\left(xb^2\right), & x=2; \\ \sqrt{|a+bx|}, & x=3; \\ a\cdot\ln\left|bx+cx^2\right|, & x=4; \\ e^{-ax}+\sin bx+cx, & x=5. \end{cases}$$

В противном случае (если *x* не входит в заданный диапазон) вывести сообщение.

2. По введенному номеру времени года (1 – зима, 2 – весна, 3 – лето, 4 – осень) выдать соответствующие этому времени года месяцы, количество дней в каждом из месяцев.

3. В старояпонском календаре был принят 12-летний цикл. Годы внутри цикла носили названия животных: крысы, коровы, тигра, зайца, дракона, змеи, лошади, овцы, обезьяны, курицы, собаки и свиньи. Определить по номеру некоторого года его название по старояпонскому календарю. (Справка: 1996 г. – год Крысы – начало очередного цикла.)

4. По введенному номеру единицы измерения (1 – дециметр, 2 – километр, 3 – метр, 4 – миллиметр, 5 – сантиметр) и длине отрезка *L* выдать соответствующее значение длины отрезка в метрах.

5. По введенному числу от 1 до 12 (номеру месяца) выдать все приходящиеся на этот месяц праздничные выходные дни (например, если введено число 1, то должно получиться 1 января – Новый год, 7 января – Рождество).

6. При вводе знака препинания выдать на экран дисплея его название. Например, на ввод точки выдает текст: «Точка».

7. По введенному номеру единицы измерения (1 – килограмм, 2 – миллиграмм. 3 – грамм, 4 – тонна, 5 – центнер) и массе M выдать соответствующее значение массы в килограммах.

8. Пусть элементами равностороннего треугольника являются:

– сторона *a*;

– площадь S;
– высота h;
– радиус вписанной окружности r;
– радиус описанной окружности R.

По заданному номеру и значению соответствующего элемента вычислить значение всех остальных элементов треугольника.

9. Определить подходящий возраст кандидатуры для вступления в брак, используя следующее соображение: возраст девушки равен половине возраста мужчины плюс 7, возраст мужчины определяется соответственно как удвоенный возраст девушки минус 14.

10. По заданной дате определить время года. Учесть корректность введенной даты.

11. Ввести натуральное число в десятичном представлении, а на выходе вывести это же число в десятичном представлении и на естественном языке.

Например, 7 семь; 204 двести четыре; 52 пятьдесят два.

12. Вычислить порядковый номер дня в невисокосном году по заданным числу и месяцу.

13. Требуется ввод времени дня и, в зависимости от введенного значения, вывести сообщение «Доброе утро!», «Добрый день!», «Добрый вечер!» или «Спокойной ночи».

14. Дано натуральное число. Определить остаток от деления на 4 и вывести его в текстовом виде (ноль, один, два, три).

15. Дано неотрицательное число k, не превышающее десяти тысяч. Напечатать фразу «k ворон» русскими словами. (Пример: если k=23, то должно быть напечатано «двадцать три вороны»; если k=3651, то «три тысячи шестьсот пятьдесят одна ворона»).

Практическая работа №4.
построение Циклических алгоритмов

Цикл – это многократно повторяющаяся последовательность операторов. В качестве примера цикла можно привести изменение времени года (зима, весна, лето, осень).

Операторы цикла используются для организации многократно повторяющихся вычислений. Любой цикл состоит из тела цикла, т.е. тех операторов, которые выполняются несколько раз, начальных установок, модификации параметра цикла и проверки условия выполнения цикла.

Цикл называется ***детерминированным,*** если число повторений тела цикла заранее известно или определено. Цикл называется ***итерационным,*** если число повторений тела цикла заранее неизвестно, а зависит от значений параметров (некоторых переменных), участвующих в вычислениях. Один проход цикла называется итерацией.

Существуют три вида циклов.

1. Если проверка условия выполняется после тела цикла – цикл с постусловием.
2. Если проверка условия выполняется на каждой итерации либо до тела цикла – это цикл с предусловием.
3. Если число повторений тела цикла заранее известно или определено – цикл со счетчиком.

Цикл с постусловием (итерационный)

Тело цикла с постусловием всегда выполняется хотя бы один раз, после чего проверяется необходимость следующего его выполнения. Фрагмент алгоритма цикла с постусловием приведен на рис.9.1.

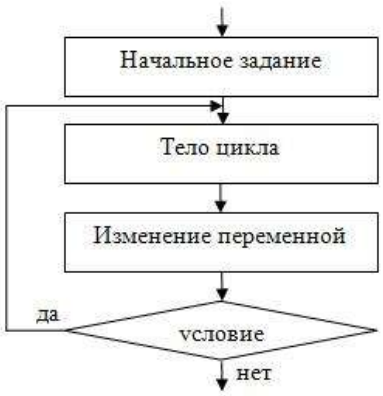


Рис. 9.1. Фрагмент алгоритма цикла с постусловием

Цикл с предусловием (итерационный)

Проверка необходимости выполнения цикла с предусловием делается до тела цикла, поэтому, возможно, что он не выполнится ни разу (рис. 9.2).

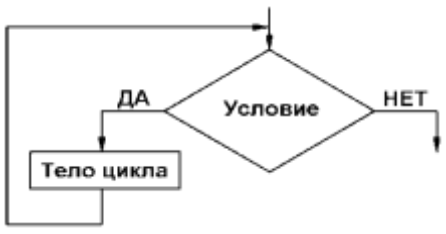


Рис. 9.2. Фрагмент алгоритма цикла с предусловием

Цикл с заданными параметрами (детерминированный, со счетчиком)

Переменные, изменяющиеся в теле цикла и используемые при проверке условия выполнения, называются параметрами цикла. Целочисленные параметры цикла, изменяющиеся с постоянным шагом при каждой итерации, называются счетчиками цикла.

Начальные установки могут явно не присутствовать в программе, их смысл состоит в том, чтобы до входа в цикл задать значения переменным, которые в нем используются (рис. 9.3).

Рис. 9.3. Фрагмент алгоритма цикла с параметром

Существенно то, что проверка условия всегда выполняется в начале цикла. Это значит, что тело цикла может ни разу не выполниться, если условие выполнения сразу будет ложным. Модификации выполняются после каждой итерации цикла и служат обычно для изменения параметров цикла. Простой или составной оператор представляет собой тело цикла.

Задача 9.1.

1. Формулировка задачи

Составить алгоритм расчета факториала N! с использованием различных видов цикла в виде блок-схемы и программы на алгоритмическом языке.

2. Математическая постановка задачи

Факториал числа N (обозначается N!) – произведение всех натуральных чисел до N включительно:

$$N!=1\cdot 2\cdot 3\cdot \ldots \cdot N$$

По определению полагают 0! = 1. Факториал определён только для целых неотрицательных чисел.

3. Выбор переменных программы

Из приведенного выше решения определяем следующие переменные:

- исходные данные – значение *N*;
- результат – значение *factorial*.

Выбор переменных: аргумент и значение функции могут принимать целые неотрицательные значения.

4. Блок-схема алгоритма

Составим алгоритм решения данной задачи в виде блок-схем (рис.9.4) и на алгоритмическом языке.

алгфакториал1(аргцел n, **резцел**factorial)
нач
цел i
i:=1

```
        factorial:=1;
        нц
    factorial := factorial * i
    i:=i+1
кцпри i>n
вывод factorial
кон
```

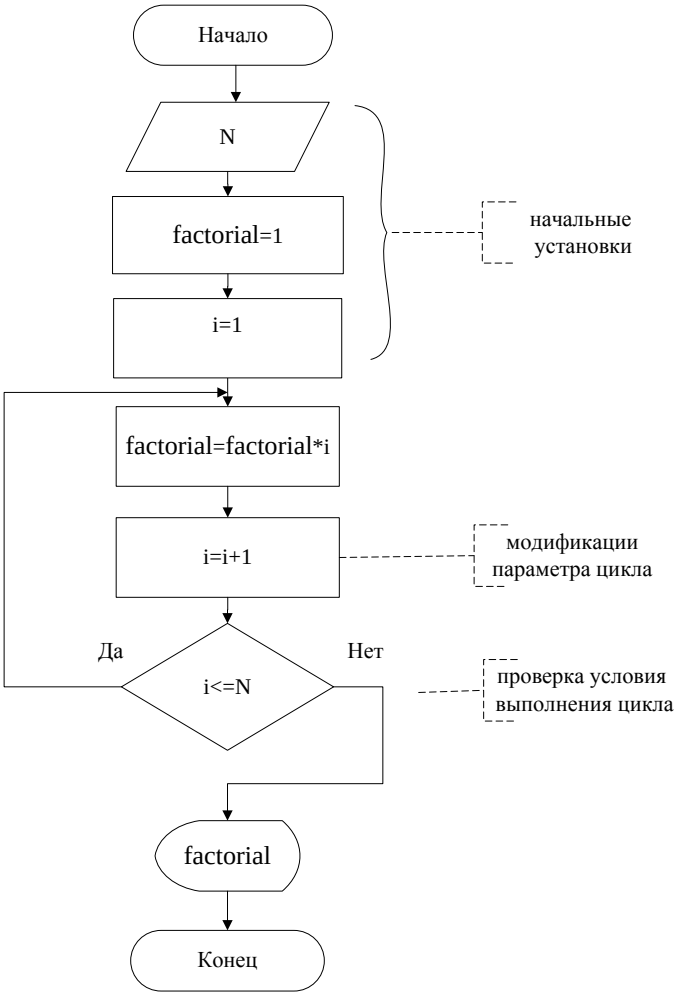


Рис. 9.4. Блок-схема цикла с постусловием к задаче 9.1

Составим блок схему для задачи 9.1 с использованием цикла с предусловием, рис. 9.5.

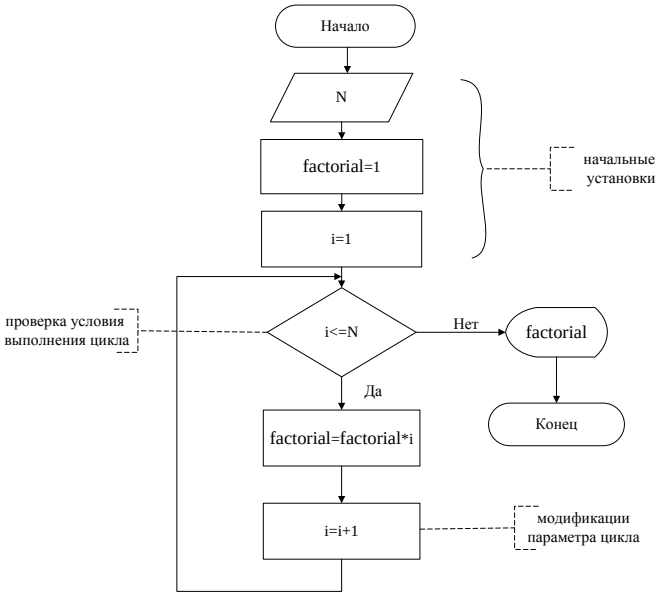


Рис. 9.5. Блок схема цикла с предусловием к задаче 9.1

алгфакториал2(**аргцел** n, **резцел**factorial)
нач
цел i
i:=1
factorial:= 1;
нцпока i<=n
factorial := factorial * i
i:=i+1
кц
вывод factorial
кон

На рис.9.6 приведен пример алгоритма расчета факториала N! с использованием цикла с параметром в виде блок-схемы.

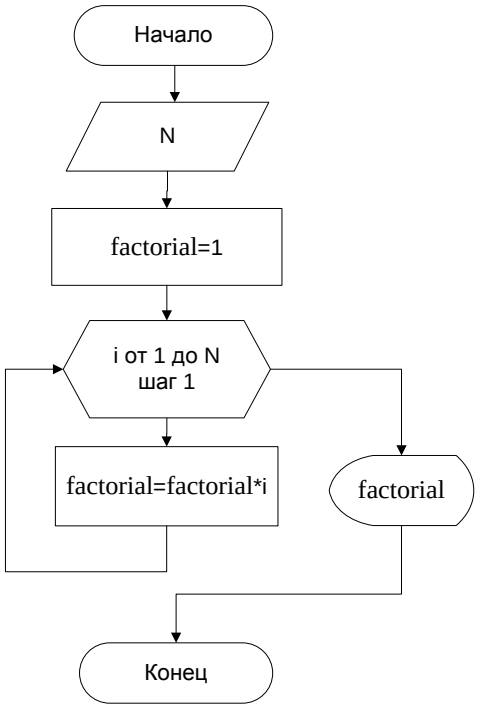


Рис.9.6. Алгоритм расчета факториала с использованием цикла с параметром в виде блок-схемы

```
алгфакториал3(аргцел n, резцел factorial)
нач
        цел i
        factorial := 1;
        нцдля i от 2 до n
          factorial := factorial * i
кц
        выводfactorial
кон
```

Задача 9.2.

1. Формулировка задачи

Составить алгоритм вычисления суммы ряда:

$y = x - \frac{x^2}{2} + \frac{x^3}{3} + \ldots + \frac{(-1)^{n-1} x^n}{n}$ с точностью *e=0.0001* с использованием цикла с постусловием и предусловием в виде блок-схемы и программу на алгоритмическом языке.

2. Математическая постановка задачи

Ряд представляет собой сумму слагаемых, в котором перед каждым слагаемым чередуется знак (перед первым слагаемым знак «+», перед вторым – «-», перед третьим – «+», перед четвертым – «-», и т.д.). Поэтому для его учета в алгоритм накопления суммы вводим переменную, отвечающую за знак – z.

При накоплении суммы, начальное значение суммы обнуляем, т.е. S=0. Первое слагаемое можно представить в виде $\frac{x^1}{1}$, т.е. начальное значение степени x и знаменатель равны 1. Шаг изменения степени и знаменателя совпадает и равен 1. Точность вычисления определяется значением слагаемого.

3. Выбор переменных программы

Из приведенного выше описания определяем следующие переменные:

- исходные данные – значение *x*;
- начальное значение суммы S равно 0 (S=0);
- начальное значение степени x и знаменатель равны 1 (i=1);
- знак перед первым слагаемым – «+» (z=1);
- результат – значение S.

4. Блок-схема алгоритма

Составим алгоритм решения данной задачи в виде блок-схем (рис.9.7, 9.8) и на алгоритмическом языке.

.

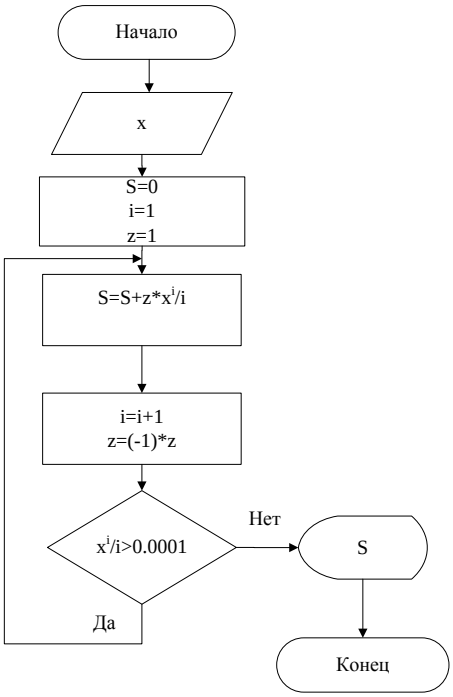


Рис.9.7. Алгоритм вычисления суммы ряда (цикл с постусловием) к задаче 9.2

```
алгСумма1(аргцелi,z, вещx, резвещS)
нач
ввод x
i:=1
z:=1
S:=0
нц
   S:=S+z*x**i/i
   i:=i+1
   z:=(-1)*z
кцпри x**i/i>0.0001
вывод S
кон
```

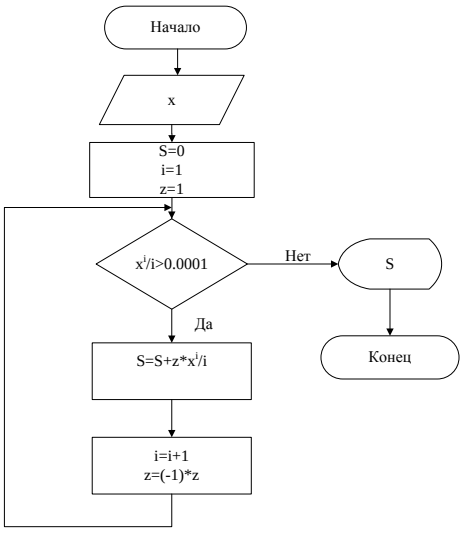


Рис. 9.8. Алгоритм вычисления суммы ряда (цикл с предусловием)

к задаче 9.2

```
алгСумма2(аргцелi,z, вещx, резвещS)
нач
ввод x
i:=1
z:=1
S:=0
нцпока x**i/i>0.0001
   S:=S+z*x**i/i
   i:=i+1
   z:=(-1)*z
кц
выводS
кон
```

Индивидуальные задания

1. Составить алгоритм решения задачи с использованием *цикла с постусловием и предусловием* в виде блок-схемы и программу на алгоритмическом языке.

1-8. Не используя стандартные функции, вычислить с точностью *0,0001*:

1. $y = sh(x) = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \ldots + \frac{x^{2n+1}}{(2n+1)!};$
2. $y = e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \ldots + \frac{x^n}{n!};$
3. $y = \ln(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} + \ldots + \frac{(-1)^{n-1} x^n}{n};$
4. $y = arctg(x) = x - \frac{x^3}{3} + \frac{x^5}{5} + \ldots + \frac{(-1)^n x^{2n+1}}{2n+1};$
5. $y = \sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} + \ldots + \frac{(-1)^n x^{2n+1}}{(2n+1)!};$
6. $y = arctg(-\frac{1}{x}) = -\frac{1}{x} + \frac{1}{3x^3} - \frac{1}{5x^5} + \ldots;$
7. $y = \cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \ldots,$
8. $y = ch(x) = 1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \ldots + \frac{x^{2n+1}}{(2n+1)!}$
9. Дана последовательность, состоящая из дробей: $\frac{1}{1}, \frac{4}{2}, \frac{9}{4}, \frac{16}{8} \ldots$ Какое минимальное количество элементов последовательности нужно сложить, чтобы сумма превысила заданное число S (S>1)?

10. Числа Фибоначчи определяются по формулам: $f_0 = 0,\ f_1 = 1,\ f_k = f_{k-2} + f_{k-1}$. Найти первое число Фибоначчи, большее m (m>0).

11. Числа Фибоначчи определяются по формулам: $f_0 = 0,\ f_1 = 1,\ f_k = f_{k-2} + f_{k-1}$. Вычислить сумму всех чисел Фибоначчи, которые не превосходят M (M>0).

12. Число π вычисляется по формуле Грегори следующим образом: $\pi = 4 \cdot \left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \ldots\right)$, причем, чем больше слагаемых в скобках, тем выше точность вычисления числа π. Определить минимальное количество слагаемых для вычисления π с точностью 0.001.

13. Дана последовательность, состоящая из дробей: $\frac{1}{1}, \frac{4}{2}, \frac{7}{3}, \frac{10}{4} \ldots$ Какое минимальное количество элементов последовательности нужно сложить, чтобы сумма превысила заданное число *S* (*S*>1)?

14. Дана последовательность, состоящая из дробей: $\frac{1}{1}, \frac{3}{2}, \frac{5}{3}, \frac{7}{4} \ldots$ Какое минимальное количество элементов последовательности нужно сложить, чтобы сумма превысила заданное число *S* (*S*>1)?

15. Дана последовательность, состоящая из дробей: $\frac{1}{2}, \frac{3}{4}, \frac{5}{6}, \frac{7}{8} \ldots$ Какое минимальное количество элементов последовательности нужно сложить, чтобы сумма превысила заданное число *S* (*S*>1)?

2. Составить алгоритм решения задачи с использованием *цикла с заданными параметрами* в виде блок-схемы и программу на алгоритмическом языке

1. Дано натуральное число *N*. Вычислить $N! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot N$.
2. Дано натуральное число *N*. Вычислить: $\left(1 + \frac{1}{1^2}\right)\left(1 + \frac{1}{2^2}\right) \ldots \left(1 + \frac{1}{N^2}\right)$.
3. Дано натуральное число *N*. Вычислить: $\frac{1}{2} + \frac{3}{4} + \frac{5}{6} + \ldots + \frac{2n-1}{2n}$.
4. Даны действительное число *a*, натуральное число *N*. Вычислить: *a(a+1)(a+2)(a+3)...(a+N-1).*
5. Даны действительное число *a*, натуральное число *N*.

Вычислить: $\frac{1}{a} + \frac{1}{a(a+1)} + \ldots + \frac{1}{a(a+1)\ldots(a+n)}$.

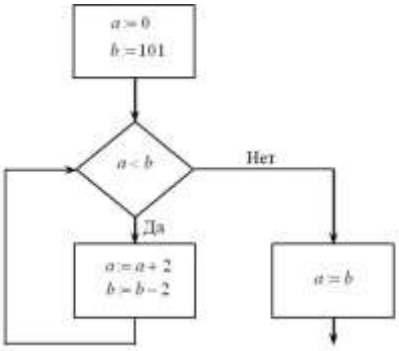
6. Даны действительное число *a*, натуральное число *N*.

Вычислить: $\frac{1}{a} + \frac{1}{a^2} + \frac{1}{a^4} + \ldots + \frac{1}{a^{2n}}$.

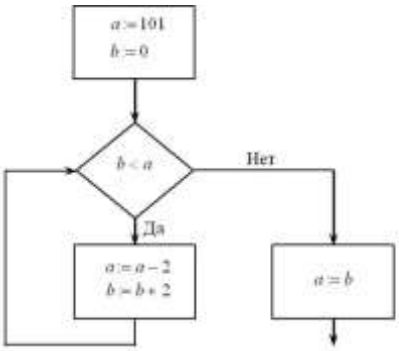
7. Даны действительное число *a*, натуральное число *N*. Вычислить: $a(a-N)(a-2N)\ldots(a-N^2)$.
8. Для натурального *N* найти: $\sin x + \sin^{2!} x + \ldots + \sin^{N!} x$, где x – любое число.
9. Для натурального *N* найти: $1 + \frac{1}{2!} + \frac{1}{3!} + \ldots + \frac{1}{N!}$.
10. Для натурального *N* найти: $\sin\frac{1}{N} + \sin\frac{1}{2N} + \ldots + \sin\frac{1}{N^2}$.
11. Для натурального *N* найти: $\frac{1!}{\cos N} + \frac{2!}{\cos N} + \ldots + \frac{N!}{\cos N}$.
12. Для натурального *N* найти: $\frac{a}{(1+1)!} + \frac{a}{(2+1)!} + \ldots + \frac{a}{(N+1)!}$, где a – любое число.
13. Дано натуральное число *N*. Вычислить: $y = \cos x + \cos x^2 + \cos x^3 + \ldots + \cos x^n$.
14. Вычислить *y*, для *n>1. y=1! + 2! + 3! + ... + n!.*
15. Составить программу для вычисления суммы ряда $\sum_{n=1}^{10} \frac{\sqrt[4]{n}}{n+1}$.

Условие выполнения цикла [14]

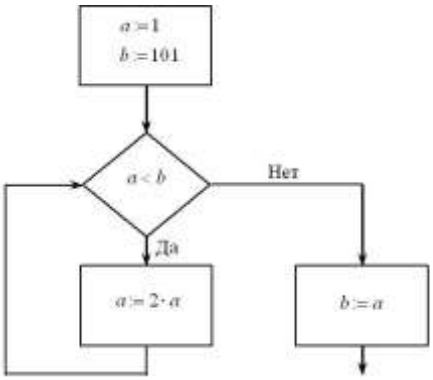
1. № 2001. Запишите значение переменной a после выполнения фрагмента алгоритма:



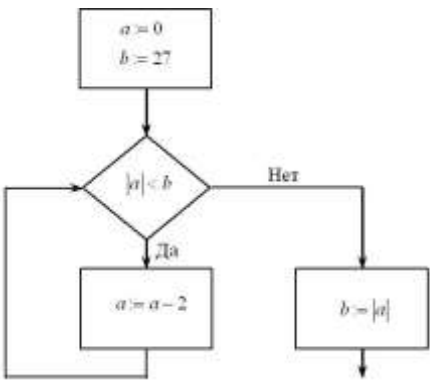
2. № 2002. Запишите значение переменной a после выполнения фрагмента алгоритма:



3. № 2006. Запишите значение переменной a после выполнения фрагмента алгоритма:



4. № 2008. Запишите значение переменной b после выполнения фрагмента алгоритма:



5. № 7753. Запишите число, которое будет напечатано в результате выполнения программы.

алг

```
нач
цел s, n
s := 47
n := 1
нцпока s > 0
   s := s - 9
   n := n + 4
кц
выводn
кон
```

6. № 7780. Запишите число, которое будет напечатано в результате выполнения программы.

```
алг
нач
цел s, n
s := 42
n := 1
нцпока s > 0
   s := s - 5
   n := n + 3
кц
выводn
кон
```

7. № 7919. Запишите число, которое будет напечатано в результате выполнения программы.

```
 алг
нач
цел n, s
n := 1
s := 0
нцпока n <= 100
   s := s + 30
   n := n * 3
кц
вывод s
кон
```

8. № 7984. Запишите число, которое будет напечатано в результате выполнения программы.

```
 алг
нач
   цел n, s
   n := 1
   s := 0
   нцпока n <= 300
      s := s + 30
      n := n * 3
   кц
   вывод s
кон
```

9. № 8096. Запишите число, которое будет напечатано в результате выполнения программы.

```
алг
нач
   цел n, s
   s := 301
   n := 0
   нцпока s > 0
      s := s - 10
      n := n + 2
   кц
   выводn
кон
```

10. № 8656. Запишите число, которое будет напечатано в результате выполнения программы.

```
алг
нач
   цел s, n
   s := 78
   n := 0
   нцпока n < 12
      s := s - 8
      n := n + 2
   кц
   вывод s
кон
```

11. № 9160. Запишите число, которое будет напечатано в результате выполнения программы.

```
алг
нач
   цел s, n
   s := 56
   n := 0
   нцпока n < 15
      s := s - 6
      n := n + 3
   кц
   вывод s
кон
```

12. № 9192. Определите, что будет напечатано в результате выполнения программы.

```
алг
нач
   цел n, s
   n := 1
   s := 0
   нцпока n <= 650
      s := s + 20
      n := n * 5
   кц
   вывод s
кон
```

13. № 9300. Определите, что будет напечатано в результате выполнения программы.

```
алг
нач
```

```
   цел n, s
   n := 1
   s := 0
   нцпока n <= 300
      s := s + 30
      n := n * 5
   кц
   вывод s
кон
```

14. № 9359. Запишите число, которое будет напечатано в результате выполнения следующей программы.

```
алг
нач
цел n, s
  n := 0
  s := 0
нцпока s < 111
   s := s + 8
   n := n + 2
кц
выводn
кон
```

15. № 9758. Запишите число, которое будет напечатано в результате выполнения следующей программы.

```
 алг
нач
  цел n, s
  n := 0
  s := 0
  нцпока 3*s < 111
    s := s + 8
    n := n + 2
  кц
  выводn
кон
```

Дополнительные задачи

1. Дано натуральное число *n*. Найти сумму первой и последней цифры этого числа.
2. Дано натуральное число *n*. Переставив местами первую и последнюю цифры этого числа, получить новое число.
3. Даны два натуральных числа *m* и *n* ($m \le 9999, n \le 9999$). Проверить, есть ли в записи числа *m* цифры, совпадающие с цифрами в записи числа *n*.
4. Даны натуральные числа *n, k*. Проверить, есть ли в записи числа n^k (не более 10^9) цифра *m*.
5. Среди всех *n*-значных чисел указать те, сумма цифр которых равна данному числу *k*.
6. Найти наибольшую и наименьшую цифры в записи данного натурального числа.
7. Найти на отрезке *[n, m]* минимальное натуральное число, имеющее наибольшее количество делителей.

8. Дано целое число в диапазоне от 0 до 1000000. Проверить присутствуют ли одновременно в записи числа цифры 1 и 5.

9. Дано целое число в диапазоне от 0 до 1000000. Проверить отсутствуют ли одновременно в записи числа цифры 2 и 6.

10. Дано целое число в диапазоне от 0 до 1000000. Проверить присутствуют ли в записи числа хотя бы одна цифра кратная трем.

11. Дано целое число в диапазоне от 0 до 1000000. Проверить отсутствуют ли в записи числа четные цифры.

12. Дано произвольное целое положительное число K ($K \le 10^9$). Вывести цифры этого числа в обратном порядке (например, 5485 => 5845).

13. Дано произвольное целое положительное число K ($K \le 10^9$). Вывести новое число, полученное из K вычеркиванием всех четных цифр (например, 234583 => 353).

14. Дано произвольное целое положительное число K ($K \le 10^9$). Вывести это число без первой и последней цифры (например, 234583 => 3458).

15. Дано произвольное целое положительное число K ($K \le 10^9$). Найти сумму всех четных цифр этого числа.

Задачи на моделирование циклических процессов

1. На заводе собирают прибор из трех блоков. Известно, что среди блоков первого типа встречаются 2% со скрытыми дефектами, среди блоков второго и третьего типа – соответственно 3% и 5% дефектных. С использованием генератора случайных чисел промоделировать сборку 1000 деталей и определить, сколько будет собрано приборов без брака.

2. Составить программу, позволяющую промоделировать опрос 100 человек и на его основании выяснить: переименовать село Папинск в село Маминск или оставить за ним прежнее название. Примечание: Программа должна генерировать ответы самостоятельно с использованием генератора случайных чисел.

3. Имеется две спичечные коробки, в каждой из которых находится по 10 спичек. Случайным образом выбирается коробка и из нее достается одна спичка. Процесс продолжается до тех пор, пока одна из коробок не опустеет. С использованием генератора случайных чисел промоделировать этот процесс, и ответить на вопрос: сколько спичек будет всего сожжено?

4. В детском саду имеется группа детей из 20 человек. Каждому ребенку на утреннике Дед Мороз случайным образом дарит одну из следующих домашних игрушек: зайца, мяч или куклу. С использованием генератора случайных чисел промоделировать этот процесс, и ответить на вопрос: сколько игрушек каждого вида было подарено?

5. Робот находится в центре окружности радиусом 3,5 метра и в каждый момент времени делает шаг (длиной 1 м) в случайном направлении: на север, на юг, на восток или на запад. С использованием генератора случайных чисел промоделировать этот процесс, и ответить на вопрос: хватит ли 12 шагов, чтобы выйти за пределы окружности?

6. Известно, что в среднем из 100 выстрелов солдат А поражает мишень 75 раз, а солдат В – 80 раз. С использованием генератора случайных чисел промоделировать соревнование между ними, в котором каждому нужно попасть в цель по 10 раз. Кто быстрее поразит все мишени?

7. В среднем из 128 компьютеров в течении месяца на одном выходит из строя дисплей. За тот же период на одной из 67 ЭВМ происходит поломка дисковода и на двух из 53 машин происходит крах системы из-за заражения вирусом. С использованием генератора случайных чисел смоделировать работу дисплейного

класса из 13 компьютеров за один месяц и ответить на вопрос: каково общее количество поломок за этот период?

8. Некий мужчина отправляется на службу, которая находится на расстоянии 1 км от дома. Дойдя до места работы, он вдруг вспоминает, что забыл поцеловать жену, и поворачивает назад. Пройдя полпути, он меняет решение, посчитав, что правильнее вернуться на работу. Пройдя 1/3 км по направлению к конторе, он вдруг осознает, что будет настоящем подлецом, если так и не поцелует жену. На этот раз, прежде чем снова изменить мнение он проходит 1/4 км. Так он продолжает метаться, и после N-го этапа, пройдя 1/N км, снова меняет решение. Это продолжается пока расстояние 1/N превышает длину шага, после чего человек останавливается. Определить координату точки остановки.

Практическая работа №5.
Построение алгоритмовпри работе с массивами

Массивы – это группа пронумерованных однотипных элементов.

В зависимости от размерности массивы подразделяются на одномерные, двумерные, трехмерные и т. д.

Алгоритмы работы с одномерными массивами

Одномерный (линейный) массив – это массив, в котором для обращения к элементам используется только один порядковый номер. Например, в табл.10.1 содержится информация о среднемесячной температуре в городе Магнитогорск в 2016 году.

Таблица 10.1

Таблица среднемесячной температуры

Месяц	1	2	3	4	5	6	7	8	9	10	11	12
Температура	-17	-16	-9	4	12	17	20,5	17	11	2	-6	-13

Таблица представляет собой линейную последовательность упорядоченных чисел. Значения температуры являются однотипными элементами, т.е. все числа вещественные. Элементы массива пронумерованы, т.е. все элементы массива существуют с прямым доступом. Например, дадим таблице имя Т, тогда через Т[1] обозначается температура в январе (первом месяце года), Т[5] - температура в мае и т. д. Такой порядковый номер называется индексом массива. Запишем общую форму:

<имя массива> [<индекс>]

Такая таблица называется одномерным массивом. Все элементы массива должны иметь одинаковый тип. Если массив состоит только из целых чисел, то тип массива – целый, если массив состоит из слов, то тип массива – строковый. В примере значения температур могут быть дробными, поэтому тип массива - вещественный.

При работе с массивом зачастую необходимо производить перебор элементов, двигаясь по индексам. Такой перебор происходит в цикле, в котором изменяется индекс массива от начального до конечного значения.

Ввод и вывод элементов массива

Для организации ввода и вывода элементов массива необходимо использовать цикл.

Рассмотрим алгоритм ввода и вывода 5 элементов *одномерного* массива вещественного типа в виде блок-схемы и на алгоритмическом языке, рис. 10.1.

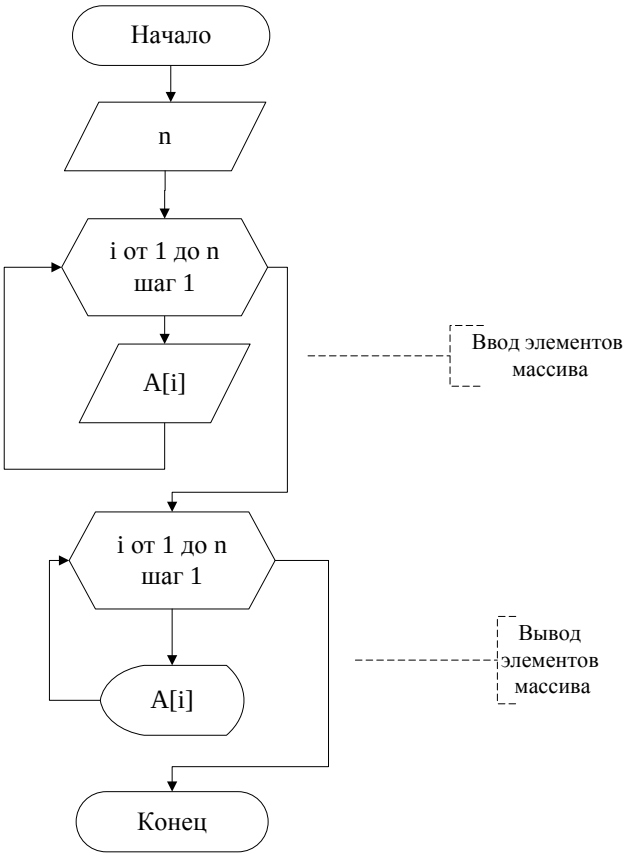


Рис. 10.1 Алгоритм ввода и вывода элементов *одномерного* массива в виде блок-схемы

```
алгввод_элементов
нач
цел i,n
  n:=5
вещтаб A[1:n]
нцдля i от 1 до n
ввод A[i]
кц
нцдля i от 1 до n
вывод "Элемент массива A за номером ",i," = ", A[i],нс
кц
кон
```

Заполнение массива с помощью генератора случайных чисел

Для генерации чисел в диапазоне *[A, B]* можно использовать следующее выражение: *random(B-A+1)+A.*

Например, выражение A[i]=random(100) генерирует числа от 0 до 99, при этом 100 не входит в диапазон.

Записать выражение для генерации чисел в диапазоне *[–15; 38]:*

A[i]=random(38-(-15)+1)+(-15), т.е. A[i]=random(54)-15.

Алгоритмы нахождения суммы и произведения

Задача 10.1.

1. Формулировка задачи

Заполнить массив *n* целыми случайными числами в диапазоне [–10; 15]. Составить алгоритм вычисления суммы четных элементов в виде блок-схемы, рис. 10.2, и программу на алгоритмическом языке.

2. Математическая постановка задачи

При накоплении суммы, начальное значение суммы обнуляем, т.е. Sum=0.
Выражение для генерации чисел в диапазоне *[–10; 15]:*
A[i]=random(15-(-10)+1)+(-10), т.е. запишем в следующем виде A[i]=random(26)-10.
Для четного значения элемента выполняется условие A[i] mod 2=0.

3. Выбор переменных программы

Из приведенного выше описания определяем следующие переменные:

- исходные данные – количество элементов массива n;
- начальное значение суммы равно 0 (Sum=0);
- результат – значение накопленной суммы Sum.

```
алгсумм
нач
целi,Sum,n
ввод n
  Sum:=0;
целтаб A[1:n]
нцдля i от 1 до n
A[i]:=rnd(26)-10
еслиmod(A[i],2) = 0 то Sum:=Sum+A[i] все
кц
вывод "Сумма четных элементов массива  = ", Sum
кон
```

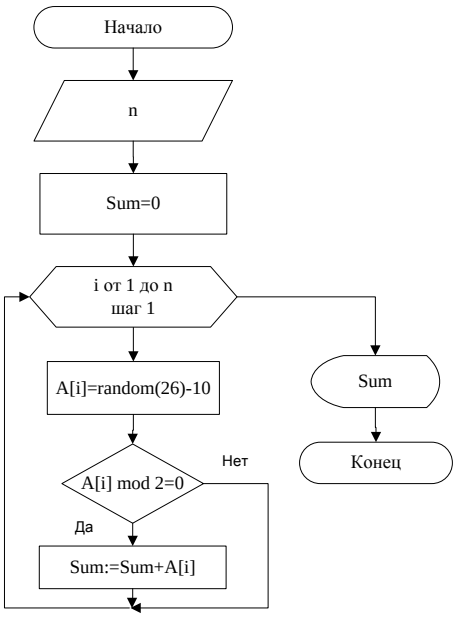


Рис. 10.2 Алгоритм подсчета суммы четных элементов массива в виде блок-схемы

Задача 10.2.

1. Формулировка задачи

Заполнить массив *n* целыми случайными числами в диапазоне [–5; 10]. Составить алгоритм вычисления произведения положительных элементов в виде блок-схемы, рис. 10.3, и программу на алгоритмическом языке.

2. Математическая постановка задачи

При накоплении произведения, начальное значение произведения равно 1, т.е. pr=1.
Выражение для генерации чисел в диапазоне *[–5; 10]:*
A[i]=random(10-(-5)+1)+(-5), т.е. запишем в следующем виде A[i]=random(16)-5.
Для положительного значения элемента выполняется условие A[i] >0.

3. Выбор переменных программы

Из приведенного выше описания определяем следующие переменные:

- исходные данные – количество элементов массива n;
- начальное значение произведения равно 1 (pr=1);
- результат – значение произведения pr.

```
алгпроизв
нач
цел i,pr,n
ввод n
pr:=1;
целтаб A[1:n]
нцдля i от 1 до n
A[i]:=rnd(16)-5
если A[i]>0 то pr:=pr*A[i] все
кц
вывод "Произведение положительных элементов массива  = ", pr
кон
```

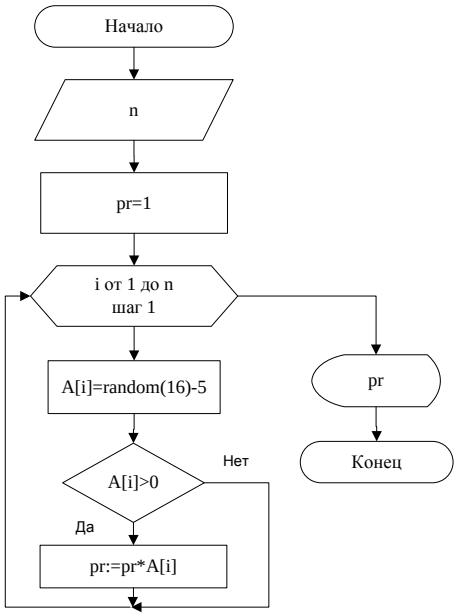


Рис. 10.3 Алгоритм подсчета произведения положительных элементов массива в виде блок-схемы

Практическая работа №6. Построение алгоритмов поиска значений

Задача 10.3.

1. Формулировка задачи

Заполнить массив *n* целыми случайными числами в диапазоне [–8; 7]. Составить алгоритм поиска минимального элемента массива и его индекса в виде блок-схемы, рис.10.4, и программу на алгоритмическом языке.

2. Математическая постановка задачи

Выражение для генерации чисел в диапазоне *[–8; 7]:*

A[i]=random(7-(-8)+1)+(-8), т.е. запишем в следующем виде A[i]=random(16)-8.

Минимальное значение присваивается первому элементу массива, т.е. min_ch=A[1], индекс минимального элемента равен 1.

Сравниваем минимальный элемент с последующими. Если значение элемента меньше минимального, то минимальному значению присваивается значение элемента массива, а его индекс фиксируется в переменной k.

3. Выбор переменных программы

Из приведенного выше описания определяем следующие переменные:

- исходные данные – количество элементов массива n;
- минимальное значение min_ch=A[1], индекс k=1;
- результат – значение минимума min_ch и индекс элемента k.

```
алгмин
нач
цел i,min_ch,n
вводn
целтабA[1:n]
нцдля i от 1 до n
A[i]:=rnd(16)-8
кц
min_ch:=A[1]
k:=1
нцдляiот 2 доn
если A[i]<min_ch то
min_ch:=A[i]
k:=i
все
кц
вывод "минимальное число массива = ", min_ch
вывод "индекс элемента массива = ", k
кон
```

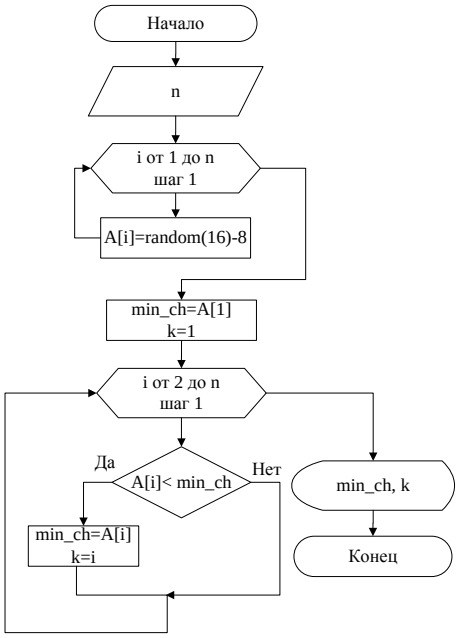


Рис. 10.4. Алгоритм поиска минимального значения в виде блок-схемы

Индивидуальные задания

Сгенерировать элементы массива случайным образом в заданном диапазоне [a; b] согласно варианту.

№	Диапазон		№	Диапазон		№	Диапазон	
1.	a = -10	b = 7	*6.*	a = -15	b = 88	*11.*	a = -19	b = 37
2.	a = -15	b = 37	*7.*	a = -18	b = 23	*12.*	a = -105	b = 66
3.	a = 25	b = 57	*8.*	a = -37	b = 56	*13.*	a = -48	b = 32
4.	a = -65	b = 2	*9.*	a = -22	b = 132	*14.*	a = -25	b = 87
5.	a = -100	b = 107	*10.*	a = -8	b = 45	*15.*	a = -13	b = 10

1. Дана последовательность целых чисел a_1, ..., a_{50}. Получить сумму тех чисел данной последовательности, которые:
   - кратны 5;
   - нечетны и отрицательны;
   - удовлетворяют условию $a_i < i^2$.
2. Даны натуральное число n и последовательность целых чисел a_1, ..., a_n. Найти количество и сумму тех элементов данной последовательности, которые делятся на 5 и не делятся на 7.
3. Даны натуральные числа n, p и последовательность целых чисел a_1, ..., a_n. Получить произведение элементов последовательности a_1, ..., a_n, кратных p.
4. Даны натуральные p, q и последовательность целых чисел a_1, ..., a_{67} ($p > q \geq 0$). В последовательности a_1, ..., a_{67} заменить на нуль все элементы, модуль которых при делении на p дает в остатке q.
5. Даны натуральное число n, последовательность действительных чисел a_1, ..., a_n. Получить удвоенную сумму всех положительных элементов последовательности a_1, ..., a_n.
6. Даны натуральное число n, последовательность действительных чисел a_1, ..., a_n. В последовательности a_1,..., a_n все отрицательные элементы увеличить на 0,5, а все неотрицательные заменить на 0,1.

7. Даны натуральное число n, последовательность действительных чисел $a_1, ..., a_n$. В последовательности $a_1, ..., a_n$ все элементы меньше 2, заменить нулями. Получить сумму элементов, принадлежащих отрезку [3; 7], а также число таких элементов.

8. Даны натуральное число n, последовательность действительных чисел $a_1,..., a_n$. В последовательности $a_1, ..., a_n$ все неотрицательные элементы, не принадлежащие отрезку [1; 2], заменить на единицу. Получить число отрицательных элементов и число элементов, принадлежащих отрезку [1;2].

9. Даны натуральное число n, целые числа $a_1, ..., a_n$. Получить сумму положительных и число отрицательных четных элементов последовательности $a_1, ..., a_n$.

10. Даны натуральное число n, последовательность целых чисел $a_1, ..., a_n$. Заменить все большие семи элементы последовательности $a_1, ..., a_n$ числом 7. Вычислить количество таких элементов, которые имеют четный индекс.

11. Дана последовательность целых чисел $a_1, ..., a_{45}$. Получить число отрицательных элементов последовательности $a_1, ..., a_{35}$ и число нулевых элементов всей последовательности.

12. Дан массив целых чисел, состоящих из 20 элементов. Найти:

– сумму элементов, имеющих нечетное значение.

– вывести индексы тех элементов, значения которых больше заданного числа *A*.

– определить, есть ли в заданном массиве положительные элементы кратные *K* (*K* вводится с клавиатуры). Если таковых нет, то вывести сообщение.

13. Дан массив целых чисел. Найти, сколько в нем пар одинаковых соседних элементов.

14. Дан массив целых чисел, состоящий из 25 элементов. Найти:

– сумму элементов, имеющих нечетные индексы.

– подсчитать количество элементов массива, значения которых больше заданного числа *A* и кратны 5.

– найти номер первого отрицательного элемента, делящегося на 5 с остатком.

15. Заданы *N* целых чисел. Определить, сколько среди них четных и притом делящихся на 7 без остатка. Вывести эти числа.

Алгоритмы работы с двумерными массивами

Двумерный массив – это группа пронумерованных однотипных данных, доступ к которым осуществляется по двум индексам. Двумерный массив представляется таблицей, где n – это количество строк, m – это количество столбцов. Таблица, изображенная на рис. 10.5 имеет имя A. Ячейке, стоящей в i-й строке и j-м столбце соответствует элемент массива A[i,j].

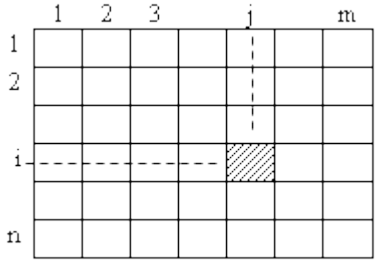


Рис. 10.5. Двумерный массив

Рассмотрим алгоритм ввода и вывода элементов *двумерного* массива размером *nxm* в виде блок-схемы, рис. 10.6, и на алгоритмическом языке.

```
алгввод_элементов
нач
цел i, j ,n,m
n:=3
  m:=4
вещтаб A[1:n,1:m]
нцдля i от 1 до n
нцдля j от 1 доm
ввод A[i,j]
кц
кц
нцдля i от 1 до n
нцдля j от 1 доm
вывод A[i,j],нс
кц
кц
кон
```

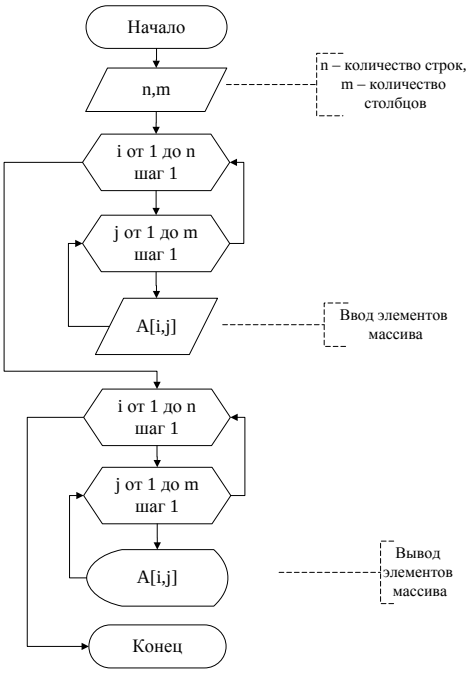


Рис. 10.6 Алгоритм ввода и вывода элементов *двумерного* массива в виде блок-схемы

Алгоритмы нахождения суммы и произведения

Задача 10.4.

1. Формулировка задачи

Заполнить двумерный массив *nxm* целыми случайными числами в диапазоне [–10; 15]. Составить алгоритм вычисления суммы четных элементов программу на алгоритмическом языке и в виде блок-схемы, рис.10.7.

2. Математическая постановка задачи

При накоплении суммы, начальное значение суммы обнуляем, т.е. Sum=0.

Выражение для генерации чисел в диапазоне *[–10; 15]:*

A[i,j]=random(15-(-10)+1)+(-10), т.е. запишем в следующем виде A[i,j]=random(26)-10.

Для четного значения элемента выполняется условие A[i,j] mod 2=0.

3. Выбор переменных программы

Из приведенного выше описания определяем следующие переменные:

- исходные данные – количество строк массива n, столбцов – m;
- начальное значение суммы равно 0 (Sum=0);
- результат – значение накопленной суммы Sum.

```
алгсумм_двум_мас
нач
цели,Sum,n,m
вводn
вводm
Sum:=0;
целтабA[1:n,1:m]
нцдля i от 1 до n
нцдля j от 1 доm
A[i,j]:=rnd(26)-10
еслиmod(A[i,j],2) = 0 то Sum:=Sum+A[i,j] все
кц
вывод "Сумма четных элементов массива  = ", Sum
кон
```

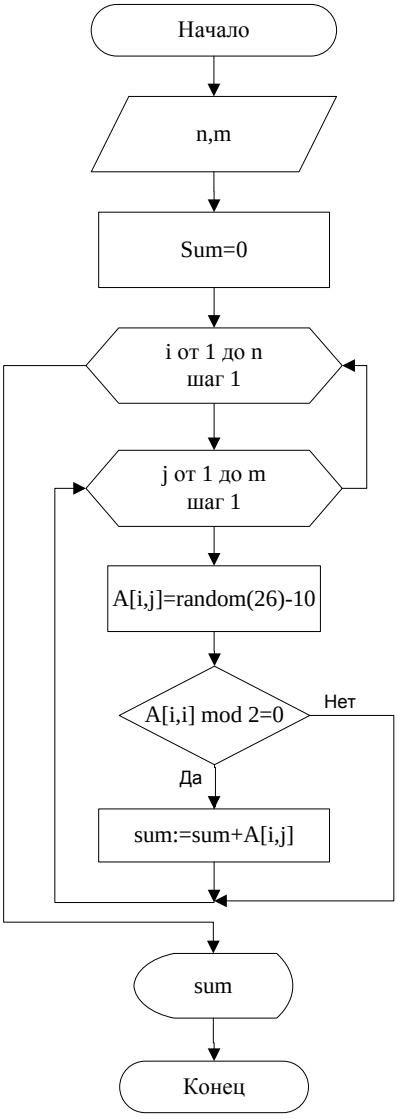


Рис. 10.7 Алгоритм подсчета суммы четных элементов
двумерного массива в виде блок-схемы

Задача 10.5.

1. Формулировка задачи

Заполнить двумерный массив *nxm* целыми случайными числами в диапазоне [–5; 10]. Составить алгоритм вычисления произведения положительных элементов в виде блок-схемы, рис.10.8, и программу на алгоритмическом языке.

2. Математическая постановка задачи

При накоплении произведения, начальное значение произведения равно 1, т.е. pr=1.
Выражение для генерации чисел в диапазоне *[–5; 10]:*
A[i,j]=random(10-(-5)+1)+(-5), т.е. запишем в следующем виде A[i,j]=random(16)-5.
Для положительного значения элемента выполняется условие A[i,j] >0.

3. Выбор переменных программы

Из приведенного выше описания определяем следующие переменные:

- исходные данные – количество строк массива n, столбцов – m;
- начальное значение произведения равно 1 (pr=1);
- результат – значение произведения pr.

алгпроизв_двум_мас
нач
цел i,pr,n,m
ввод n
ввод m
  pr:=1;
целтабA[1:n,1:m]
нцдля i **от** 1 **до** n
нцдля j **от** 1 **до**m
A[i,j]:=rnd(16)-5
если A[i,j]>0 **то** pr:=pr*A[i,j] **все**
кц
кц
вывод "Произведение положительных элементов массива = ", pr
кон

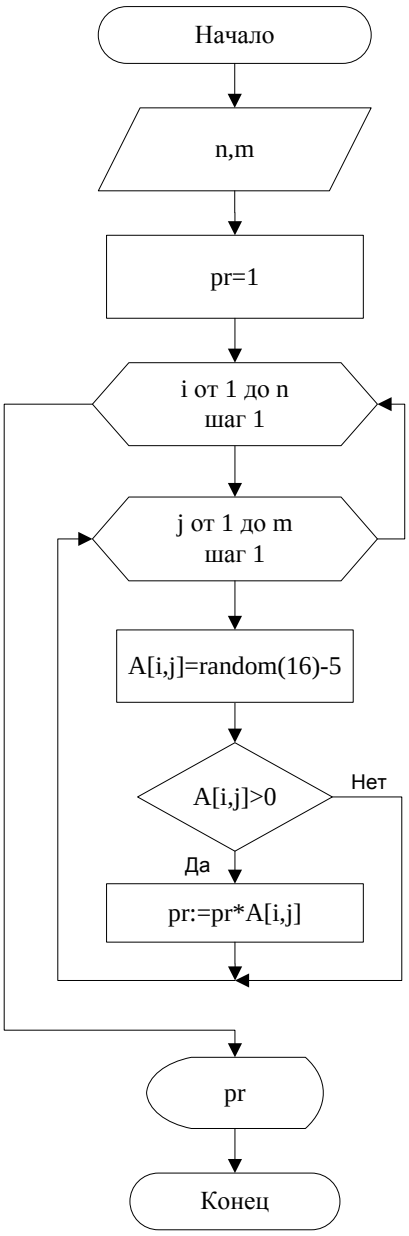


Рис. 10.8 Алгоритм подсчета произведения положительных элементов двумерного массива в виде блок-схемы

Алгоритмы поиска значений

Задача 10.6.

1. Формулировка задачи

Заполнить двумерный массив *nxm* целыми случайными числами в диапазоне [–8; 7]. Составить алгоритм поиска минимального элемента массива и его индексов в виде блок-схемы, рис.10.9, и программу на алгоритмическом языке.

2. Математическая постановка задачи

Выражение для генерации чисел в диапазоне *[–8; 7]:*

A[i,j]=random(7-(-8)+1)+(-8), т.е. запишем в следующем виде A[i,j]=random(16)-8.

Минимальное значение присваивается первому элементу массива, т.е. min_ch=A[1,1], индексы минимального элемента равны 1.

Сравниваем минимальный элемент с элементами массива. Если значение элемента меньше минимального, то минимальному значению присваивается значение элемента массива, а его индекс строки фиксируется в переменной k, индекс столбца – z.

3. Выбор переменных программы

Из приведенного выше описания определяем следующие переменные:

- исходные данные – количество элементов массива n;
- минимальное значение min_ch=A[1,1], индекс k=1, z=1;
- результат – значение минимума min_ch и индекс строки элемента k, индекс столбца – z.

```
алгмин
нач
цел i,min_ch,n,m
ввод n
ввод m
целтаб A[1:n, 1:m]
нцдля i от 1 до n
нцдля j от 1 доm
A[I,j]:=rnd(16)-8
кц
кц
min_ch:=A[1,1]
k:=1
z:=1
нцдляiот 1 доn
нцдля j от 1 доm
если A[i,j]<min_ch то
                    min_ch:=A[I,j]
                    k:=i
                    z:=1
все
кц
кц
вывод "минимальное число массива = ", min_ch
вывод "индекс элемента массива = ", k
кон
```

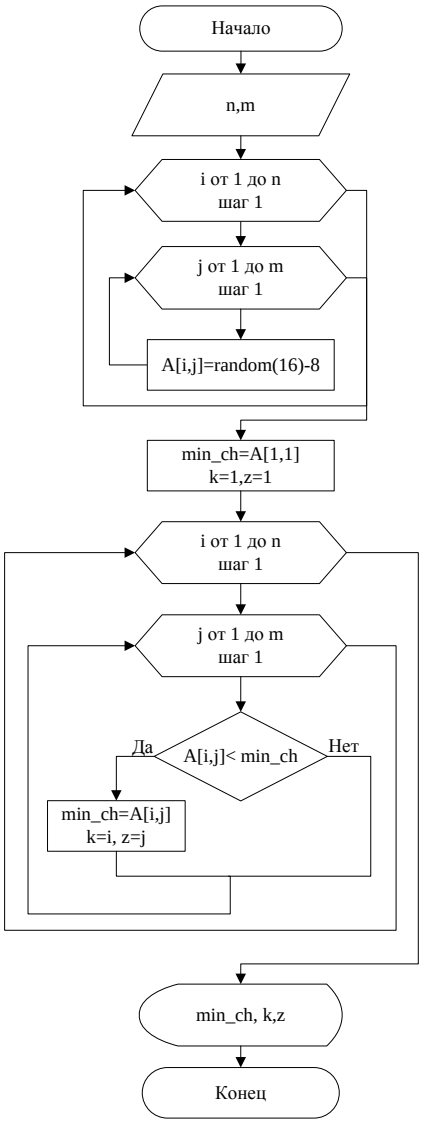


Рис. 10.9. Алгоритм поиска минимального значения в виде блок-схемы

Индивидуальные задания

1. Задана квадратная матрица размером $k \times k$, составить алгоритм вычисления суммы и произведения элементов матрицы, находящихся на главной диагонали.
2. Составить алгоритм, который позволяет для матрицы размерностью $m \times n$ вычислить произведение положительных, сумму отрицательных и количество нулевых элементов.
3. Дана матрица *B*. Для каждого столбца с чётным номером вычислить и напечатать сумму квадратов элементов этого столбца, а для каждого столбца с нечетным номером вычислить произведение элементов.
4. Дан двухмерный массив *A*. Каждый элемент массива, стоящий выше главной диагонали, заменить его квадратом, а ниже диагонали – кубом его значения. Элементы главной диагонали оставить без изменения.
5. Задан двухмерный массив *B*. Найти в нём максимальный элемент и разделить на него каждый элемент массива *B*.
6. Дана матрица *B*. Для каждого столбца матрицы вычислить и напечатать разность между квадратом суммы и суммой квадратов элементов этого столбца.
7. Заданы двухмерный массив 5×5 и число *K*. Разделить элементы *K*-ой строки на диагональный элемент, расположенный в данной строке.

8. Определить и напечатать среднее значение элементов двухмерного массива размером $n \times m$. Найти индексы элемента, наиболее близкого по значению к среднему.

9. Составить алгоритм нахождения минимального положительного элемента матрицы размерностью $m \times n$, а также значение индексов этого элемента.

10. Составить алгоритм для определения количества отрицательных, положительных и равных нулю элементов матрицы $R(m \times n)$.

11. Составить алгоритм вычисления суммы и произведения элементов квадратной матрицы A размерностью $n \times n$, расположенных ниже главной диагонали.

12. Дана действительная матрица размера $m \times n$. Найти значение наибольшего по модулю элемента матрицы, а также индексы всех элементов с найденными значениями модуля.

13. Дана действительная матрица размера $m \times n$. Найти максимальные элементы по строкам и их сумму.

14. В данной действительной квадратной матрице порядка n найти сумму элементов строки, в которой расположен элемент с наименьшим значением. Предполагается что такой элемент единственный.

15. В данной действительной матрице размера 6x9 поменять местами строку, содержащую элемент с наибольшим значением, со строкой, содержащий элемент с наименьшим значением. Предполагается, что эти элементы единственны.

Практическая работа №7. Построение алгоритмов сортировки

Сортировка – это процесс перестановки объектов данного множества в определенном порядке. Цель сортировки – ускорить последующий поиск элементов в отсортированном множестве.

Пусть даны элементы – a_1, a_2a_3, …, a_n. Сортировка означает перестановку этих элементов в таком порядке: a_{k1}, a_{k2}, a_{k3}, a_{kn}, что при заданной функции упорядочения f справедливо отношение $f(a_{k1})<=f(a_{k2})<=f(a_{k3})<=…<=f(a_{kn})$.

Алгоритм сортировки – это алгоритм для упорядочивания элементов в списке. Эффективность алгоритмов сортировки определяется быстродействием, экономией памяти и сокращением времени, затрачиваемого программистом, на реализацию алгоритма

На рис. 10.10 приведена классификация алгоритмов сортировки в зависимости от их эффективности.

Рис. 10.10 Классификация алгоритмов сортировки

Устойчивой сортировкой называют алгоритм, который не меняет последовательность одинаковых элементов. К устойчивым сортировкам относятся сортировка выбором, пузырьком, перемешиванием, вставками, «гномья», слиянием, с помощью двоичного дерева, Timsort, подсчетом и блочная сортировка.

Неустойчивой сортировкой называют алгоритм, который может менять последовательность одинаковых элементов. К неустойчивым сортировкам относятся сортировки Шелла, «расческой», пирамидальная, плавная, быстрая, интроспективная, «терпеливая», поразрядная и Stoogesort.

Непрактичные алгоритмы сортировки требуют значительно большего времени, значительно большей памяти или специализированного аппаратного обеспечения. К ним относятся сортировки перестановками, «глупая», «блинная», Bogosort и BeadSort.

Достоинством алгоритмов, не основанных на сравнениях, является их быстрота при условии использования подходящего типа входных данных. К данному виду сортировок относятся блочная или корзинная, лексикографическая или поразрядная и сортировки подсчетом.

К прочим сортировкам относятся сортировки топологическая и внешняя.

В табл. 10.2 приведен сравнительный анализ алгоритмов сортировки [15].

Таблица 10.2

Сравнение преимуществ и недостатков основных видов алгоритмов сортировки

№ п/п	Виды алгоритмов сортировки	Преимущества	Недостатки
1.	Алгоритмы устойчивой сортировки	Сохранение взаимного расположения равных элементов важно при сортировке по одному полю данных, состоящих из нескольких полей	Для обеспечения устойчивости практически всегда необходимы дополнительная память и время.

2.	Алгоритмы неустойчивой сортировки	Как правило, для сортировки требуется меньше памяти и времени, чем при использовании алгоритмов устойчивой сортировки.	При сортировке по одному полю данных, состоящих из нескольких полей, не сохраняется взаимное расположение равных элементов
3.	Непрактичные алгоритмы сортировки	Эффективны для специфических целей, например, обучения или для сортировки с особыми условиями.	По сравнению с другими алгоритмами требуют значительно большего времени, значительно большей памяти или специализированного аппаратного обеспечения.
4.	Алгоритмы, не основанные на сравнениях	Быстрота при условии использования подходящего массива входных данных.	Низкая эффективность, если массив входных данных не является удачным для соответствующего алгоритма сортировки.
5.	Алгоритмы топологической сортировки	Позволяют построить корректную последовательность выполнения действий, каждое из которых может зависеть от другого.	Сравнительно узкая сфера возможного использования.

Рассмотрим наиболее распространенные методы сортировки.

Сортировка пузырьком

Сортировка пузырьком или простыми обменами (англ. Bubble sort) относится к простому алгоритму сортировки. Он эффективен для небольших массивов, но данный метод сортировки лежит в основе более совершенных алгоритмов сортировки, таких как шейкерная сортировка, пирамидальная сортировка и быстрая сортировка.

Приведем пример алгоритма сортировки пузырьком.

Исходный набор чисел	3	9	5	2	7	6
1 шаг	3	9	5	2	7	6
2 шаг	3	5	9	2	7	6
3 шаг	3	5	2	9	7	6
4 шаг	3	5	2	7	9	6
5 шаг	3	5	2	7	6	***9***
6 шаг	3	5	2	7	6	***9***
7 шаг	3	2	5	7	6	***9***
8 шаг	3	2	5	7	6	***9***

9 шаг	3	2	5	6	***7***	***9***
10 шаг	2	3	5	6	***7***	***9***
11 шаг	2	3	5	6	***7***	***9***
12 шаг	2	3	5	***6***	***7***	***9***
13 шаг	2	3	5	***6***	***7***	***9***
14 шаг	2	3	***5***	***6***	***7***	***9***
15 шаг	2	***3***	***5***	***6***	***7***	***9***
Отсортированный массив	2	3	5	6	7	9

Алгоритм состоит из повторяющихся проходов по сортируемому массиву. За каждый проход элементы последовательно сравниваются попарно и, если порядок в паре неверный, выполняется обмен элементов. Проходы по массиву повторяются до тех пор, пока на очередном проходе не окажется, что обмены больше не нужны, что означает – массив отсортирован. При каждом проходе алгоритма по внутреннему циклу, очередной наибольший элемент массива ставится на своё место в конце массива рядом с предыдущим, «наибольшим элементом», а наименьший элемент перемещается на одну позицию к началу массива («всплывает» до нужной позиции, как пузырёк в воде, отсюда и название) [16].

Задача 10.7.

1. Формулировка задачи

Дан одномерный массив из N целыхслучайных чисел в диапазоне [–18; 17]. Составить алгоритм сортировки элементов массива по убыванию (невозрастанию) в виде блок-схемы и программу на алгоритмическом языке.

2. Математическая постановка задачи

Выражение для генерации чисел в диапазоне *[–18; 17]:*

A[i]=random(17-(-18)+1)+(-18), т.е. запишем в следующем виде A[i]=random(36)-18.

Сравниваем попарно элементы массива. Если значение элемента A[i] меньше A[i+1], то производим их перестановку.

3. Выбор переменных программы

Из приведенного выше описания определяем следующие переменные:

- исходные данные – количество элементов массива N;
- для перестановки введем переменную temp.

На рис.10.11. приведен алгоритм сортировки элементов массива по убыванию (невозрастанию) методом пузырька в виде блок-схемы.

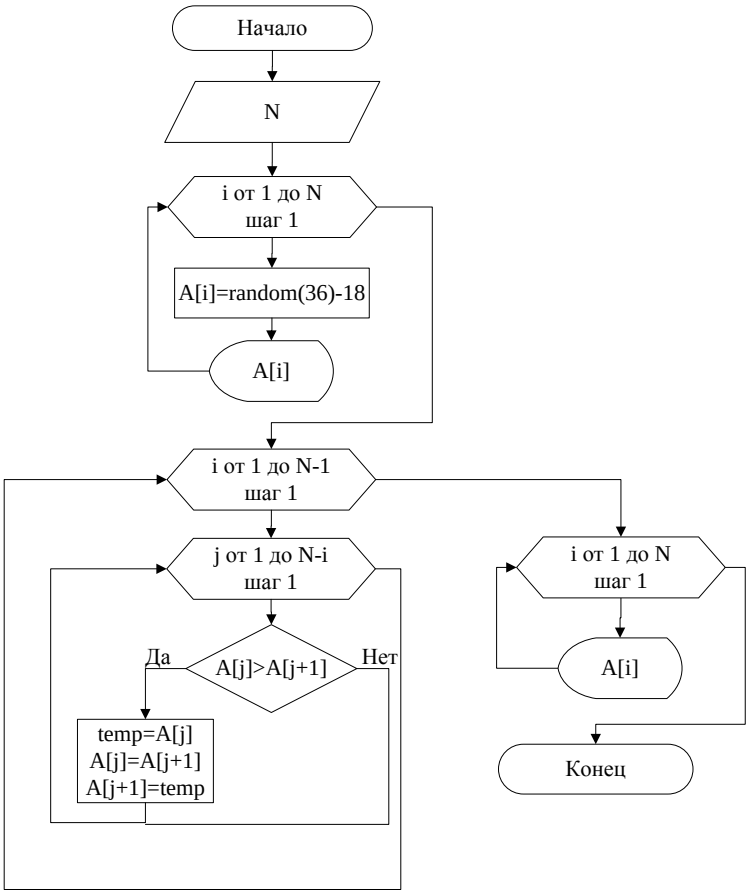


Рис.10.11. Алгоритм сортировки элементов массива по убыванию (невозрастанию) методом пузырька в виде блок-схемы

```
алг сортировка пузырьком
нач
вводцел N
целтаб A[1:N]
цел i, j, temp
  нцдля i от 1 до N
   A[i] :=rnd(36)-18
вывод A[i], " "
кц
выводнс
  нцдля i от 1 до N-1
нцдля j от 1 до N-i
если A[j] > A[j+1] то
      temp := A[j]
      A[j] := A[j+1]
      A[j+1] := temp
все
кц
кц
  нцдля i от 1 до N
выводA[i], " "
кц
кон
```

Сортировка выбором

Сортировка выбором – один из наиболее простых алгоритмов сортировки, в основе которого лежит операция сравнения.

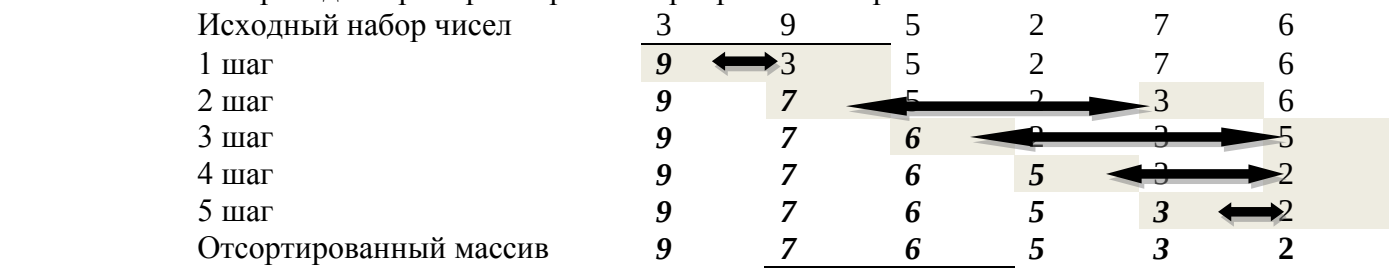
Алгоритм сортировки выбором состоит из следующих этапов:

1.Найдем элемент с наибольшим значением.

2. Поменяем его местами с первым элементом.

3. Те же действия проделаем с оставшимися (без первого) N-1 элементами, затем с N-2 элементами и т. д., пока не останется один элемент – последний. Он будет наименьшим.

Приведем пример алгоритма сортировки выбором.



На рис.10.12. приведен алгоритм сортировки выбором элементов массива по убыванию (невозрастанию) в виде блок-схемы.

```
алг сортировка выбором
нач
вводцел N
целтабA[1:N]
цел i, j, temp, ind
  нцдля i от 1 до N
   A[i] :=rnd(36)-18
вывод A[i], " "
кц
выводнс
  нцдля j от 1 до N-1
   ind := N
нцдля i от j до N-1
если A[i] < A[ind] то
      ind := i
все
кц
   temp := A[j]
   A[j] := A[ind]
   A[ind] := temp
кц
 нцдляiот 1 доN
вывод A[i], " "
кц
кон
```

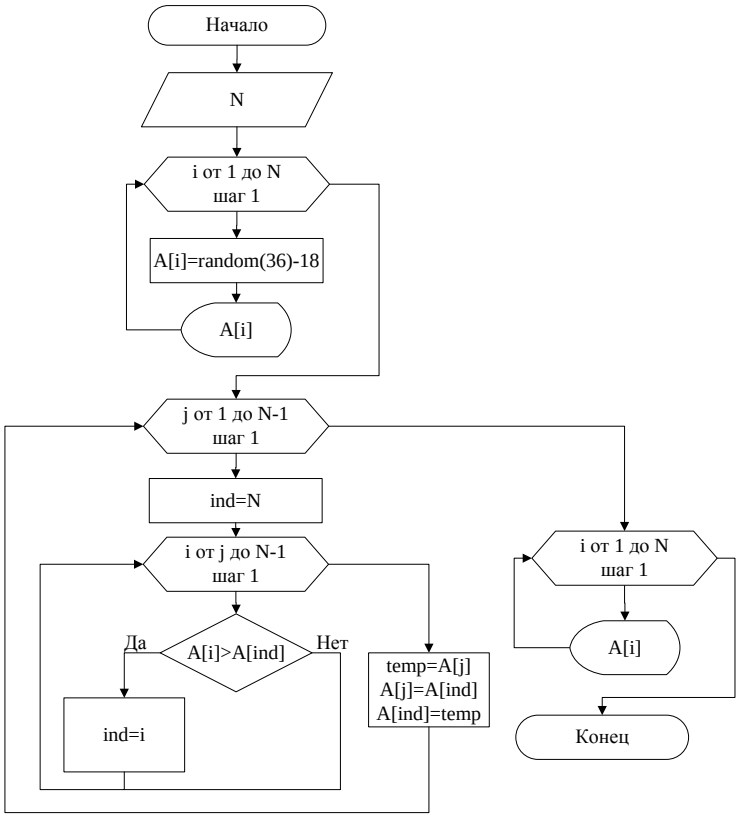


Рис.10.11. Алгоритм сортировки выбором элементов массива по убыванию (невозрастанию) в виде блок-схемы

Индивидуальные задачи

1. В массиве из 10 целых чисел найти наименьший элемент. Выполнить сортировку по невозрастанию между минимальным и последним элементом.
2. В массиве из 15 вещественных чисел найти наибольший элемент. Выполнить сортировку по неубыванию между максимальным и последним элементом.
3. В массиве из 25 вещественных чисел найти наименьший элемент. Выполнить сортировку по убыванию между минимальным и первым элементом.
4. Упорядочить по неубыванию массив, содержащий 20 целых чисел.
5. Упорядочить по невозрастанию массив, содержащий 10 целых чисел.
6. Упорядочить по неубыванию массив, содержащий 15 вещественных чисел.
7. Упорядочить по невозрастанию массив, содержащий 25 вещественных чисел.
8. Дан массив целых чисел, содержащий 20 элементов, записать в этот же массив сначала все отрицательные числа и нули, затем все положительные, сохраняя порядок их следования.
9. Дан двухмерный массив, содержащий 5 строк и 3 столбца. Элементами массива являются целые числа. Переставить строки таким образом, чтобы элементы первого столбца были упорядочены по неубыванию.
10. Дан двухмерный массив, содержащий 4 строки и 5 столбцов. Элементами массива являются целые числа. Переставить строки таким образом, чтобы элементы пятого столбца были упорядочены по невозрастанию.
11. Дан двухмерный массив, содержащий 3 строки и 4 столбца. Элементами массива являются вещественные числа. Переставить строки таким образом, чтобы элементы первого столбца были упорядочены по невозрастанию.

12. Дан двухмерный массив, содержащий 4 строки и 4 столбца. Элементами массива являются вещественные числа. Переставить строки таким образом, чтобы элементы второго столбца были упорядочены по невозрастанию.

13. Дан двухмерный массив, содержащий 3 строки и 4 столбца. Элементами массива являются целые числа. Переставить столбцы таким образом, чтобы элементы второй строки были упорядочены по невозрастанию.

14. Дан одномерный массив A[n]. Упорядочить его по неубыванию до первого максимального числа, а от него до конца массива по невозрастанию.

15. Определите номер наибольшего элемента массива A и наибольшего значения среди модулей элементов массива A. Упорядочить массив по убыванию абсолютных величин.

Практическая работа №8.
Построение рекурсивных алгоритмов

Рекурсия – это определение объекта через обращение к самому себе. Рекурсивный алгоритм – это алгоритм, в описании которого прямо или косвенно содержится обращение к самому себе. В технике процедурного программирования данное понятие распространяется на функцию, которая реализует решение отдельного блока задачи посредством вызова из своего тела других функций, в том числе и себя самой. Если при этом на очередном этапе работы функция организует обращение к самой себе, то такая функция является рекурсивной. Прямое обращение функции к самой себе предполагает, что в теле функции содержится вызов этой же функции, но с другим набором фактических параметров. Такой способ организации работы называется *прямой* рекурсией. Например, чтобы найти сумму первых n натуральных чисел, надо сумму первых (n-1) чисел сложить с числом n, то есть имеет место зависимость: Sn=Sn-1+n. Вычисление происходит с помощью аналогичных рассуждений. Такая цепочка взаимных обращений в конечном итоге сведется к вычислению суммы одного первого элемента, которая равна самому элементу [17].

Задача 11.1.[18]

1. Формулировка задачи

Алгоритм вычисления значения функции F(n), где n – натуральное число, задан следующими соотношениями:

F(n) = 1 при n ≤2;

F(n) = 2 × F(n−1) + F(n−2) при n> 2.

Чему равно значение функции F(6)?

Решение

По условию задачи n=6. Подставим аргумент в функцию:

1. n=6, т.е. n>2, следовательно,
   F(6) = 2 × F(6−1) + F(6−2)= 2 × F(5) + F(4)
2. n=5, т.е. n>2, следовательно,
   F(5) = 2 × F(5−1) + F(5−2)= 2 × F(4) + F(3)
3. n=4, т.е. n>2, следовательно,
   F(4) = 2 × F(4−1) + F(4−2)= 2 × F(3) + F(2)
4. n=3, т.е. n>2, следовательно,
   F(3) = 2 × F(3−1) + F(3−2)= 2 × F(2) + F(1)
5. n=2, т.е. выполняется условие n ≤2, следовательно, F(2) = 1
6. n=1, т.е. выполняется условие n ≤2, следовательно, F(1) = 1

Последовательно в обратном направлении подставляем найденные значения функции:

7. F(1) и F(2) в функцию F(3)= 2* F(***2***) + F(***1***)=2*1+1=3;
8. F(2) и F(3) в функцию F(4)= 2* F(***3***) + F(***2***)=2*3+1=7;
9. F(3) и F(4) в функцию F(5)= 2* F(***4***) + F(***3***)=2*7+3=17;
10. F(4) и F(5) в функцию F(6)= 2* F(***5***) + F(***4***)=2*17+7=41.

Ответ: F(6) = 41.

Индивидуальные задачи

Алгоритмы, опирающиеся на несколько предыдущих значений [19]

1. № 4645. Алгоритм вычисления значения функции F(n), где n – натуральное число, задан следующими соотношениями:

F(1) = 1

F(2) = 3

F(n) = F(n–1) * n + F(n–2) * (n – 1) , при n >2

Чему равно значение функции F(5)?

2. № 4646. Алгоритм вычисления значения функции F(n), где n – натуральное число, задан следующими соотношениями:

F(1) = 1

F(2) = 3

F(n) = F(n−1) * F(n−2) + (n−2), при n > 2

Чему равно значение функции F(5)?

3.№ 4647. Алгоритм вычисления значения функции F(n), где n – натуральное число, задан следующими соотношениями:

F(1) = 1

F(2) = 2

F(n) = 2 * F(n–1) + (n – 2) * F(n–2), при n >2

Чему равно значение функции F(6)?

4.№ 4648. Последовательность чисел Фибоначчи задается рекуррентным соотношением:

F(1) = 1

F(2) = 1

F(n) = F(n–2) + F(n–1), при n >2, где n – натуральное число.

Чему равно восьмое число в последовательности Фибоначчи?

5. № 4649. Последовательность чисел Фибоначчи задается рекуррентным соотношением:

F(1) = 1

F(2) = 1

F(n) = F(n–2) + F(n–1), при n >2, где n – натуральное число.

Чему равно девятое число в последовательности Фибоначчи?

6. № 4650. Последовательность чисел Трибоначчи задается рекуррентным соотношением:

F(1) = 0

F(2) = 1

F(3) = 1

F(n) = F(n–3) + F(n–2) + F(n–1), при n >3, где n – натуральное число.

Чему равно девятое число в последовательности Трибоначчи?

7. № 4651. Последовательность чисел Трибоначчи задается рекуррентным соотношением:

F(1) = 0

F(2) = 1

F(3) = 1

F(n) = F(n–3) + F(n–2) + F(n–1), при n >3, где n – натуральное число.

Чему равно одиннадцатое число в последовательности трибоначчи?

8.№ 4652. Последовательность чисел Люка задается рекуррентным соотношением:

F(1) = 2

F(2) = 1

F(n) = F(n–2) + F(n–1), при n >2, где n – натуральное число.

Чему равно восьмое число в последовательности Люка?

9. № 4653. Последовательность чисел Люка задается рекуррентным соотношением:

F(1) = 2

F(2) = 1

F(n) = F(n–2) + F(n–1), при n >2, где n – натуральное число.

Чему равно десятое число в последовательности Люка?

10. № 4654. Последовательность чисел Падована задается рекуррентным соотношением:

F(1) = 1

F(2) = 1

F(3) = 1

F(n) = F(n–3) + F(n–2), при n >3, где n – натуральное число.

Чему равно десятое число в последовательности Падована?

11. № 4655. Последовательность чисел Падована задается рекуррентным соотношением:

F(1) = 1

F(2) = 1

F(3) = 1

F(n) = F(n–3) + F(n–2), при n >3, где n – натуральное число.

Чему равно двенадцатое число в последовательности Падована?

12. № 4658. Алгоритм вычисления значения функции F(n), где n – натуральное число, задан следующими соотношениями:

F(1) = 1

F(2) = 1

F(n) = F(n–1) * n − 2 * F(n–2), при n >2

Чему равно значение функции F(6)?

13. № 4659. Алгоритм вычисления значения функции F(n), где n – натуральное число, задан следующими соотношениями:

F(1) = 1

F(2) = 2

F(n) = F(n–1) − F(n–2) + 2 * n, при n >2

Чему равно значение функции F(6)?

14. № 4660. Алгоритм вычисления значения функции F(n), где n – натуральное число, задан следующими соотношениями:

F(1) = 1

F(2) = 2

F(n) = (F(n–1) − F(n–2)) * n, при n >2

Чему равно значение функции F(8)?

15. № 5089. Алгоритм вычисления значения функции F(n), где n - натуральное число, задан следующими соотношениями:

F(1) = 5; F(2) = 5;

F(n) = 5*F(n − 1) − 4*F(n − 2) при n >2.

Чему равно значение функции F(13)?