Стандартные функции Python

АИШ, «Структурное программирование на Python»

- **Функция** – кусочек кода, который можно вызвать с различными параметрами.

- Функции помогают структурировать код и уменьшить его количество.

Поиск максимума и минимума трех чисел без использования встроенных функций

```
a,b,c = int(input()), int(input()), int(input())
if a >= b and a >= c:
    print("Максимум:", a)
elif b >= a and b >= c:
    print("Максимум:", b)
else:
    print("Максимум:", c)
if a <= b and a <= c:
    print("Минимум:", a)
elif b <= a and b <= c:
    print("Минимум:", b)
else:
    print("Минимум:", c)
```

Поиск максимума и минимума трех чисел с использованием встроенных функций

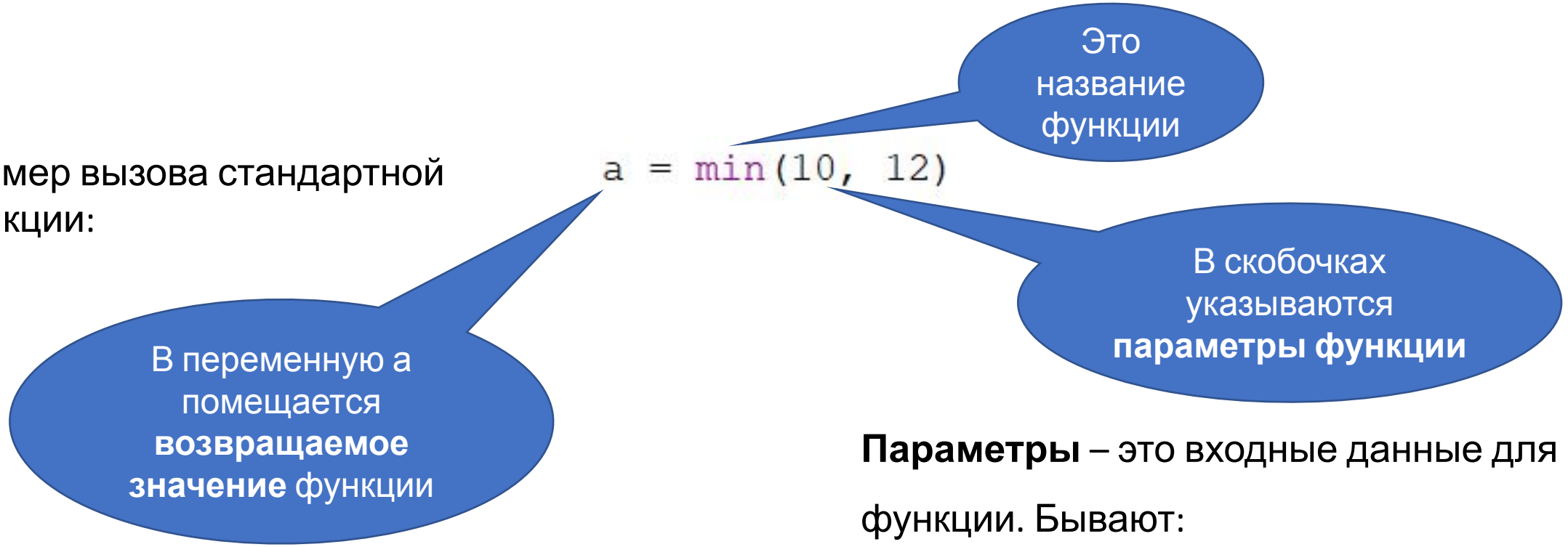
```
a,b,c = int(input()), int(input()), int(input())
print("Максимум:", max(a,b,c))
print("Минимум:", min(a,b,c))
```

Использование встроенных функций **min** и **max** позволило сократить количество кода и упростить его понимание

Откуда берутся функции?

- Есть **стандартные (встроенные)** функции python, которые можно вызвать в любом месте программы (например, print).
- Есть **подключаемые** функции, для их использования придется сначала подключить модуль, а потом обращаться к функции в формате имя_модуля.имя_функции.
- А еще вы можете самостоятельно написать функцию. Такие функции называются **пользовательскими**.

Пример вызова стандартной функции:



Не все функции явно возвращают какое-либо значение.

Например, **min(a, b)** возвратит наименьшее из двух чисел a и b. А функция **print(a, b)** напечатает оба числа в консоль, но ничего не вернет.

Параметры – это входные данные для функции. Бывают:

- Обязательные – их при вызове функции обязательно указывать.
- Необязательные – их указывать можно, но не обязательно.

Вы уже с ними знакомы…

Вот так в квадратных скобочках указываются необязательные параметры

- input([подсказка пользователю]) – возвращает введенные пользователем данные;
- int(x) – преобразует переменную x к целочисленному типу данных, если число было вещественного типа, то оно округлится в сторону нуля;
- float(x) - преобразует переменную x к типу данных с плавающей точкой;
- str(x) – преобразует переменную x к строковому типу данных;
- print(x) – печатает содержимое переменной x;
- range([начало,] конец, [, шаг]) – возвращает набор чисел от начала до конца (не включая) с заданным шагом (по умолчанию 1).

Пример вызова подключаемой функци

```
import random
random_number = random.randint(1,100)
print(random_number)
```

Подключаем модуль **random**, чтобы использовать функцию **randint**

В переменную **random_number** функция **randint** вернет случайное число от 1 до 100

Обращаемся к функции так: **имя_модуля. имя_функции**. В скобочках указываем параметры

Модуль – это любой файл с программой на python. Если лень писать название модуля целиком, можно воспользоваться пс

```
import random as rnd
random_number = rnd.randint(1,100)
```

Подключаем модуль **random** и даем ему псевдоним **rnd**

Теперь к функции можно обратиться так: **псевдоним. имя_функции**

Если какие-то подключаемые функции используются в программе часто:

```
from math import ceil, floor
a = 1.75
b = -3.17
print(ceil(a))
print(floor(a))
print(ceil(b))
print(floor(b))
```

Подгружаем только нужные функции

И теперь обращаемся к этим функциям просто по имени, без названия модуля или псевдонимов

Кстати, а что выведет код этой программы?

```
2
1
-3
-4
```

Полезности

- abs(x) – возвращает модуль числа x;
- divmod (x, y) – возвращает частное и остаток от деления x на y;
- len(x) – возвращает число элементов в объекте x;
- min(a, b [, c], ..) – возвращает минимальное из нескольких чисел;
- max(a, b [, c], ..) – возвращает максимальное из нескольких чисел;
- round(x [, n]) – округляет число x до n знаков после запятой

Некоторые подключаемые функции из модуля **math**:

- ceil(x) – округление вверх;
- floor(x) – округление вниз;
- pow(x, y) – возвращает число x в степени y, аналог операции x**y;
- sqrt(x) – квадратный корень из x;
- trunc(x) – усекает значение x до целого.

```
import math
a = 2.43
print(math.ceil(a))
print(math.floor(a))
```

Программа напечатает сначала 3, потом 2

```
import math
b = -2.43
print(math.ceil(b))
print(math.floor(b))
```

Программа напечатает сначала -2, потом -3

Некоторые подключаемые функции из модуля **random**:

- randrange(начало, конец, шаг) – возвращает случайное число из последовательности с заданным шагом;
- randint(x, y) – возвращает случайное целое число в промежутке от x до y включительно;
- random() – возвращает случайное число от 0 до 1;
- uniform(x, y) – случайное число с плавающей точкой от x до y включительно.

Инициализация начального состояния генератора

Случайные числа в диапазоне от 0 до 100 с шагом 10

```
import random as rnd
rnd.seed()
for k in range(10):
    n = rnd.randrange(0,100,10)
    print(n)
```

Результаты нескольких запусков программы

```
40
70
80
50
70
50
50
20
40
10
```

```
80
60
90
0
40
60
50
30
20
0
```

```
30
10
0
10
50
30
20
0
60
30
```