Программирование на языке Python

1. Введение
2. Ветвления
3. Сложные условия
4. Циклы
5. Циклы с условием
6. Оператор выбора
7. Графика
8. Графики функций
9. Процедуры
10. Рекурсия
11. Анимация
12. Функции
13. Случайные числа

Программирование на языке Python

Тема 1. Введение

Алгоритм

Алгоритм – это четко определенный план действий для исполнителя.

Свойства алгоритма

- **дискретность**: состоит из отдельных шагов (команд)
- **понятность**: должен включать только команды, известные исполнителю (входящие в СКИ)
- **определенность**: при одинаковых исходных данных всегда выдает один и тот же результат
- **конечность**: заканчивается за конечное число шагов
- **массовость**: может применяться многократно при различных исходных данных
- **корректность**: дает верное решение при любых допустимых исходных данных

Программа

Программа – это

- алгоритм, записанный на каком-либо языке программирования
- набор команд для компьютера

Команда – это описание действий, которые должен выполнить компьютер.

- откуда взять исходные данные?
- что нужно с ними сделать?

Языки программирования

- **Машинно-ориентированные (низкого уровня)** - каждая команда соответствует одной команде процессора (ассемблер)
- **Языки высокого уровня** – приближены к естественному (английскому) языку, легче воспринимаются человеком, **не зависят от конкретного компьютера**
  - *для обучения*: Бейсик, Кумир, Паскаль, Python
  - *профессиональные*: Си, Python, Паскаль
  - *для задач робототехники и искусственного интеллекта*: *Пролог, ЛИСП, C++, Python*
  - *для Интернета*: HTML, CSS, *JavaScript, Java, Python, PHP, ASP*

Язык Python

1991 – разработан Гвидо ван Россумом

- **объектно-ориентированный язык**
- **успешно применяться для интернета**

Из чего состоит программа?

```
a=2
b=3
c=a+b
print(c)
```

Из чего состоит программа?

Константа – постоянная величина, имеющая имя, в питоне нет констант.

Переменная – изменяющаяся величина, имеющая имя (ячейка памяти).

Процедура – вспомогательный алгоритм, описывающий некоторые действия (рисование окружности).

Функция – вспомогательный алгоритм для выполнения вычислений (вычисление квадратного корня, **sin**).

Имена программы, констант, переменных

Имена могут включать

- латинские буквы (A-Z)

заглавные и строчные буквы не различаются

- цифры

имя не может начинаться с цифры

- знак подчеркивания _ (Кадр 6)

Имена НЕ могут включать

- русские буквы
- пробелы
- скобки, знаки +, =, !, ? и др.

Какие имена правильные??

AXby R&B 4Wheel Вася “PesBarbos”
TU154 [QuQu] _ABBA A+B

Переменные

Язык Python чувствителен к регистру. Переменная Z и z – разные переменные. Python, в отличие от многих языков, не требует описания переменных.

Типы переменных:

- int { целая }
- float { вещественная }
- list { список, аналог массивов}
- str { символьная строка }
- bool { логическая }

Объявление переменных (выделение памяти):

```
int(“88”) результат 88

str(88) результат “88”

float(88) результат 88.00
```

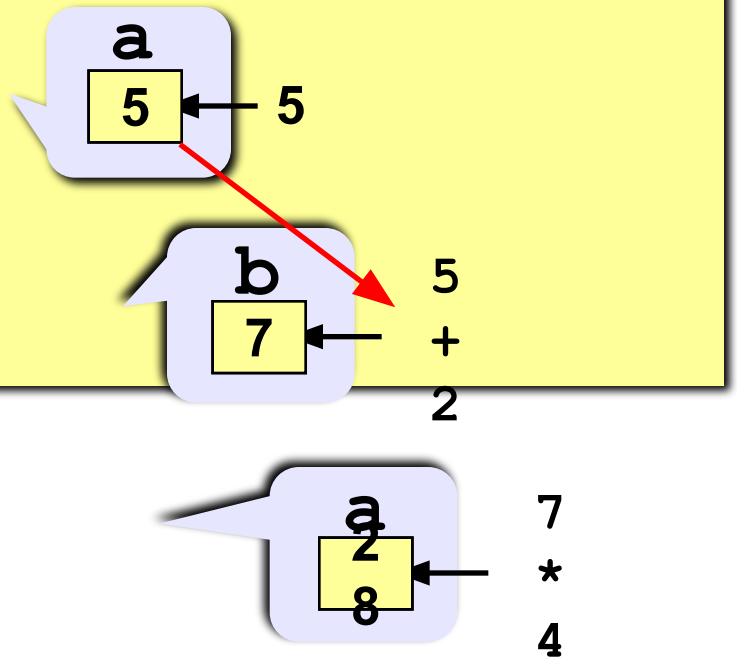
Как изменить значение переменной?

Оператор – это команда языка программирования высокого уровня.

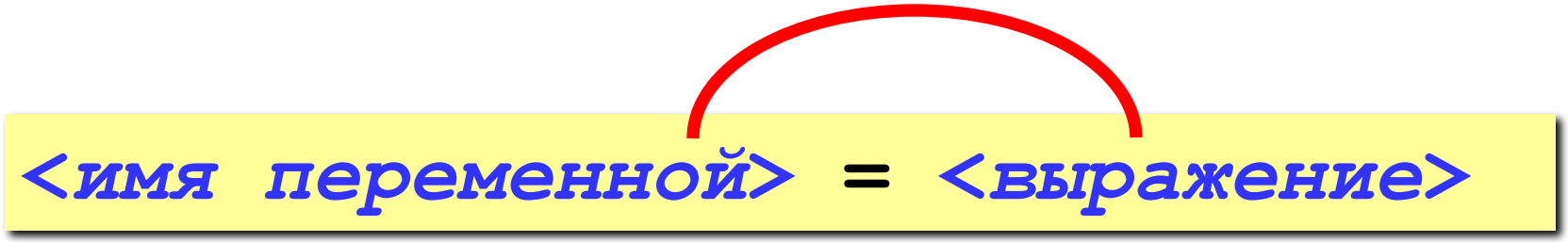
Оператор присваивания служит для изменения значения переменной.

Пример:

```
a=2
b=a+2
a=b*4
print(a)
```



Оператор присваивания



Арифметическое выражение может включать

- имена переменных
- знаки арифметических операций:

`+ -  *  /  //  %  **`

умножение

деление

деление нацело

остаток от деления

Возведение в степень

- вызовы функции
- круглые скобки ( )

Какие операторы неправильные?

```
a := 5
10 = x;
y  = 7,8
b  = 2.5
x  = 2*(a + y)
a  = b + x
```

имя переменной должно быть слева от знака **=**

целая и дробная часть отделяются **точкой**

Ручная прокрутка программы

```
a = 5
b = a + 2
a = (a + 2)*(b - 3)
b = a // 5
a = a % b
a = a + 1
b = (a + 14) % 7
```

a	b
?	?

```
a=3
b=4
c=6
d=8
z = (5*a*c+3*(c-d))/a*(b-c)/ b
print (z)
```

Порядок выполнения операций

- вычисление выражений в скобках
- умножение, деление, //, % слева направо
- сложение и вычитание слева направо

```
          2  3  5  4    1      7  8    6
z := (5*a*c+3*(c-d))/a*(b-c)/ b;
```

$$z = \frac{5ac + 3(c-d)}{ab}(b-c)$$

$$x = \frac{a^2 + 5c^2 - d(a+b)}{(c+d)(d-2a)}$$

```
          2  6  3  4  7  5    1      12    8  11  10
x:=(a*a+5*c*c-d*(a+b))/((c+d)*(d-2*a));
```

Оператор ввода

```
a=input( )      { ввод значения
                  переменной a}
```

```
print('Введите число')
a=input()
c = a + 2
print ( 'c=", c )
```

Протокол:

компьютер

```
Введите число
25
c=27
```

пользователь

Оператор вывода

```
print ( a )     { вывод значения
                  переменной a}

print (' Hello!') { вывод текста}


print ( 'Otvet: ', c )    { вывод
          текста и значения переменной c}

print ( a+b ) { вывод суммы чисел }
```

Сложение двух чисел

Задача. Ввести два целых числа и вывести на экран их сумму.

Простейшее решение:

```
a=int(input())
b=int(input()}
c = a + b
print ( c )
```

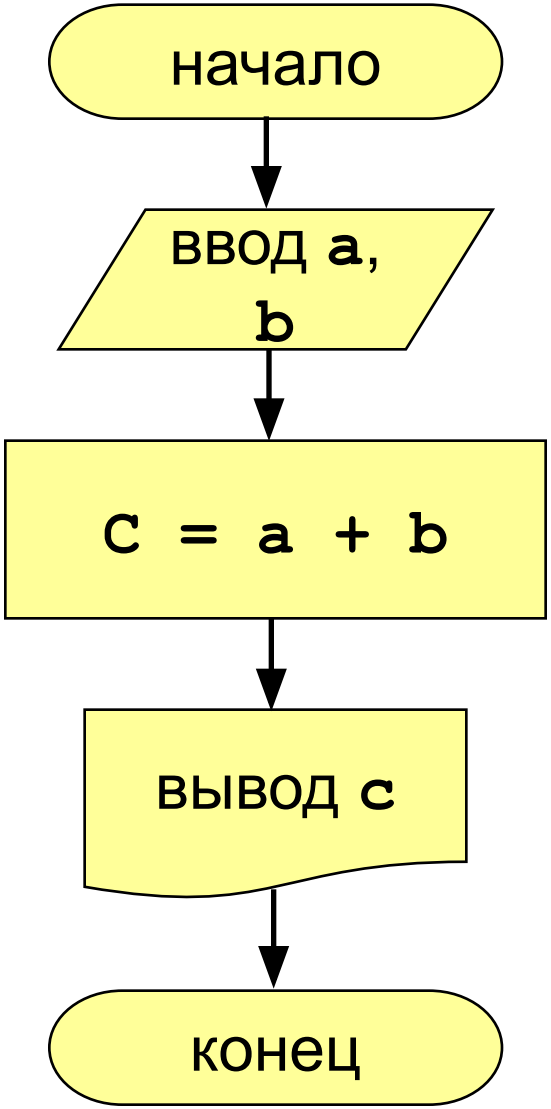
Форматирование вывода

```
import math
x=math.pi
print ("%.50f" % (x))
```

всего символов

в дробной части

Блок-схема линейного алгоритма



блок «начало»

блок «ввод»

блок «процесс»

блок «вывод»

блок «конец»

Задания

«4»: Ввести три числа, найти их сумму и произведение.

Пример:

```
Введите три числа:
4   5   7
4+5+7=16
4*5*7=140
```

«5»: Ввести три числа, найти их сумму, произведение и среднее арифметическое.

Пример:

```
    Введите три числа:
4   5   7
4+5+7=16
4*5*7=140
(4+5+7)/3=5.33
```

Программирование на языке Python

Тема 2. Ветвления

© С. Ефимов, 2013

Разветвляющиеся алгоритмы

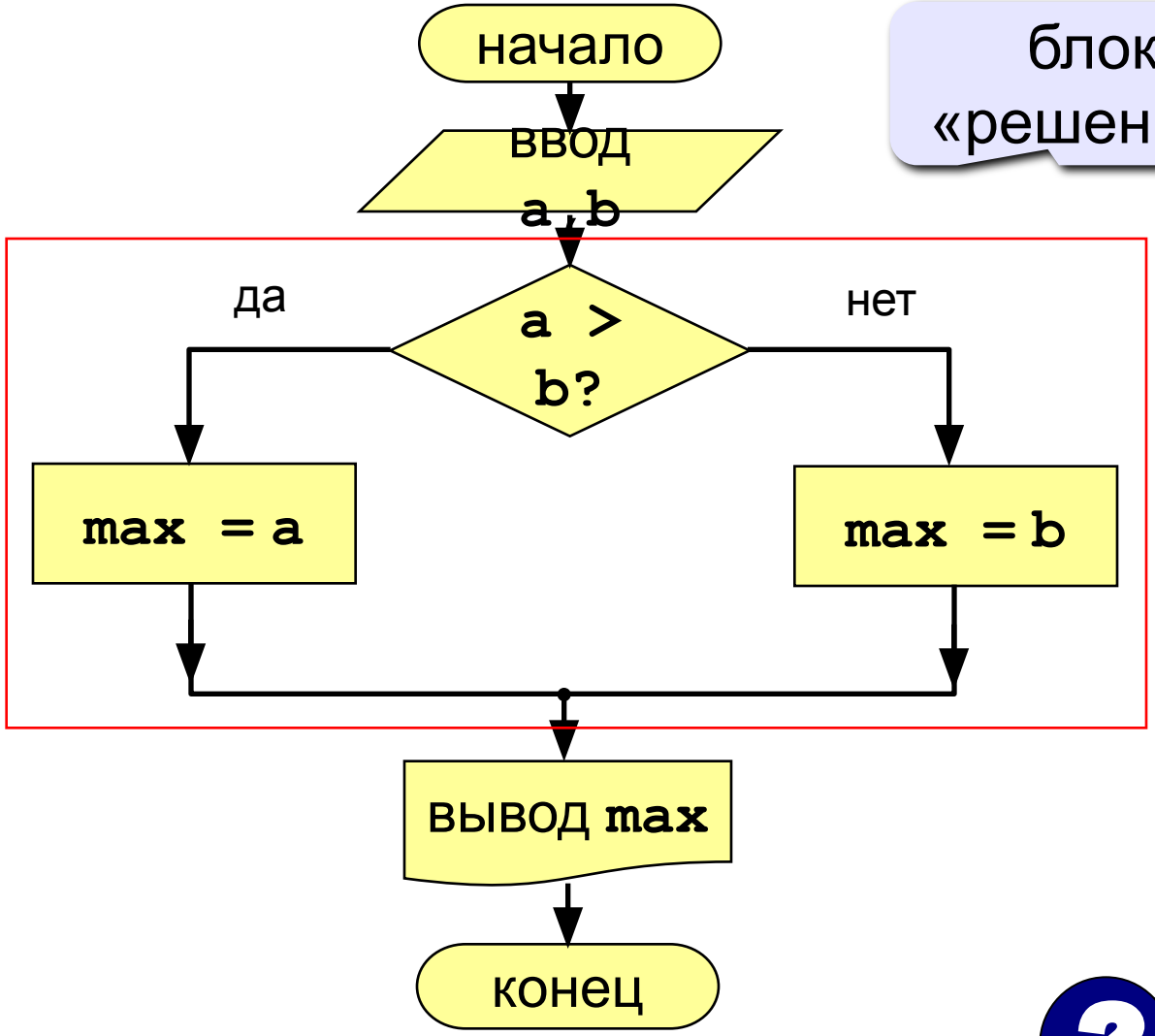
Задача. Ввести два целых числа и вывести на экран наибольшее из них.

Идея решения: надо вывести на экран первое число, если оно больше второго, или второе, если оно больше первого.

Особенность: действия исполнителя зависят от некоторых условий (***если … иначе …***).

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

Вариант 1. Блок-схема



блок «решение»

полная форма ветвления

? Если a = b?

Вариант 1. Программа

```
print(‘input 2 number‘)
a=int(input())
b=int(input())
if a > b :
    max = a

else :
    max = b

print (‘Max number', max)
```

Условный оператор

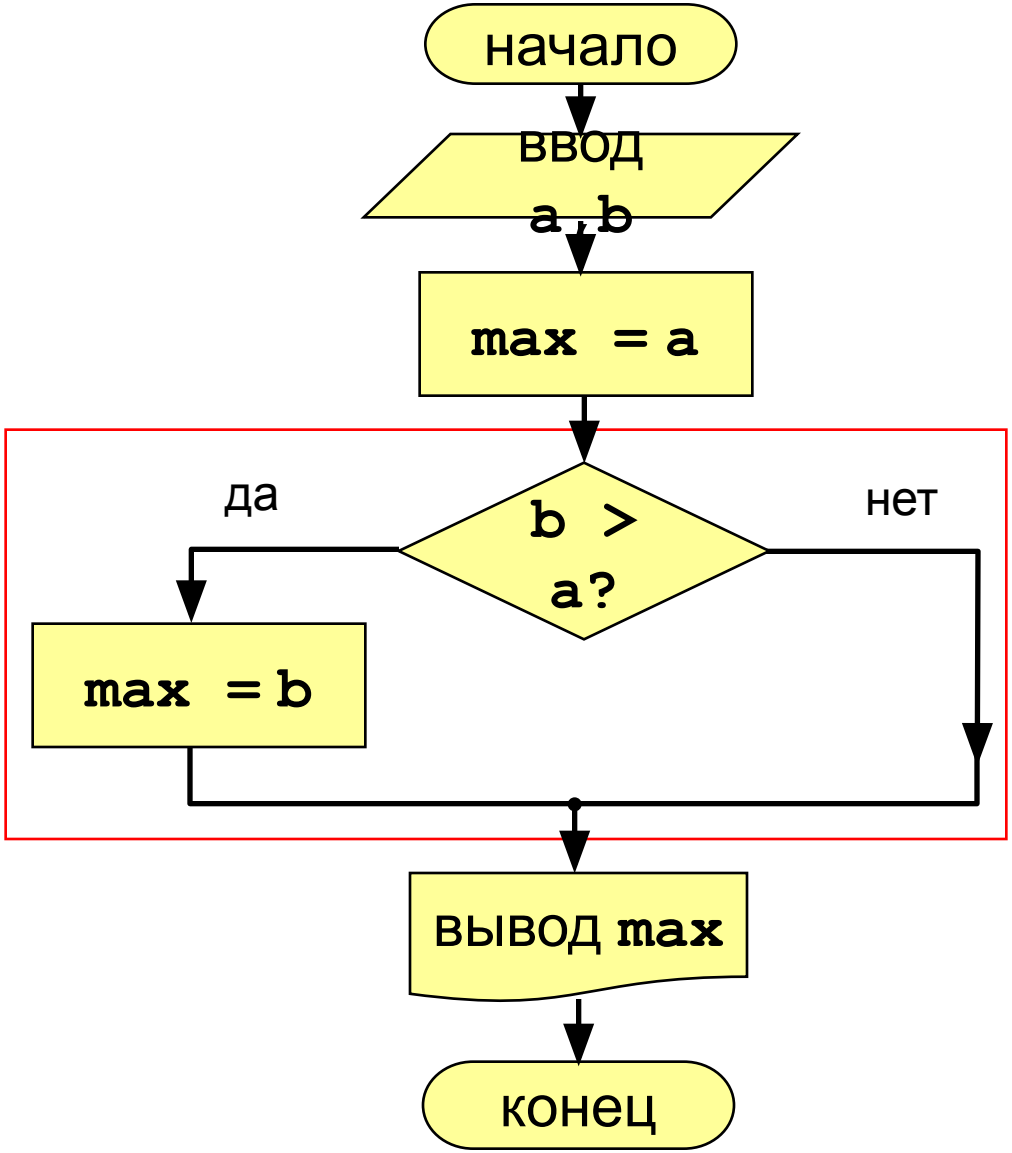
```
if <условие> :
   {что делать, если условие верно}

else :
   {что делать, если условие неверно}
```

Особенности:

- вторая часть (***else*** …) может отсутствовать (неполная форма)

Вариант 2. Блок-схема



неполная форма ветвления

Вариант 2. Программа

```
print('input 2 number')
a=input()
b=input()
max = a
if b > a :
   max = b
print ('Max number', max)
```

неполная форма условного оператора

Задания

«4»: Ввести три числа и найти наибольшее из них.

Пример:

```
Введите три числа:
4   15   9
Наибольшее число 15
```

«5»: Ввести пять чисел и найти наибольшее из них.

Пример:

```
Введите пять чисел:
4    15    9    56    4
Наибольшее число 56
```