

# Программирование (Python)

## § 17. Введение

# Что такое программирование?

**Программирование** — это создание программ для компьютеров. Этим занимаются **программисты**.

Чем занимаются **программисты**:

**анализ задачи** (выделение исходных данных, связей между ними, этапов решения задачи)

системные аналитики

разработка **алгоритмов**

алгоритмисты

написание и отладка **программ**

кодировщики

**тестирование** программ

тестировщики

написание **документации**

технические писатели

# Направления в программировании

---

**системный программист**

операционные системы,  
утилиты, драйверы

**прикладной программист**

прикладные программы, в  
т.ч. для мобильных  
устройств

**веб-программист**

веб-сайты

**программист баз данных**

системы управления  
базами данных

# Простейшая программа

```
# Это пустая программа
```

**?** Что делает эта программа?

комментарии после #  
не обрабатываются

кодировка utf-8  
по умолчанию)

```
# coding: utf-8
```

```
# Это пустая программа
```

```
"""
```

```
Это тоже комментарий
```

```
"""
```

# Вывод на экран

оператор  
вывода

**Оператор** — это команда  
языка программирования.

```
print ( "Привет!" )
```

```
print ( "Привет", Вася! )
```



Что плохо?

```
print ( "Привет, Вася!" )
```

вся строка в  
кавычках

## Переход на новую строку

---

```
print ( "Привет, Вася!" )  
print ( "Привет, Петя!" )
```

Результат:

```
Привет, Вася!  
Привет, Петя!
```

переход на новую строку автоматически

Нужно в одной строке:

```
Привет, Вася!Привет, Петя!
```

Решение:

```
print ( "Привет, Вася!" , end="" )  
print ( "Привет, Петя!" )
```

после вывода данных ничего не выводить

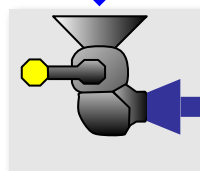
# Системы программирования

**Системы программирования** — это средства для создания новых программ.

**Транслятор** — это программа, которая переводит тексты программ, написанных программистом, в машинные коды (команды процессора).

- **компилятор** — переводит всю программу в машинные коды, строит исполняемый файл (**.exe**)

```
program Hello;  
begin  
  write('Привет!')  
end.
```



**privet.exe**

- **интерпретатор** — сам выполняет программу по частям (по одному оператору).



**Python – интерпретатор!**

# Системы программирования

**Отладчик** — это программа для поиска ошибок в других программах.

- **пошаговый режим** — выполнение программы по шагам (по одному оператору)
- **просмотр значений переменных** во время выполнения программы
- **точки останова** – операторы в программе, перед выполнением которых нужно остановиться.

**Среда программирования (IDE):**

- редактор текста программ
- транслятор
- отладчик



# Программирование (Python)

## § 18. Линейные программы

# Пример задачи

---

*Задача.* Ввести два числа и вычислить их сумму.

```
# ввести два числа  
# вычислить их сумму  
# вывести сумму на экран
```



Выполнится?

**Псевдокод** – алгоритм на русском языке с элементами языка программирования.



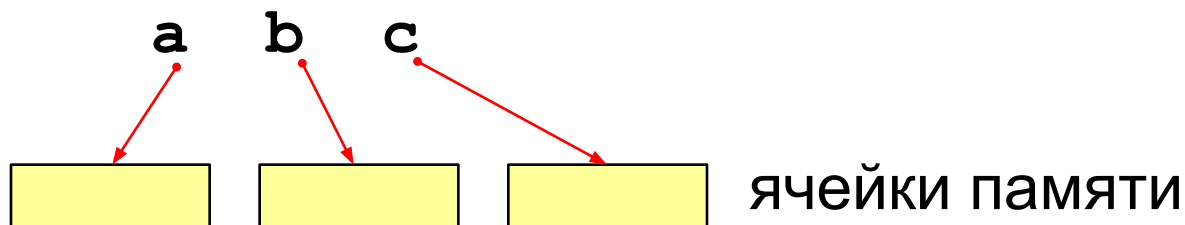
Компьютер не может исполнить псевдокод!

# Зачем нужны переменные?

```
# ввести два числа  
# вычислить их сумму  
# вывести сумму на экран
```

Где запомнить?

**Переменная** — это величина, которая имеет имя, тип и значение. Значение переменной может изменяться во время выполнения программы.



# Имена переменных

**Идентификатор** — это имя программы или переменной.

a b c

заглавные и строчные  
буквы **различаются**

**МОЖНО** использовать

- латинские буквы (A-Z, a-z)
- цифры



Имя не может начинаться с цифры!

- знак подчеркивания \_

**НЕЛЬЗЯ** использовать ~~скобки, знаки ", &, |, \*, +, =, !, ? и др.~~

Какие имена правильные?

**Axby R&B 4Wheel Вася "PesBarbos"**  
**TU154 [QuQu] \_ABBA A+B**

# Работа с переменными

## Присваивание (запись значения)

```
a = 5
```

оператор  
присваивания

$a \leftarrow 5$

```
a = 5  
a = 18
```

? Что будет храниться в *a*?

## Вывод на экран

```
print(a)
```

? В чём разница?

```
c = 14  
print(c)
```

14

```
c = 14  
print("c")
```

c

# Работа с переменными

## Изменение значения

```
i = i + 1
```

увеличить на 1

$$i \leftarrow i + 1$$

```
a = 4
b = 7
a = a + 1
b = b + 1
a = a + b
b = b + a
a = a + 2
b = b + a
```

a	b
4	
	7
5	
	8
13	
	21
15	
	36

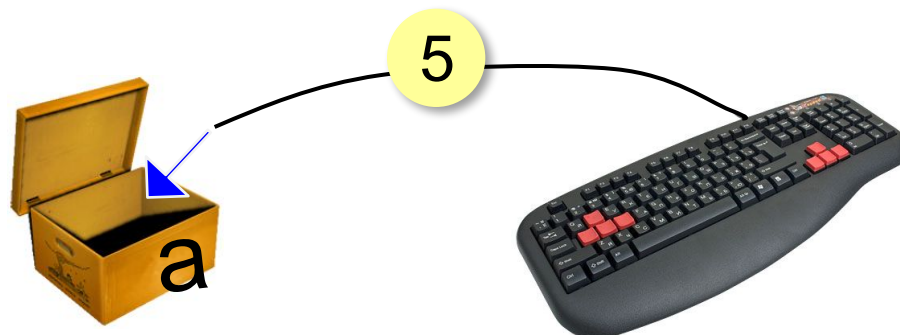
Python:

```
a, b = 4, 7
a += 1
b += 1
a += b
b += a
a += 2
b += a
```

# Ввод с клавиатуры

Цель – изменить исходные данные, не меняя программу.

```
a = input()
```



1. Программа ждет, пока пользователь введет значение и нажмет *Enter*.
2. Введенное значение записывается в переменную **a**.

# Ввод с клавиатуры

```
a = input ()
```

ввести строку с клавиатуры  
и связать с переменной a

```
b = input ()
```

```
c = a + b
```

```
print ( c )
```

Протокол:

21

33

2133



Почему?



Результат функции `input` – строка символов!

преобразовать в  
целое число

```
a = int ( input () )
```

```
b = int ( input () )
```



## Ввод с подсказкой

Введите число: 26

```
a = input("Введите число: ")
```



Что не так?

подсказка

```
a = int(input("Введите число: "))
```



Что будет?

преобразовать  
в целое число

Введите число: Qu-Qu

**ValueError: invalid literal for int() with base 10: 'Qu-Qu'**

# Ввод вещественных чисел

---

```
print( "Введите число:" )  
x = float( input() )
```

или так:

```
x = float( input("Введите число:") )
```

# Программа сложения чисел

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



Что плохо?

ожидание:

Введите два числа:

5

7

5+7=12

реальность:

5

7

12



Как улучшить диалог?

# Вывод данных с текстом

значение *a*

значение *b*

значение *c*

5+7=12

ТЕКСТ

```
print(a, "+", b, "=", c)
```

ожидание:

5+7=12

реальность:

5 + 7 = 12

это пробелы не заказывали!

```
print(a, "+", b, "=", c, sep=" ")
```

*separator*

пустой

# Вывод данных с текстом (f-строки)

значение  $a$

значение  $b$

значение  $c$

$5+7=12$

текст

имена переменных в  
фигурных скобках

```
print ( f "{a}+{b}={c}" )
```

форматная  
строка

# Программа сложения чисел

---

```
print ( "Введите два числа: " )  
a = int ( input() )  
b = int ( input() )  
c = a + b  
print ( f "{a}+{b}={c}" )
```



Как переделать для 3-х чисел?

# Ввод двух чисел в одной строке

```
a, b = map ( int, input() .split() )
```

21 33

`input()`

ввести строку с клавиатуры

21 33

`input().split()`

целые

применить

разделить строку на части по пробелам

21 33

`map ( int, input().split() )`

эту операцию

к каждой части

```
a, b = map ( int, input() .split() )
```

# Арифметические выражения

---

$$a \leftarrow \frac{c + b - 1}{2} \cdot d$$

Линейная запись (в одну строку):

```
a = (c + b - 1) / 2 * d
```

Операции: + -

\* – умножение

/ – деление

\*\* – возведение в степень ( $x^2 \rightarrow \mathbf{x**2}$ )



# Порядок выполнения операций

3      1    2      4          5    6

```
a = (c + b**5*3 - 1) / 2 * d
```

**Приоритет** (старшинство):

- 1) скобки
- 2) возведение в степень \*\*
- 3) умножение и деление
- 4) сложение и вычитание

```
a = (c + b**5*3 - 1) \
      / 2 * d
```

```
a = (c + b**5*3
      - 1) / 2 * d
```

$$a = \frac{c + b^5 \cdot 3 - 1}{2} \cdot d$$

перенос на  
следующую строку

перенос внутри  
скобок разрешён

# Деление

---

Классическое деление:

```
a = 9; b = 6
x = 3 / 4      # = 0.75
x = a / b      # = 1.5
x = -3 / 4     # = -0.75
x = -a / b     # = -1.5
```

Целочисленное деление (округление «вниз»!):

```
a = 9; b = 6
x = 3 // 4     # = 0
x = a // b     # = 1
x = -3 // 4    # = -1
x = -a // b    # = -2
```

# Частное и остаток

---

// – деление нацело (остаток отбрасывается)

% – остаток от деления

175 сек = 2 мин 55 сек



Как получить 2 и 55?


```
t = 175
```

```
m = t // 60
```

```
s = t % 60
```

# Частное и остаток

---

 Что получится?

```
n = 123
d = n // 10
k = n % 10
```

При делении на 10 нацело отбрасывается последняя цифра числа.

Остаток от деления на 10 – это последняя цифра числа.

# Операторы // и %

```
a = 1234
d = a % 10; print( d )
a = a // 10
d = a % 10; print( d )
a = a // 10
d = a % 10; print( d )
a = a // 10
d = a % 10; print( d )
a = a // 10 :
```

4

3

2

1

# Сокращенная запись операций

```
a += b # a = a + b
a -= b # a = a - b
a *= b # a = a * b
a /= b # a = a / b
a //= b # a = a // b
a %= b # a = a % b
```

```
a += 1
```

увеличение на 1

# ФОРМАТНЫЙ ВЫВОД

---

```
a, b = 1, 2  
print( f"{a}+{b}={a+b}" )
```

→ 1+2=3

# ФОРМАТНЫЙ ВЫВОД

```
a = 1; b = 2; c = 3
print( a, b, c )
```

1 2 3

форматная строка

```
print( f"{a}{b}{c}" )
```

123

```
print( f"{a}{b:3}{c:5}" )
```

КОЛИЧЕСТВО ЗНАКОВ  
НА ВЫВОД ЧИСЛА

1 2 3  
3 5



Сколько знаков для вывода *a*?



# ФОРМАТНЫЙ ВЫВОД

```
x=12.345678
print( f"x={x}" )
```

x=12.345678

всего на  
число

в дробной  
части

```
print( f"x={x:10.3f}" )
```

x=\_\_\_\_\_12.346

3

10

```
print( f"x={x:8.2f}" )
```

x=\_\_\_\_\_12.34

# ФОРМАТНЫЙ ВЫВОД

---

```
print ( f"x={x:2.2f}" )
```

→ x=12.34

```
print ( f"x={x:.2f}" )
```

→ x=12.34

МИНИМАЛЬНО  
ВОЗМОЖНОЕ

```
print ( f"x={x:0.1f}" )
```

→ x=12.3

# Научный формат чисел

---

```
x=123456789  
print( f"x={x:e}" )
```

→ `x=1.234568e+008`  
 $1,234568 \cdot 10^8$

```
x=0.0000123456789  
print( f"x={x:e}" )
```

→ `x=1.234568e-005`  
 $1,234568 \cdot 10^{-5}$

# Операции с вещественными числами

---

**int** – целая часть числа

```
x=1.6  
print( int(x) )
```



1

**round** – ближайшее целое число

```
x=-1.2  
print( round(x) )
```



-1

# Математические функции

загрузить  
модуль `math`

= подключить математические  
функции

```
import math
# квадратный корень
print( math.sqrt(25) )
r = 50 # радиус окружности
print( 2*math.pi*r )
print( math.pi*r**2 )
```



Что считаем?

число  $\pi$

округление  
вверх

```
print( math.ceil(2.123) ) # 3
```

# Операции с вещественными числами

$$1/3 = 0,33333\dots$$

бесконечно много знаков



Большинство вещественных чисел хранятся в памяти компьютера с ошибкой!

```
x = 1/2
y = 1/3
z = 5/6 # 5/6=1/2+1/3
print(x+y-z)
```



-1.110223e-016

# Случайные и псевдослучайные числа

## Случайные явления

- встретил слона – не встретил слона
- жеребьёвка на соревнованиях
- лотерея
- случайная скорость (направление выстрела ) в игре
- ...



**Случайные числа** — это последовательность чисел, в которой невозможно предсказать следующее число, даже зная все предыдущие.

# Случайные и псевдослучайные числа

**!** Компьютер неслучаен!

**Псевдослучайные числа** — похожи на случайные, но строятся по формуле.

следующее

предыдущее

$$X_{n+1} = (a * X_n + b) \% c \# \text{от } 0 \text{ до } c-1$$

$$X_{n+1} = (X_n + 3) \% 10 \# \text{от } 0 \text{ до } 9$$

$X = 0 \rightarrow 3 \rightarrow 6 \rightarrow 9 \rightarrow 2 \rightarrow 5 \rightarrow 8$

$8 \rightarrow 1 \rightarrow 4 \rightarrow 7 \rightarrow 0$

зерно

зацикливание



# Датчик случайных чисел

## Целые числа на отрезке:

ПОДКЛЮЧИТЬ функцию `randint`  
ИЗ модуля `random`

```
from random import randint
K = randint(1, 6) # отрезок [1, 6]
L = randint(1, 6) # это уже другое число!
```

англ. *integer* – целый  
*random* – случайный



Не нужно имя модуля!

```
K = random.randint(1, 6)
```

# Датчик случайных чисел

---

## Вещественные числа:

```
from random import random, uniform
x = random()           # полуинтервал [0,1)
y = 7*random()
z = 7*random()+5
```

## Вещественные числа на отрезке [a, b]:

```
from random import uniform
x = uniform(1.5, 2.8)  # [1,5; 2,8]
y = uniform(5.25, 12.75) # [5,25; 12,75]
```

# Задачи

---

- «А»: В игре «Русское лото» из мешка случайным образом выбираются бочонки, на каждом из которых написано число от 1 до 90. Напишите программу, которая выводит наугад первые 5 выигрышных номеров.
- «В»: + Доработайте программу «Русское лото» так, чтобы все 5 значений гарантированно были бы разными (используйте разные диапазоны).

# Задачи

---

**«С»:** + Игральный кубик бросается три раза (выпадает три случайных значения). Из этих чисел составляется целое число, программа должна найти его квадрат.

## Пример:

Выпало очков :

1 2 3

Число 123

Его квадрат 15129

# Задачи

---

«D»: + Получить случайное трёхзначное число и вывести в столбик его отдельные цифры.

**Пример:**

Получено число 123

сотни: 1

десятки: 2

единицы: 3