По одной из классификаций, имеющей значение в программировании, алгоритмы делятся на следующие:

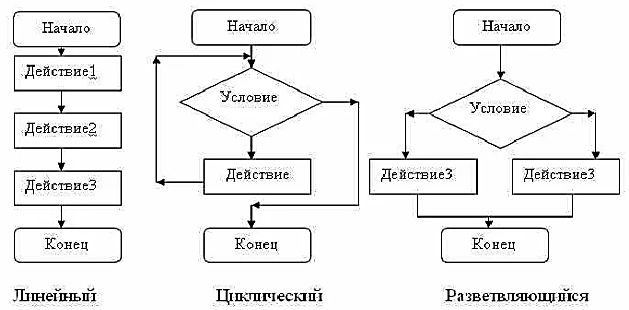
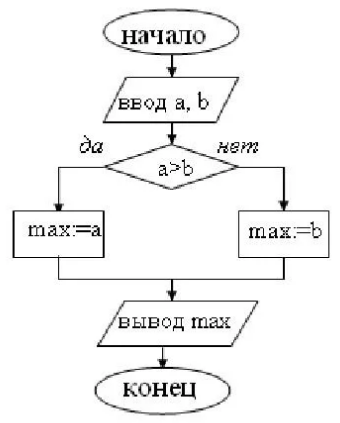
******

Рис 1. Виды алгоритмов

Рассмотрим более подробно разветвляющийся алгоритм или алгоритм с условиями.

***Разветвляющийся*** алгоритм – набор команд (указаний), среди которых имеется хотя бы одно условие. Когда условие встречается, то компьютер проверяет условие и в зависимости от результата проверки выполняет одну из ветвей алгоритма. Рассмотрим пример:



В этом примере определяется максимальное число.

На практике, подобные алгоритмы встречаются довольно часто, для самых разных случаев. Рассмотрим несколько алгоритмов с условиями.

**Деление двух чисел.**

Все знают, что при делении знаменатель не должен быть равным нулю, поэтому требуется проверка.

В общем случае, если это делать профессионально, то деление на ноль для программы рассматривается как «событие». В современных языках программирования имеются стандартные возможности для обработки таких «событий». Таким образом, если при делении знаменатель равен нулю, обработчик перехватывает это событие и сообщает пользователю.

Если же не использовать такие возможности, то при делении на ноль программа аварийно завершает вычисление. Это неприятная для пользователя ситуация. Пользователи не любят, когда программа «вылетает».

Но при делении двух чисел можно поступить и более просто. Выполнить введенного знаменателя на ноль. Если введен ноль, то выдать сообщение и попросить повторить ввод или закончить работу программы.

Составим алгоритм, для чего вначале введем переменные:

r – вещественное число для записи результата деления. После расчета выдается на экран. r объявляем вещественным, т.к. деление всегда дает вещественный результат (в программировании).

x – числитель, вещественное число. Вводится с клавиатуры.

y –знаменатель, вещественное число. Вводится с клавиатуры. Не равно нулю

Для записи алгоритма сначала используем псевдокод.

**АЛГ** Дел

**ВВОД** x

**ВВОД** y

**ЕСЛИ** y=0 **ТО** сообщение об ошибке и **КОНЕЦ**

**ИНАЧЕ** d**=**x/y

**ВЫВОД** d

**КОНЕЦ** Дел

Реализуем вычисление на С++

**#include <iostream> //библиотека для ввода-вывода**

**using namespace std; //пространство имен std**

**int main() //название программы без типа**

**{ //начало программы**

**float d , x, y;**

**cout << "x = ";**

**cin >> x; //ввод с клавиатуры в переменную x**

**cout << "y = ";**

**cin >> y; //ввод с клавиатуры в переменную y**

**if (y == 0)**

**{**

**cout << "not 0" << endl;**

**}**

**else**

**{**

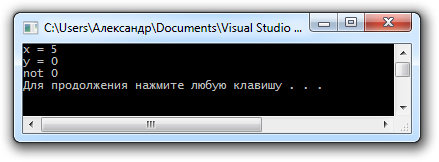
**d = x / y;**

**cout << "d = " << d << endl;**

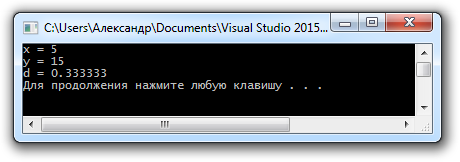
**}**

**}**

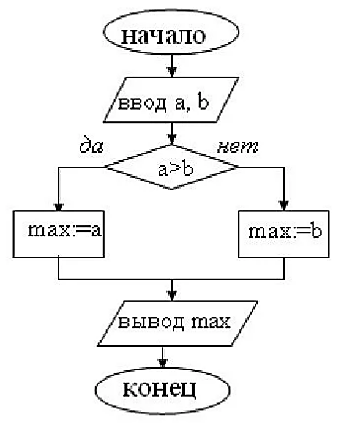
Результат при x=5 и y=0:



и при x=5 и y=15:



Рассмотрим еще один пример из начала лекции:



Реализация алгоритма на С++

**#include <iostream> //библиотека для ввода-вывода**

**using namespace std; //пространство имен std**

**int main()**

**{**

**int a , b, max;**

**cout << "a = ";**

**cin >> a;**

**cout << "b = ";**

**cin >> b;**

**if (a>b)**

**{max = a;}**

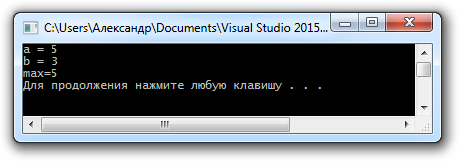
**else**

**{max = b;}**

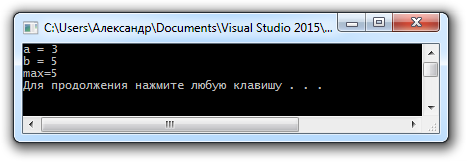
**cout << "max=" << max << endl;**

**}**

Результат для a=5 и b=3

****

Результат для a=3 и b=5



Примечание. В программе max=a и max=b заключены в фигурные скобки. В данном случае они необязательны. Но если в одной ветки условия два и более оператора, то они обязательны.

Таким образом структура условия такова:

**if** (условие)

{операторы если условие **true**}

**else**

{операторы если условие **false**}

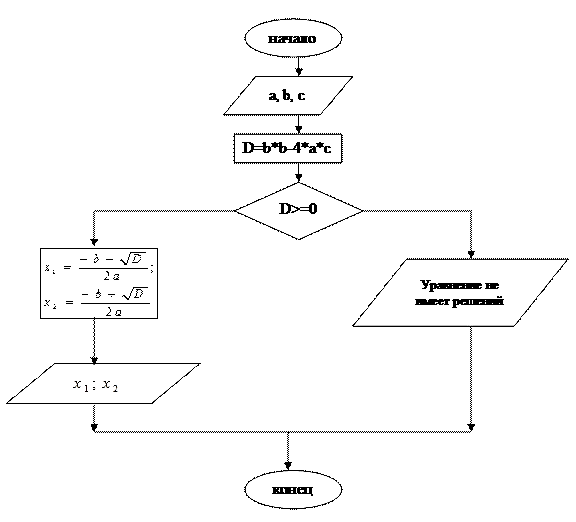
Операторами могут быть и другие условия. Операторов может быть несколько в одной ветке.

Допускаются и такие варианты:

**if** (условие)

{операторы если условие **true**}

Реализуем более сложный алгоритм с условием. Нахождение корней квадратного уравнения:

****

**#include <iostream> //библиотека для ввода-вывода**

**#include <cmath> //математическая библиотека С++**

**using namespace std; //пространство имен std**

**int main()**

**{**

**float a , b, c, D, x1, x2;**

**cout << "a = ";**

**cin >> a;**

**cout << "b = ";**

**cin >> b;**

**cout << "c = ";**

**cin >> c;**

**D = b\*b - 4 \* a\*c;**

**if (D >= 0)**

**{**

**x1=(-b+ sqrt(D))/2\*a;**

**x2= (-b - sqrt(D))/2\*a;**

**cout << "x1 = " << x1 << endl;**

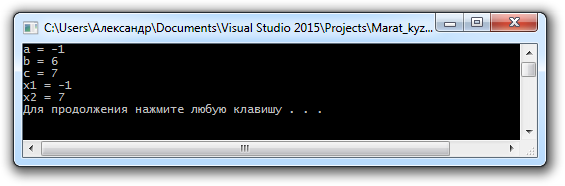
**cout << "x2 = " << x2 << endl;**

**}**

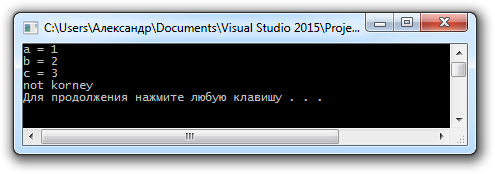
**else cout << "not korney" << endl;**

**}**

Корни существуют для а=-1, в=6 и с=7. Результат:



Корни НЕ существуют для а=1, в=2 и с=3. Результат:



В этой программе заказана дополнительная библиотека, для извлечения квадратного корня и в условии, в одной ветке присутствует несколько операторов. В таком случае фигурные скобки обязательны.

Примечание. В условии с++ знак равенства записывается как ==. Например, if (a==b). Запись if (a=b) будет ошибочной.

**Более сложный пример с условием и символами.**

Вводятся с клавиатуры два числа и символ операции.

s – если числа требуется сложить и v – если вычесть.

Введем переменные:

а – первое целое число

b – второе целое число

rez - сумма

op – символьная переменная для ввода операции

pr – логическая переменная. Если операция равна s или v, то pr=true. Если операция любой другой символ, то pr=false

**Блок-схема алгоритма**

Да

Да

Начало

Ввод

a, b, op

op= ‘s’

Rez=a+b,

Pr=true

op= ‘v’

Rez=a-b,

Pr=true

Pr=false

pr= true

Вывод

rez

Вывод

error

Да

Конец

Программа на с++

#include <iostream> //библиотека для ввода-вывода

using namespace std; //пространство имен std

int main()

{

int a , b, rez;

char op;

bool pr;

//ввод слагаемых

cout << "a = ";

cin >> a;

cout << "b = ";

cin >> b;

//ввод операции

cout << "Operacia = ";

cin >> op;

//проверка операции

if (op == 's')

{

rez = a + b;

pr = true;

}

else

if (op == 'v')

{

rez = a - b;

pr = true;

}

else pr = false;

//анализ результата

if (pr) cout << "Rezultat= " << rez << endl;

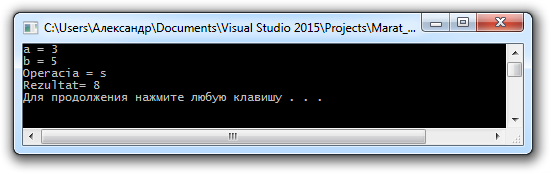
else cout << "Error" << endl;

system("PAUSE");

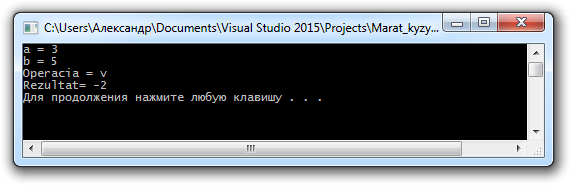
}

Для проверки проверяем три варианта: 1.сложение, 2.вычитание и 3.операция ошибочная:

1



2



3

