

Программирование на языке Python

§ 59. Процедуры

Зачем нужны процедуры?

```
print ( "Ошибка программы" )
```

много раз!

Процедура:

define
определит
ь

```
def Error () :  
    print ( "Ошибка программы" )
```

```
n = int ( input () )  
if n < 0:  
    Error ()
```

ВЫЗОВ
процедуры

Что такое процедура?

Процедура – вспомогательный алгоритм, который выполняет некоторые действия.

- текст (расшифровка) процедуры записывается **до** её вызова в основной программе
- в программе может быть **много процедур**
- чтобы процедура заработала, нужно **вызвать** её по имени из основной программы или из другой процедуры

Процедура с параметрами

Задача. Вывести на экран запись целого числа (0..255) в 8-битном двоичном коде.

много раз!

Алгоритм:

178 \Rightarrow 10110010₂



Как вывести первую цифру?

n := 1 0 1 1 0 0 1 0₂ разряды
 7 6 5 4 3 2 1 0

n // 128

n % 128



Как вывести вторую цифру?

n1 // 64

Процедура с параметрами

Задача. Вывести на экран запись целого числа (0..255) в 8-битном двоичном коде.

Решение:

```
k = 128
while k > 0:
    print ( n // k,
           end = " " )
    n = n % k
    k = k // 2
```

178 ⇒ 10110010

n	k	ВЫВОД
178	128	1



**Результат
зависит
от n!**

Процедура с параметрами

Параметры – данные, изменяющие работу процедуры.

локальная
переменная

```
def printBin( n ) :  
    k = 128  
    while k > 0 :  
        print ( n // k, end = "" )  
        n = n % k ;  
        k = k // 2
```

```
printBin ( 99 )
```

значение параметра
(аргумент)

Несколько параметров:

```
def printSred( a, b ) :  
    print ( (a + b) / 2 )
```

Локальные и глобальные переменные

глобальная
переменная

локальная
переменная

```
a = 5
def qq():
    a = 1
    print ( a )
qq()
print ( a )
```

1

5

```
a = 5
def qq():
    print ( a )
qq()
```

5

```
a = 5
def qq():
    global a
    a = 1
qq()
print ( a )
```

работаем с
глобальной
переменной

1

Неправильная процедура

```
x = 5; y = 10
```

```
xSum ()
```



Что плохо?

```
def xSum():  
    print(x+y)
```



- 1) процедура связана с глобальными переменными, нельзя перенести в другую программу
- 2) печатает только сумму x и y , нельзя напечатать сумму других переменных или сумму $x*y$ и $3x$



Как исправить?

передавать
данные через
параметры

Правильная процедура

Глобальные:

x	y
5	10
z	w
17	3

```

x = 5 ; y = 10
Sum2 ( x , y )
z=17 ; w=3
Sum2 ( z , w )
Sum2 ( z+x , y*w )
  
```

```

def Sum2 ( a , b ) :
    print ( a+b )
  
```

Локальные:

a	b	
5	10	15
		20
		52



- 1) процедура не зависит от глобальных переменных
- 2) легко перенести в другую программу
- 3) печатает только сумму любых выражений

Задачи

«А»: Напишите процедуру, которая принимает параметр – натуральное число N – и выводит на экран линию из N символов '-'.

Пример:

Введите N :

10

«В»: Напишите процедуру, которая выводит на экран в столбик все цифры переданного ей числа, начиная с первой.

Пример:

Введите натуральное число:

1234

1

2

3

4

Задачи

«С»: Напишите процедуру, которая выводит на экран запись переданного ей числа в римской системе счисления.

Пример:

Введите натуральное число :

2013

MMXIII

Программирование на языке Python

§ 60. Функции

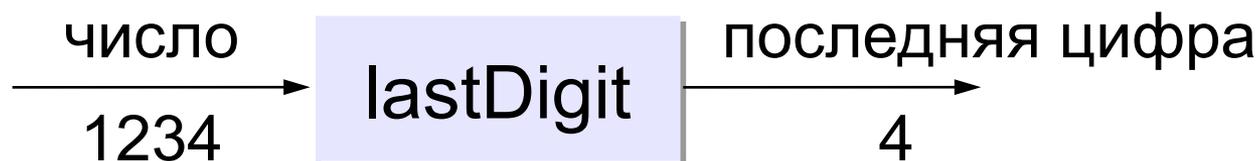
Что такое функция?

Функция – это вспомогательный алгоритм, который возвращает *значение-результат* (число, символ или объект другого типа).

```
s = input()  
n = int( s )  
x = randint( 10, 20 )
```

Что такое функция?

Задача. Написать функцию, которая вычисляет младшую цифру числа (разряд единиц).



```
def lastDigit( n ):  
    d = n % 10  
    return d
```

результат работы функции – значение **d**

передача результата

```
# вызов функции  
k = lastDigit( 1234 )  
print( k )
```

Сумма цифр числа

Задача. Написать функцию, которая вычисляет сумму цифр числа.

```
def sumDigits ( n ) :  
    sum = 0  
    while n != 0 :  
        sum += n % 10  
        n = n // 10  
    return sum
```

передача
результата

```
# основная программа  
sumDigits (12345)
```

```
# сохранить в переменной  
n = sumDigits (12345)
```

```
# сразу вывод на экран  
print ( sumDigits (12345) )
```



Что плохо?

Использование функций

```
x = 2*sumDigits ( n+5 )
z = sumDigits ( k ) + sumDigits ( m )
if sumDigits ( n ) % 2 == 0:
    print ( "Сумма цифр чётная" )
    print ( "Она равна", sumDigits ( n ) )
```



Функция, возвращающая целое число, может использоваться везде, где и целая величина!

Одна функция вызывает другую:

```
def middle ( a, b, c ):
    mi = min ( a, b, c )
    ma = max ( a, b, c )
    return a + b + c - mi - ma
```

ВЫЗЫВАЮТСЯ
min И max



Что вычисляет?

Задачи

«А»: Напишите функцию, которая находит наибольший общий делитель двух натуральных чисел.

Пример:

Введите два натуральных числа:

7006652 112307574

$\text{НОД}(7006652, 112307574) = 1234.$

«В»: Напишите функцию, которая определяет сумму цифр переданного ей числа.

Пример:

Введите натуральное число:

123

Сумма цифр числа 123 равна 6.

Задачи

«С»: Напишите функцию, которая «переворачивает» число, то есть возвращает число, в котором цифры стоят в обратном порядке.

Пример:

Введите натуральное число:

1234

После переворота: 4321.

Как вернуть несколько значений?

```
def divmod ( x, y ) :  
    d = x // y  
    m = x % y  
    return d, m
```

d – частное,
m – остаток

```
a, b = divmod ( 7, 3 )  
print ( a, b )           # 2 1
```

кортеж – набор
элементов

```
q = divmod ( 7, 3 )  
print ( q )           # (2, (2, 1))
```

q[0]

q[1]

Задачи

«А»: Напишите функцию, которая переставляет три переданные ей числа в порядке возрастания.

Пример:

Введите три натуральных числа:

10 15 5

5 10 15

«В»: Напишите функцию, которая сокращает дробь вида M/N .

Пример:

Введите числитель и знаменатель дроби:

25 15

После сокращения: $5/3$

Задачи

«С»: Напишите функцию, которая вычисляет наибольший общий делитель и наименьшее общее кратное двух натуральных чисел.

Пример:

Введите два натуральных числа:

10 15

НОД (10 , 15) = 5

НОК (10 , 15) = 30

Логические функции

Логическая функция – это функция, возвращающая логическое значение (True/False).

```
def even(n):  
    if n % 2 == 0:  
        return True  
    else:  
        return False
```



```
def even(n):  
    return (n % 2 == 0)
```

```
k = int( input() )  
if even( k ):  
    print( "Число", k, "чётное." )  
else:  
    print( "Число", k, "нечётное." )
```

Логические функции

Задача. Найти все простые числа в диапазоне от 2 до 1000.

```
for i in range(2, 1001):  
    if isPrime(i):  
        print(i)
```

функция,
возвращающая
логическое значение
(True/False)

Функция: простое число или нет?

? Какой алгоритм?

```
def isPrime ( n ) :  
    k = 2  
    while k*k <= n and n % k != 0 :  
        k += 1  
    return (k*k > n)
```

```
if k*k > n :  
    return True  
else :  
    return False
```

Логические функции: использование



Функция, возвращающая логическое значение, может использоваться везде, где и логическая величина!

```
n = int ( input () )  
while isPrime (n) :  
    print ( n, "- простое число" )  
    n = int ( input () )
```

Задачи

«А»: Напишите логическую функцию, которая определяет, является ли переданное ей число совершенным, то есть, равно ли оно сумме своих делителей, меньших его самого.

Пример:

Введите натуральное число:

28

Число 28 совершенное.

Пример:

Введите натуральное число:

29

Число 29 не совершенное.

Задачи

«В»: Напишите логическую функцию, которая определяет, являются ли два переданные ей числа взаимно простыми, то есть, не имеющими общих делителей, кроме 1.

Пример:

Введите два натуральных числа:

28 15

Числа 28 и 15 взаимно простые.

Пример:

Введите два натуральных числа:

28 16

Числа 28 и 16 не взаимно простые.

Задачи

«С»: Простое число называется гиперпростым, если любое число, получающееся из него откидыванием нескольких цифр, тоже является простым. Например, число 733 – гиперпростое, так как и оно само, и числа 73 и 7 – простые. Напишите логическую функцию, которая определяет, верно ли, что переданное ей число – гиперпростое. Используйте уже готовую функцию `isPrime`, которая приведена в учебнике.

Пример:

Введите натуральное число:

733

Число 733 гиперпростое.

Пример:

Введите натуральное число:

19

Число 19 не гиперпростое.