Программирование (Python)

Ветвление.

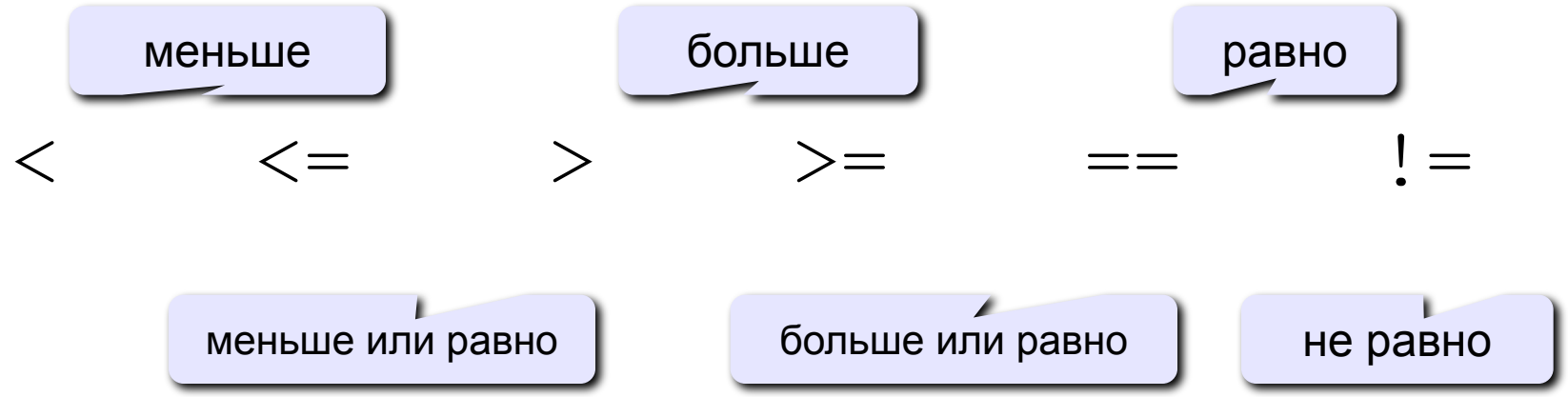
© К.Ю. Поляков, Е.А. Ерёмин, 2018 http://kpolyakov.spb.ru

Разветвляющиеся алгоритмы. Ветвление.

Алгоритмы, в которых последовательность шагов зависит от *выполнения* некоторых *условий*, называются разветвляющимися.

Условие - это логическое выражение, которое может содержать знаки отношений.

Знаки для записи условий



✔ Простые условия записываются в виде равенств или неравенств.

Примеры записи простых условий:

ПРИМЕРЫ

a!=b

a<=0

a+3*c==20

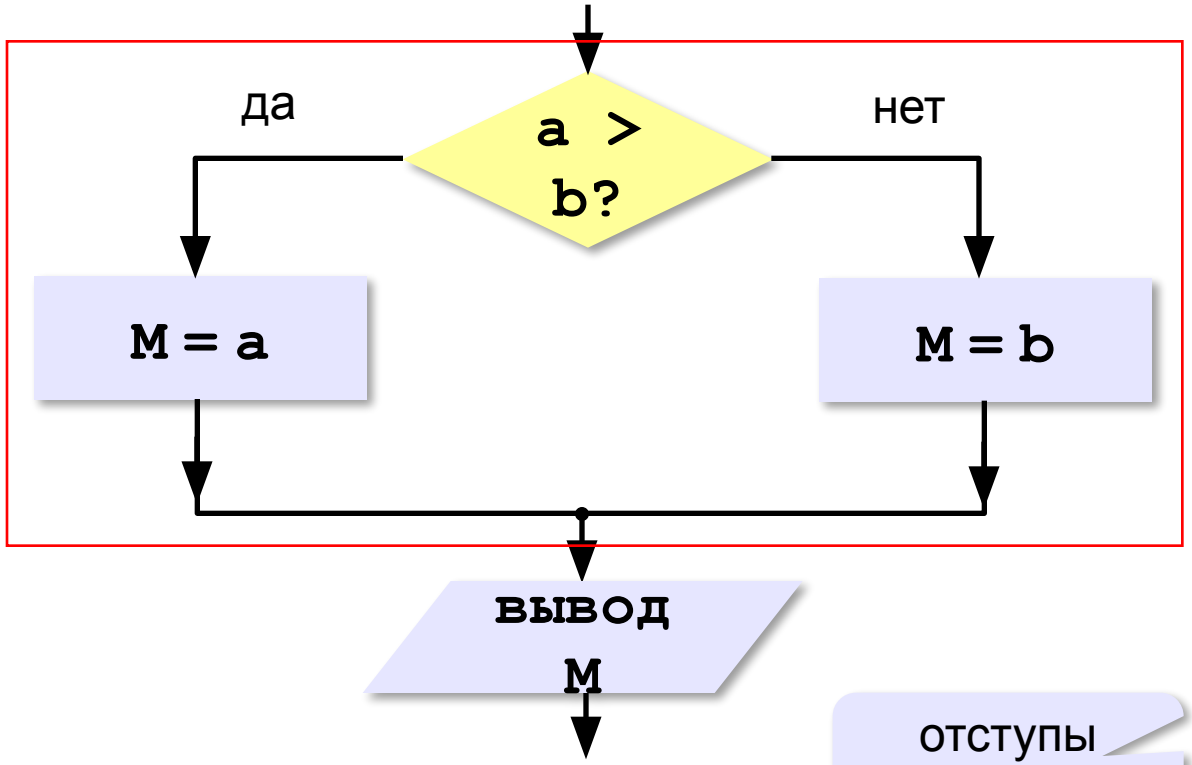
Разветвляющиеся алгоритмы

Для записи разветвляющихся алгоритмов в языке Python используют условный оператор IF – ELSE.

Оператор имеет две формы записи: полного и неполного ветвления.

Выбор наибольшего из двух чисел (вар.1)

Задача: **изменить порядок действий** в зависимости от выполнения некоторого условия.



полная форма ветвления

? Если a = b?

```
if a > b:
    M = a
else:
    M = b
```

отступы **4 пробела!!!**

Вариант 1. Программа (полное ветвление)

```
print("Введите два целых числа")
a = int(input())
b = int(input())
if a > b:
    M = a
else:
    M = b
print("Наибольшее число", M)
```

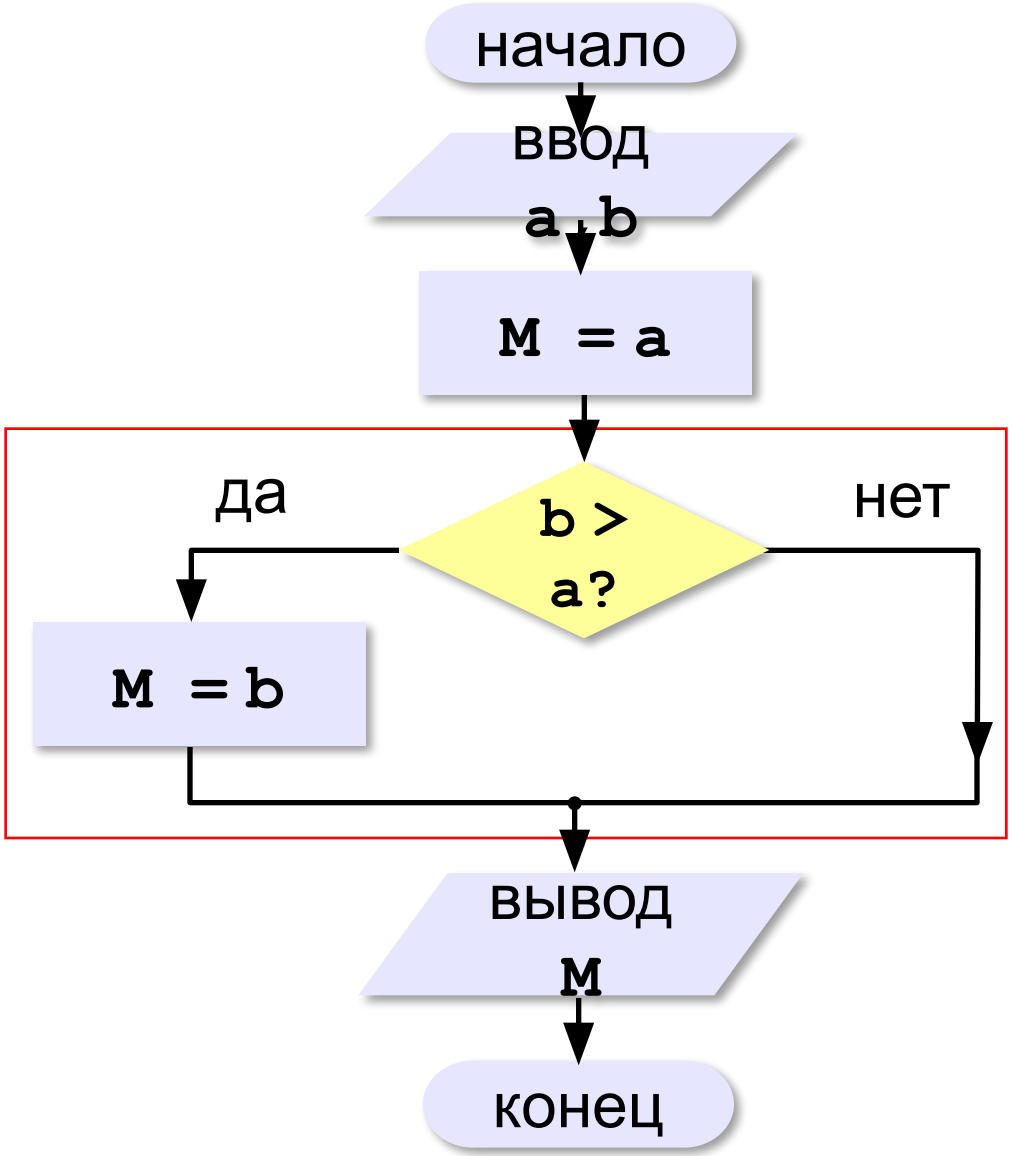
полная форма условного оператора

Решение с вызовом функции max и особой записью условного оператора :

```
M = max(a, b)
```

```
M = a if a > b else b
```

Выбор наибольшего из двух чисел (вар. 2)



неполная форма ветвления

Вариант 2. Программа (неполное ветвление)

```
print("Введите два целых числа")
a = int(input())
b = int(input())
M = a
if b > a:
    M = b
print("Наибольшее число", M)
```

неполная форма условного оператора

Примеры

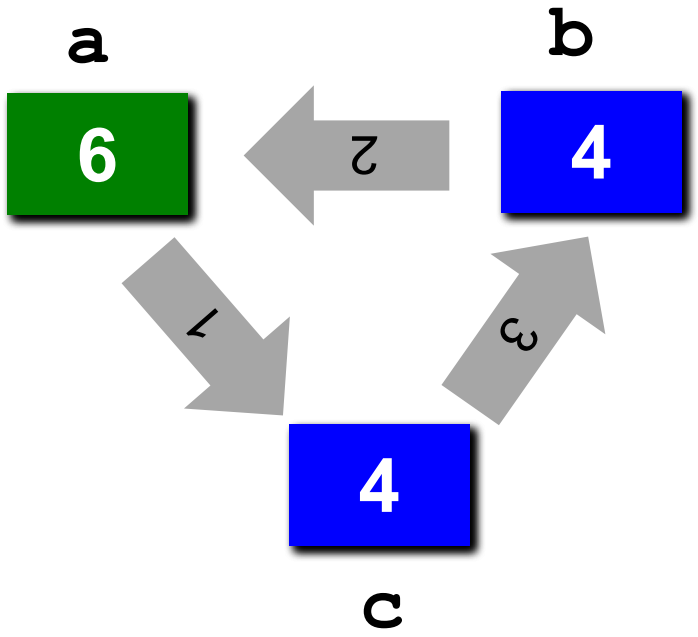
Поиск минимального:

```
if a < b:
  M = a
if b < a:
  M = b
```

Сортировка по убыванию

```
if a < b:
    c = a
    a = b
    b = c
```

? Что делает эта программа?



Запись решения с помощью множественного присваивания:

```
a, b = b, a
```

Вложенные условные операторы

Задача: в переменных **a** и **b** записаны возрасты Андрея и Бориса. Кто из них старше?

Сколько вариантов?

```
if a > b:
   print("Андрей старше")
else:
   if a == b:
      print("Одного возраста")
   else:
      print("Борис старше")
```

Зачем нужен?

вложенный условный оператор

Каскадное ветвление

```
if a > b:
  print("Андрей старше")
elif a == b:
  print("Одного возраста")
else:
  print("Борис старше")
```

```
elif = else if
```

Если после **else** сразу следует ещё один оператор **if**, можно использовать «каскадное» ветвление с ключевым словом **elif** (сокращение от else-if)

Каскадное ветвление

```
cost = 1500
if cost<1000:
  print( "Скидок нет." )
elif cost<2000:
  print( "Скидка 2%." )
elif cost<5000:
  print( "Скидка 5%." )
else:
  print( "Скидка 10%." )
```

первое сработавшее условие

? Что выведет?

```
Скидка 2%.
```

Знаки для записи условий

Сложное условие состоит из нескольких простых, связанных с помощью логических связок:

and – И (конъюнкция - одновременное выполнение условий)

or – ИЛИ (дизъюнкция - выполнение хотя бы одного из условий)

not – не (инверсия - отрицание условия)

ПРИМЕРЫ

x>5 and x<9

x>0 and x<=5

x>2 or x<1

a=0 or (b=0 and c=0)

not x > 5

Задачи (без функций min и max!)

«A»: Ввести два целых числа, найти наибольшее и наименьшее из них.

Пример:

```
Введите два целых числа:
1 5
Наибольшее число 5
Наименьшее число 1
```

«B»: Ввести четыре целых числа, найти наибольшее из них.

Пример:

```
Введите четыре целых числа:
1 5 4 3
Наибольшее число 5
```

Задачи

«С»: Ввести последовательно возраст Антона, Бориса и Виктора. Определить, кто из них старше.

Пример:

```
Возраст Антона: 15
Возраст Бориса: 17
Возраст Виктора: 16
Ответ: Борис старше всех.
```

Пример:

```
Возраст Антона: 17
Возраст Бориса: 17
Возраст Виктора: 16
Ответ: Антон и Борис старше Виктора.
```