

Разработка разветвляющихся алгоритмов

- Алгоритм, в котором в зависимости от условия выполняется либо одна, либо другая последовательность действий

Стадии создания алгоритма:

- 1. Алгоритм должен быть представлен в форме, понятной человеку, который его разрабатывает (определить цель, наметить план действий).
- 2. Алгоритм должен быть представлен в форме, понятной тому объекту (в том числе и человеку), который будет выполнять описанные в алгоритме действия (выбрать среду и объект алгоритма, детализировать алгоритм).

Объект, который будет выполнять алгоритм, обычно называют исполнителем.

Исполнитель - объект, который выполняет алгоритм.

Назначение исполнителя точно выполнить предписания алгоритма, подчас не задумываясь о результате и целях, т.е. формально. Идеальными исполнителями являются машины, роботы, компьютеры...

Компьютер – автоматический исполнитель алгоритмов.

Алгоритм, записанный на «понятном» компьютеру языке программирования, называется программой.

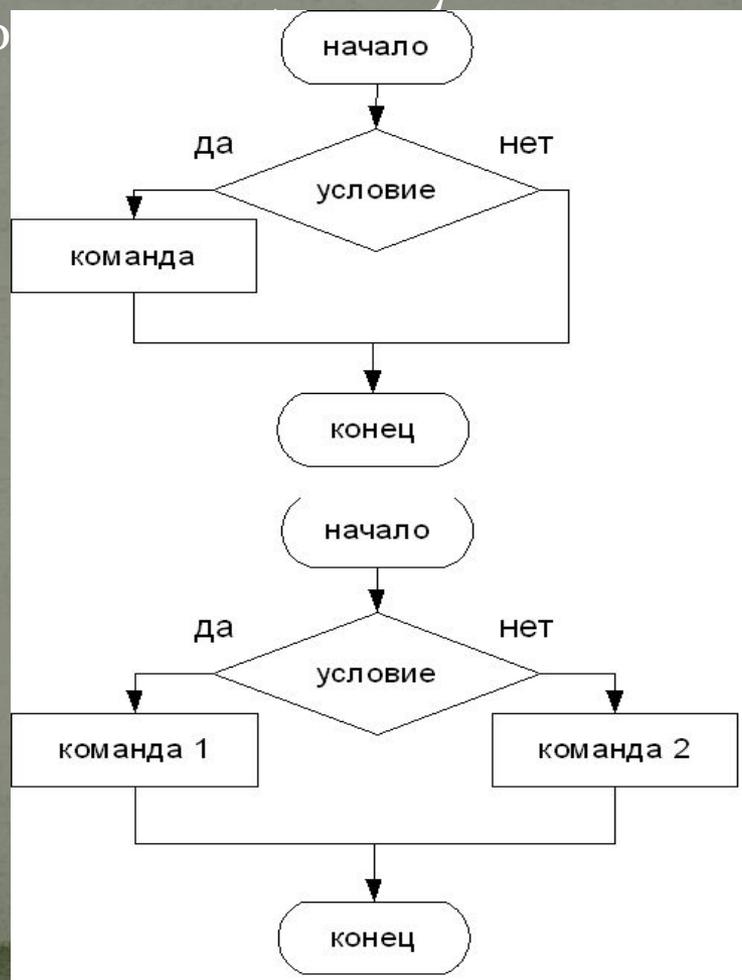
Разветвляющийся алгоритм

- Во многих случаях требуется, чтобы при одних условиях выполнялась одна последовательность действий, а при других – другая.



- **Условие** – это высказывание которое может быть либо истинно, либо ложно.

Еще раз обратим внимание, что существует две формы ветвления – неполная (когда присутствует только одна ветвь, т.е. в зависимости от истинности условия либо выполняется, либо не выполняется действие) и полная (когда присутствуют две ветви, т.е. в зависимости от истинности условия выполняется либо одно, либо другое)



Признаки

- Признаком разветвляющегося алгоритма является наличие операций проверки условия. Различают два вида условий – *простые* и *составные*.
- *Простым условием (отношением)* называется выражение, составленное из двух арифметических выражений или двух текстовых величин (иначе их еще называют *операндами*), связанных одним из знаков:
 - $<$ - меньше, чем...
 - $>$ - больше, чем...
 - $<=$ - меньше, чем... или равно
 - $>=$ - больше, чем... или равно
 - $<>$ - не равно
 - $=$ - равно

- Выражения, при подстановке в которые некоторых значений переменных, о нем можно сказать истинно (верно) оно или ложно (неверно) называются *булевыми* (логическими) выражениями.
- Название “булевые” произошло от имени математика Джорджа Буля, разработавшего в XIX веке булеву логику и алгебру логики.

Примеры

● Вычислить значение модуля и квадратного корня из выражения $(x-y)$.

Для решения этой задачи нужны уже знакомые нам стандартные функции нахождения квадратного корня - `Sqr` и модуля - `Abs`. Поэтому Вы уже можете записать следующие операторы присваивания:

```
Koren:=Sqrt(x-y);
```

```
Modul:=Abs(x-y).
```

В этом случае программа будет иметь вид:

```
Program qq;
```

```
Uses
```

```
Crt;
```

```
Var
```

```
x, y : integer;
```

```
Koren, Modul : real;
```

```
Begin
```

```
ClrScr;
```

```
write ('Введите значения переменных x и y через пробел ');
```

```
read (x, y);
```

```
Koren:=Sqrt(x-y);
```

```
Modul:=Abs(x-y).
```

```
write ('Значение квадратного корня из выражения (x-y) равно ');
```

```
write ('Значение модуля выражения (x-y) равно ');  
readln;  
End.
```

Каждая программа, насколько это возможно, должна осуществлять контроль за допустимостью величин, участвующих в вычислениях. Здесь мы сталкиваемся с разветвлением нашего алгоритма в зависимости от условия. Для реализации таких условных переходов в языке Паскаль используют операторы If и Else, а также оператор безусловного перехода Goto.

```
if x>=y  
Then  
Koren:=Sqr(x-y)  
Else
```

```
write ('Введены недопустимые значения переменных');
```

Теперь в зависимости от введенных значений переменных x и y, условия могут выполняться или не выполняться.

В общем случае полная форма конструкции условного оператора имеет вид:

```
if <логическое выражение>  
Then  
<оператор 1>  
Else  
<оператор 2>
```

Условный оператор работает по следующему алгоритму.

Сначала вычисляется значение логического выражения, расположенного за служебным словом IF. Если его результат **истина**, выполняется <оператор 1>, расположенный после слова THEN, а действия после ELSE пропускаются; если результат **ложь**, то, наоборот, действия после слова THEN пропускаются, а после ELSE выполняется <оператор 2>.

Если в качестве оператора должна выполняться серия операторов, то они заключаются в операторные скобки begin-end. Конструкция *Begin ... End* называется *составным оператором*.

Составной оператор -объединение нескольких операторов в одну группу. Группа операторов внутри составного оператора заключается в операторные скобки (begin-end).

```
Program qq;
Uses
  Crt;
Var
  x, y : integer;
  Koren, Modul : real;
Begin
  ClrScr;
  write ('Введите значения переменных x и y через пробел ');
  read (x, y);
  if x>=y
  Then
  Begin
    Koren:=Sqr(x-y)
    Modul:=Abs(x-y)
    write ('Значение квадратного корня из выражения (x-y) равно ');
    write ('Значение модуля выражения (x-y) равно ');
  End
  Else
  write ('Введены недопустимые значения переменных');
  readln;
End.
```



Спасибо за внимание! :)