

Практическая работа № 1

Разработка приложений по различным алгоритмам

Тема: Тема 2.9. Составление блок-схем разветвляющихся алгоритмов

Цели занятия: Усвоить понятия: алгоритм как фундаментальное понятие информатики, свойства алгоритмов, основные типы алгоритмов, изучить способы представления алгоритмов, научиться составлять алгоритмы в виде блок – схем и на языке Паскаль.

Ход практического занятия:

1. Ознакомиться с методическими рекомендациями.
2. Выполнить задание, согласно номеру своего варианта
3. Составьте отчет в соответствии с заданием.
4. Сделать выводы по выполненной работе.

Методические рекомендации

Алгоритм — точное и понятное предписание исполнителю совершить последовательность действий, направленных на решение поставленной задачи.

Основные свойства алгоритмов следующие:

Понятность для исполнителя — т.е. исполнитель алгоритма должен знать, как его выполнять.

Дискретность (прерывность, раздельность) — т.е. алгоритм должен представлять процесс решения задачи как последовательное выполнение простых (или ранее определенных) шагов (этапов).

Определенность — т.е. каждое правило алгоритма должно быть четким, однозначным и не оставлять места для произвола. Благодаря этому свойству выполнение алгоритма носит механический характер и не требует никаких дополнительных указаний или сведений о решаемой задаче.

Результативность (или конечность). Это свойство состоит в том, что алгоритм должен приводить к решению задачи за конечное число шагов.

Массовость. Это означает, что алгоритм решения задачи разрабатывается в общем виде, т.е. он должен быть применим для некоторого класса задач, различающихся лишь исходными данными. При этом исходные данные могут выбираться из некоторой области, которая называется областью применимости алгоритма.

На практике наиболее распространены следующие формы представления алгоритмов:

- **словесная** (записи на естественном языке);
- **графическая** (изображения из графических символов);
- **псевдокоды** (полуформализованные описания алгоритмов на условном алгоритмическом языке, включающие в себя как элементы языка программирования, так и фразы естественного языка, общепринятые математические обозначения и др.);
- **программная** (тексты на языках программирования).

Решение любой задачи на ЭВМ можно разбить на следующие этапы: разработка алгоритма решения задачи, составление программы решения задачи на алгоритмическом языке, ввод программы в ЭВМ, отладка программы (исправление ошибок), выполнение программы на ПК, анализ полученных результатов.

Первый этап решения задачи состоит в разработке алгоритма.

Алгоритм может быть описан одним из трех способов:

- словесным (пример в начале раздела);
- графическим (виде специальной блок-схемы);
- с помощью специальных языков программирования.

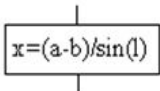
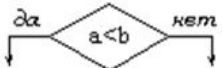
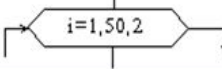
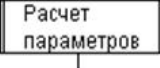
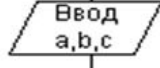
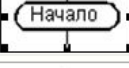

Графическая форма записи, называемая также схемой алгоритма, представляет собой изображение алгоритма в виде последовательности связанных между собой функциональных блоков, каждый из которых соответствует выполнению одного или

нескольких действий. Графическая запись является более компактной и наглядной по сравнению со словесной. В схеме алгоритма каждому типу действий соответствует геометрическая фигура. Фигуры соединяются линиями переходов, определяющими очередность выполнения действий.

Графическая форма записи, называемая также структурной схемой или блок-схемой алгоритма, представляет собой изображение алгоритма в виде последовательности связанных между собой функциональных блоков, каждый из которых соответствует выполнению одного или нескольких действий.

В дальнейшем мы будем использовать *блок-схемы алгоритмов*. Они позволяют представить алгоритмы в более наглядном виде, это дает возможность анализировать их работу, искать ошибки в их реализации и т.д. В блок-схемах всегда есть *начало* и *конец*, обозначаемые эллипсами, между ними - последовательность *шагов* алгоритма, соединенных *стрелками*.

Таблица 1. Обозначение основных алгоритмических конструкций

Название символа	Обозначение и пример заполнения	Пояснение
Процесс		Вычислительное действие или последовательность действий
Решение		Проверка условий
Модификация		Начало цикла
Предопределенный процесс		Вычисления по подпрограмме, стандартной подпрограмме
Ввод-вывод		Ввод-вывод в общем виде
Пуск-останов		Начало, конец алгоритма, вход и выход в подпрограмму
Документ		Вывод результатов на печать

Блок **"процесс"** применяется для обозначения действия или последовательности действий, изменяющих значение, форму представления или размещения данных. Для улучшения наглядности схемы несколько отдельных блоков обработки можно объединять в один блок. Представление отдельных операций достаточно свободно.

Блок **"решение"** используется для обозначения переходов управления по условию. В каждом блоке "решение" должны быть указаны вопрос, условие или сравнение, которые он определяет.

Блок **"модификация"** используется для организации циклических конструкций. (Слово модификация означает видоизменение, преобразование). Внутри блока записывается параметр цикла, для которого указываются его начальное значение, граничное условие и шаг изменения значения параметра для каждого повторения.

Блок "**предопределенный процесс**" используется для указания обращений к вспомогательным алгоритмам, существующим автономно в виде некоторых самостоятельных модулей, и для обращений к библиотечным подпрограммам.

Блок-схема – распространенный тип схем, описывающий алгоритмы или процессы, изображая шаги в виде блоков различной формы, соединенных между собой стрелками.

1. **Линейный алгоритм** – это такой алгоритм, в котором все операции выполняются последовательно одна за другой.

2. **Алгоритмы разветвленной структуры** применяются, когда в зависимости от некоторого условия необходимо выполнить либо одно, либо другое действие.

3. **Алгоритмы циклической структуры.**

Циклом называют повторение одних и тех же действий (шагов).

Последовательность действий, которые повторяются в цикле, называют **телом цикла**.

Циклические алгоритмы подразделяют на алгоритмы с предусловием, постусловием и алгоритмы с конечным числом повторов. В алгоритмах с предусловием сначала выполняется проверка условия окончания цикла и затем, в зависимости от результата проверки, выполняется (или не выполняется) так называемое тело цикла.

Ветвление — алгоритм, в котором предусмотрены разветвления, указанные в последовательности действий на два направления в зависимости от итогов проверки заданного условия. То есть такой алгоритм, обязательно содержит условие и в зависимости от результата выполнения условия происходит выбор действия.

Например: Если день рабочий, то идем в лицей, иначе будем отдыхать. Таких примеров можем привести много из обычной жизни и наук. К примеру, физика: **Если** удар упругий, **то** масса тела сохраняется, **иначе** масса изменяется.

Алгоритмическая конструкция ветвление программируется с помощью **условного оператора If**, который может быть представлен двумя вариантами (Таблица 1).

Условный оператор If

Таблица 1

Конструкция	Графическое представление блок - схема)
1 Вариант — неполное ветвление	
<p style="text-align: center;">If <условие> Then <оператор></p> <p>Неполное ветвление — в зависимости от результата проверки условия либо выполняются действия одной ветви «да» (оператор), либо эти действия не выполняются.</p>	
2 Вариант — полное ветвление	
<p>If <условие> Then <оператор 1> Else <оператор 2></p> <p>Полное ветвление — в зависимости от результата проверки условия выполняются только оператор 1 ветви «да» или только оператор 2 ветви «нет»</p>	

Условие — это логическое выражение, которое может принимать одно из двух значений: true (истина — условие выполняется) и false (ложь — условие не выполняется).

В условии используются операции отношения (=, <, >, <=, >=) и логические операции (and (И), or (ИЛИ), xor (исключающее ИЛИ), not (отрицание)). Если требуется проверить несколько условий, их объединяют с помощью логических операций. Примеры логических выражений:

A < 2 ;

(x < 0) and (y < 0).

Если между служебными словами стоят несколько операторов, то они заключаются в операторные скобки Begin...End

Рассмотрим пример:

Даны 2 вещественных числа. Если числа положительные, то возвести в квадрат первое число, иначе возвести в квадрат второе число.

Листинг программы	Графическое представление (блок -схема)
<pre> Program Primer_1; Uses crt; {Обозначим 1-ое число через переменную a, 2-ое через переменную b, результат — c} Var a, b, c: real; Begin Clrscr; Writeln('Введите первое число '); Readln(a); Writeln('Введите второе число '); Readln(b); If (a>0) and (b>0) Then c:=sqr(a) Else c:=sqr(b); Writeln('Результат = ', c:5:2); End. </pre>	<pre> graph TD Start([Начало]) --> Input[/a, b/] Input --> Decision{b > 0} Decision -- да --> CalcA[c = a²] Decision -- нет --> CalcB[c = b²] CalcA --> Merge(()) CalcB --> Merge Merge --> Output[c] Output --> End([Конец]) </pre>

Правила пунктуации при записи операторов

1. Точка с запятой не ставится в разделах описаний после зарезервированных слов `uses`, `label`, `type`, `const`, `var` и ставится после завершения каждого описания
2. Точка с запятой не ставится после `begin` и перед `end`, так как эти слова являются операторными скобками, а не операторами;
3. Точка с запятой является разграничителем операторов, ее отсутствие между операторами вызывает ошибку компиляции;
4. В операторах цикла точка с запятой не ставится после служебных слов `while`, `repeat`, `do`, `until`;
5. В условных оператора точка с запятой не ставится после `then` и перед `else`

Любая ветвь может не быть линейным участком программы, а сама содержать ветвление. Такое ветвление называется вложенным (или множественным) ветвлением.

Чаще вторично разветвляется ветка «нет». В качестве примера разберём простую задачу:

В первом магазине хозяйка приобрела a кг. огурцов. Их оказалось b штук. Во втором магазине на c кг. получилось d штук. В каком магазине огурцы крупнее?

Находим массу одного огурца в каждом магазине и сравниваем их. Блок-схема данной задачи представлена ниже.

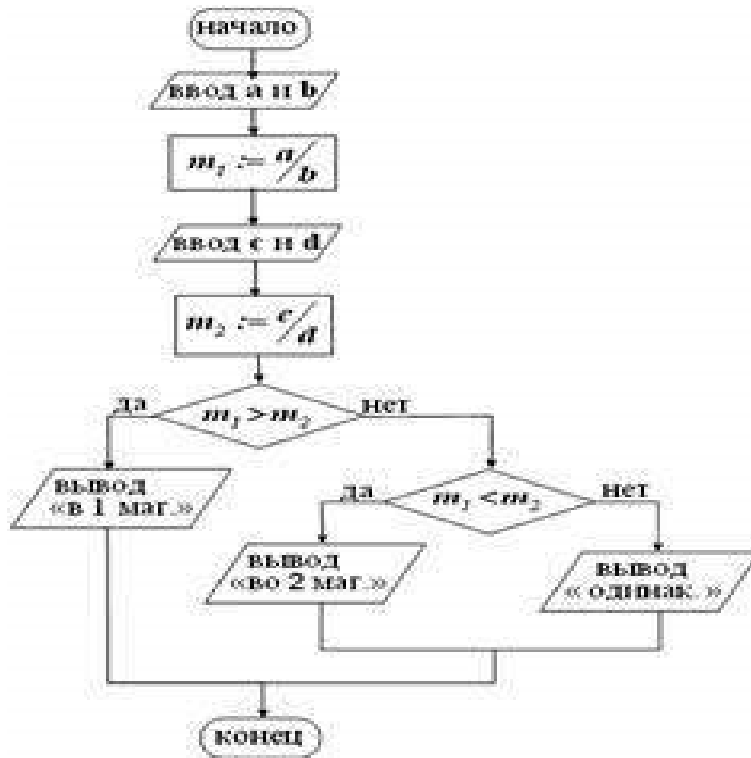


Рис. 3

Рассмотрим пример 2

Написать программу, которая вычисляет оптимальный вес пользователя, сравнивает его с реальным и выдает рекомендацию о необходимости поправиться или похудеть.

Оптимальный вес вычисляется по формуле: **рост (в сантиметрах) — 100.**

Рекомендуемый вид экрана во время работы программы приведен ниже:

*Введите в одной строке через
пробел рост (см) и вес (кг) затем
нажмите <Enter>*

-> 170 68

Вам надо поправиться на 2.00 кг.

Листинг программы

```

Program Ves;
var
w: real;    {вес} h: real; {рост}
opt: real;  {оптимальный вес}
d: real;    {отклонение от оптимального веса}
begin writeln('Введите в одной строке через
пробел'); writeln('рост (см) и вес (кг), затем
нажмите <Enter>'); write('->'); readln(h,w);
opt:=h-100;
    if w=opt then writeln('Ваш вес оптимален!') else
  
```

```
if w<opt then begin d:=opt-w; writeln('Вам надо поправиться на ',d:5:2,' кг.');
```

```
end else begin d:=w-opt; writeln('Вам надо похудеть на1, d:5:2,' кг'); end;
```

```
end.
```

ЗАДАНИЕ

Выполните задание по варианту, назначенному преподавателем.

Вариант 1

Задание 1

Даны три действительные числа. Возвести в квадрат те из них, значения которых положительны, и в четвертую степень — отрицательные.

Задание 2

Написать программу, которая вычисляет частное от деления двух чисел. Программа должна проверять правильность введенных пользователем данных и, если они неверные (делитель равен нулю), выдавать сообщение об ошибке. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление частного.

Введите в одной строке делимое и делитель,затем нажмите <Enter>

-> 12 0

Вы ошиблись. Делитель не должен быть равен нулю.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 2

Задание 1

Даны два действительные числа. Если числа положительны найти их сумму, если отрицательны — произведение

Задание 2

Написать программу вычисления стоимости покупки с учетом скидки. Скидка в 10% предоставляется, если сумма покупки больше 1000 руб. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление стоимости покупки с учетом скидки.

Введите сумму покупки и нажмите <Enter>

-> 1200

Вам предоставляется скидка 10%

Сумма покупки с учетом скидки: 1080.00 руб.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 3

Задание 1

Даны действительные числа x и y, не равные друг другу. Меньшее из этих чисел заменить половиной их суммы, а большее — их удвоенным произведением

Задание 2

Написать программу проверки знания даты основания Санкт-Петербурга. В случае неверного ответа пользователя программа должна выводить правильный ответ. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

В каком году был основан Санкт-Петербург?

Введите число и нажмите <Enter>

-> 1705

Вы ошиблись, Санкт-Петербург был основан в 1703 году.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 4

Задание 1

Данные два вещественных числа. Если первое число больше второго, то возвести его в третью степень, если равно второму — прибавить к нему второе число

Задание 2

Написать программу определения стоимости разговора по телефону с учетом скидки 20%, предоставляемой по воскресеньям. Ниже представлен рекомендуемый вид экрана программы во время ее работы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление стоимости разговора по телефону.

Введите исходные данные:

*Длительность разговора (целое количество минут) —
> 3 День недели (1 - понедельник, ... 7 — воскресенье) —> 6
Предоставляется скидка 20%.*

Стоимость разговора: 5.52 руб.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 5

Задание 1

Даны три действительные числа. Если первое число больше второго, умножить данное число на 5, если первое число больше третьего — разделить на два

Задание 2

Написать программу — модель анализа пожарного датчика в помещении, которая выводит сообщение «Пожароопасная ситуация», если температура в комнате превысила 60°C

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 6

Задание 1

Даны действительные числа a , b , c . Удвоить эти числа, если $a \geq b < c$, иначе оставить без изменения.

Задание 2

Написать программу, которая анализирует данные о возрасте и относит человека к одной из четырех групп: дошкольник, ученик, работник, пенсионер. Возраст вводится с клавиатуры.

Задание 3

Оформить отчет оформить в тетради. Отчет должен содержать коды программ и блок-схемы.

Задание 1

Даны три числа a , b , c . Определить какое из них равно d . Если ни одно не равно d , то найти сумму чисел a , b , c .

Контрольные вопросы:

1. Что такое алгоритм?
2. Назовите основные свойства алгоритма.
3. Какие существуют формы представления алгоритма?

Кратко охарактеризуйте их.

4. Чем отличается графическая форма представления алгоритма от других форм?
5. С помощью какой фигуры изображается этап вычисления? Проверка условия? Вывод данных?
6. Какие преимущества дает блок-схема?
7. Назовите три основных структуры алгоритмов.
8. В чем отличие полного ветвления от неполного?
9. Какой цикл называется итерационным? рекурсивным?
10. В чем отличие цикла с предусловием от цикла с постусловием?

Оформление результатов работ

1. Оформить работу в соответствии с заданиями.
2. Ответить на контрольные вопросы.
3. Сформулировать выводы по результатам работы.
4. Сдать и защитить работу.