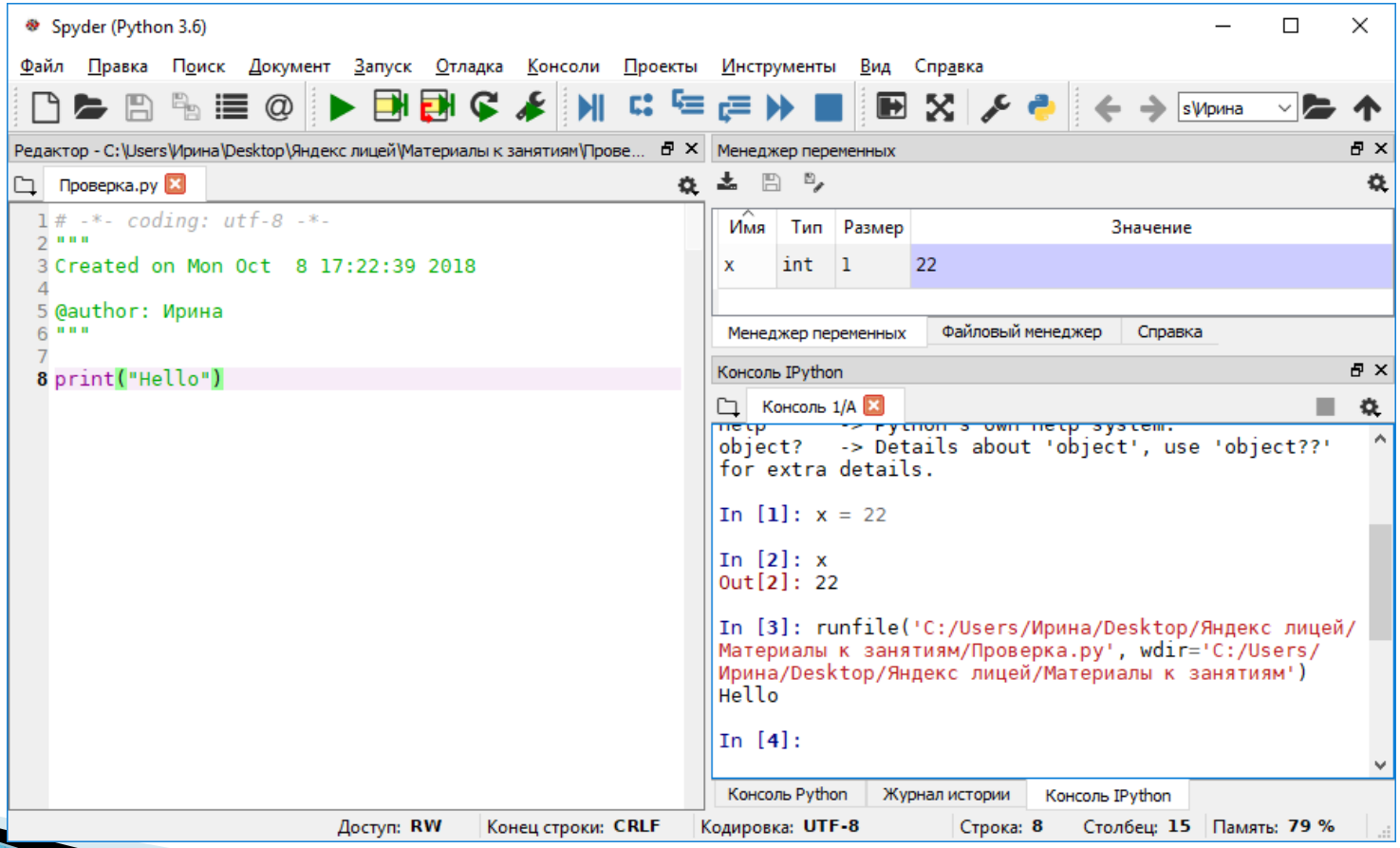
Алгоритмы и структуры данных в языке Python

Лектор: доцент Департамента анализа данных, принятия решений и финансовых технологий
Миронова Ирина Васильевна

Среда программирования

https://www.anaconda.com/distribution/



Понятие объекта

Данные в языке Python представлены в форме объектов.

Объект – это область памяти со значением. С каждым объектом связан строго определенный набор операций.

Кроме значения каждый объект имеет 2 стандартных поля: описатель типа и счетчик ссылок.

Тип объекта может быть встроенным в язык. Можно создавать новые типы средствами языка или использовать типы из библиотек.

Для доступа к объекту используется ***переменная***, в которой сохраняется ссылка на объект.

Типы данных

int – целые числа;
float – вещественные числа;
str – Unicode-строки;
bool – логический тип;
list – список;
tuple – кортеж;
dict – словарь;
set – множество.

Узнать тип значения можно с помощью функции type().

```
In [1]: type(25)
Out[1]: int

In [2]: type(12.345)
Out[2]: float

In [3]: type('строка')
Out[3]: str

In [4]: type(True)
Out[4]: bool

In [5]: type([1,2,3,4])
Out[5]: list

In [6]: type({'a':1, 'b':2})
Out[6]: dict

In [7]: type((1,2,3,4))
Out[7]: tuple
```

Динамическая типизация

В языке отсутствуют инструкции объявления переменных.

Переменная создается, когда в программе ей впервые присваивается значение. Значением переменной является ссылка. Повторные операции просто изменяют это значение.

Переменные не имеют информации о типе. Тип имеют объекты.

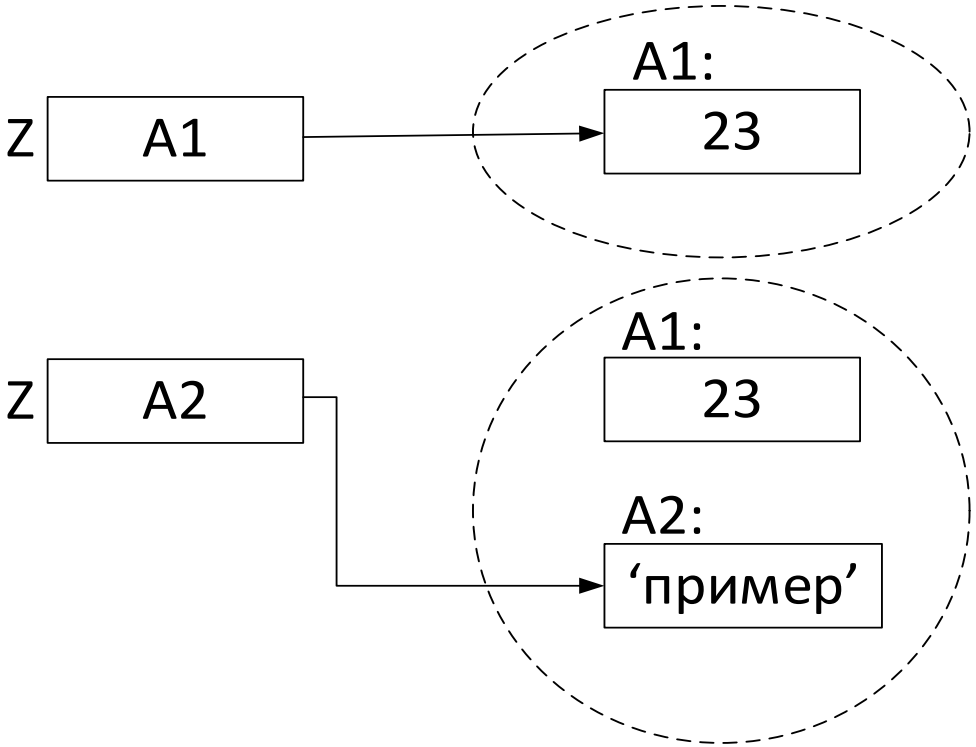
Когда переменная участвует в выражении ее имя заменяется на объект, на который она ссылается, независимо от того, что это за объект.

Чтобы переменную можно было использовать, ей нужно присвоить значение.

При присваивании переменной нового значения, счетчик ссылок у старого значения уменьшается на 1. Если счетчик ссылок становится равным 0, то объект уничтожается.

Пример использования переменной

```
>>> Z=23
>>> Z
23
>>> Z='пример'
>>> Z
'пример'
>>>
```



Имена переменных

Для доступа к данным используется переменная.

Каждая переменная имеет имя.

Имя переменной (идентификатор) состоит из букв, цифр символов подчеркивания. Имя не может начинаться с цифры.

В качестве имен нельзя использовать ключевые слова языка.

Следует избегать совпадения имен с уже имеющимися в системе, например, с именами функций. Так как это может привести к ошибкам.

В редакторе IDLE цвет переменной должен быть черным.

В именах нужно учитывать регистр букв.

Правильные имена: x1, X1, MyData, my_data.

Неправильные имена: 2y, x*, class.

Плохие имена: str, МояПеременная, _x.

Оператор присваивания

Инструкция присваивания создает ссылку на объект.

```
>>> A=25
>>> B=C=0
>>> A, B, C
(25, 0, 0)
>>> A*=2
>>> B+=A
>>> A, B, C
(50, 50, 0)
```

Операции над числами

- Возведение в степень (**)
- Унарный минус (–) и плюс (+)
- Деление (/, //), умножение (*), остаток от деления (%)
- Сложение (+) и вычитание (–)

```
>>> 2**3, 2**-3, 3.0**-2, 3.0**-1.5
(8, 0.125, 0.1111111111111111, 0.1924500897298752 6)
>>> 2**100
1267650600228229401496703205376
>>> 5/2, 5.0/2, 5/-2
(2.5, 2.5, -2.5)
>>> 5//2, 5.0//2, -5//2
(2, 2.0, -3)
>>> 5%2, 5.0%2, -5.0%2
(1, 1.0, 1.0)
```

Примеры использования функций

```
>>> import math
>>> math.pi, math.e
(3.141592653589793, 2.718281828459045)
>>> math.degrees(math.pi), math.radians(180)
(180.0, 3.141592653589793)
>>> math.sqrt(2)
1.4142135623730951
>>> math.ceil(-3.25), math.ceil(3.25), math.ceil(3.75)
(-3, 4, 4)
>>> math.floor(-3.25), math.floor(3.25)
(-4, 3)
>>> math.factorial(5)
120
```

Функция print и форматирование

```
import math
s = input("x = ")
x = float(s)
print(s, x)
#x = float(input("x = "))
print("sin(%.2f) = %.2f" % (x, math.sin(x)) )
print("sin({0:.2f}) = {1:.2f}".format(x, math.sin(x)))
```

```
x = 456
456 456.0
sin(456.00) = -0.45
sin(456.00) = -0.45
>>>
```

Логическое выражение

- Любое число, не равное 0, или непустой объект интерпретируются как истина;
- Операции сравнения <, >, ==, !=, <=, >=, is, is not возвращают значения True и False:

  A==2 B!=A C is A

  2<X<=10 x<y<z<w
- Логические операторы and, or, not . Операторы and и or возвращают истинный или ложный объект, а не просто True и False:

  A<2 and A>-2 0 or 10 v == "OK" or v == "ok"

Условный оператор (пример 1)

```
#решаем уравнение A*x+B=0
A = float(input("A = "))
B = float(input("B = "))
if A == 0:
    if B == 0:
        print("x - любое число")
    else:
        print("нет решения")
else:
    print("x = {0:.2f}".format(-B/A))
```

Условный оператор (пример 2)

```
#определяем время года по номеру месяца
m = int(input("Введите номер месяца: "))
if m > 2 and m < 6:
    print("весна")
elif 6 <= m <= 8:
    print("лето")
elif 9 <= m <= 11:
    print("осень")
elif m == 12 or m == 1 or m == 2:
    print("зима")
else:
    print("введите правильно номер месяца")
```

Цикл while

while условие**:**

блок инструкций

[**else**:

блок, выполняемый, если не было break

]

Использование инструкции **break** внутри цикла приводит к немедленному прекращению цикла, при этом не выполняется ветка else. Следующей будет выполняться инструкция, следующая сразу за циклом.

Использование инструкции **continue** внутри цикла приводит к тому, что все оставшиеся инструкции текущей итерации пропускаются, и начинает выполняться следующая итерация цикла (переходим в строку заголовка).

Пример цикла

```
#Проверяем, является ли число простым
x = int(input("Введите число: "))
d = x//2
while d>1:
    if x%d == 0:
        print ("x имеет делители")
        break
    d -= 1 # d = d-1
else:
    print ("x - простое число")
```

Цикл for

for переменная **in** последовательность**:**

блок инструкций

[**else**:

блок, выполняемый, если не было break

]

Последовательность – коллекция (набор) других объектов, в которой определен порядок следования элементов.

Последовательностями являются строки, списки, кортежи и другие объекты.

Переменная цикла ссылается на текущий элемент последовательности.

Примеры циклов

```
for i in "Пример":  #перебор строки
    print(i, end=" ")
print("")

for i in [1, 2, 3]:  #перебор списка
    print(i, end=" ")
print("")

for i in (-1, -2, -3):  #перебор кортежа
    print(i, end=" ")
print("")

for i in 1, 2, 3, 'one', 'two', 'three':
    print(i, end=" ")
print("")
```

```
П р и м е р
1 2 3
-1 -2 -3
1 2 3 one two three
```

Функция range

for i **in** range(n):
тело цикла

Значение i меняется от 0 до n-1. Если значение n равно нулю или отрицательное, то тело цикла не выполнится ни разу.

for i **in** range(*a*, *b*):
тело цикла

Значение i меняется от *a* до *b*-1. Если $b \le a$, то цикл не будет выполнен ни разу.

for i **in** range(*a*, *b, c*):
тело цикла

Переменная i принимает значения *a, a+c, a+2*c* и т.д., пока i<*b* (для *c*>0) или пока i>*b* (для *c*<0).