

# Лекция 6

Язык программирования Python.

Хайрулин Сергей Сергеевич

email: [s.khairulin@g.nsu.ru](mailto:s.khairulin@g.nsu.ru), [s.khayrulin@gmail.com](mailto:s.khayrulin@gmail.com)

Ссылка на [материалы](#)

# План

- Лекции/практические занятия
  - Тест
- Дифференцированный зачет в конце семестра
  - Защита задания

# Литература

## **Начальный уровень**

- Mark Pilgrim. Dive into Python - <http://www.diveintopython.net/>
- Марк Лутц. Изучаем Python, 4-е издание // Символ-Плюс 2011.
- ...

## **Стандарт/Документация**

- PEP-8 - <https://www.python.org/dev/peps/pep-0008/>
- <https://www.python.org/>
- <https://github.com/python/cpython>

## **Экспертный уровень**

- Лучано Рамальо: Python. К вершинам мастерства
- Mitchell L. Model. Bioinformatics Programming Using Python // O'Reilly 2010.

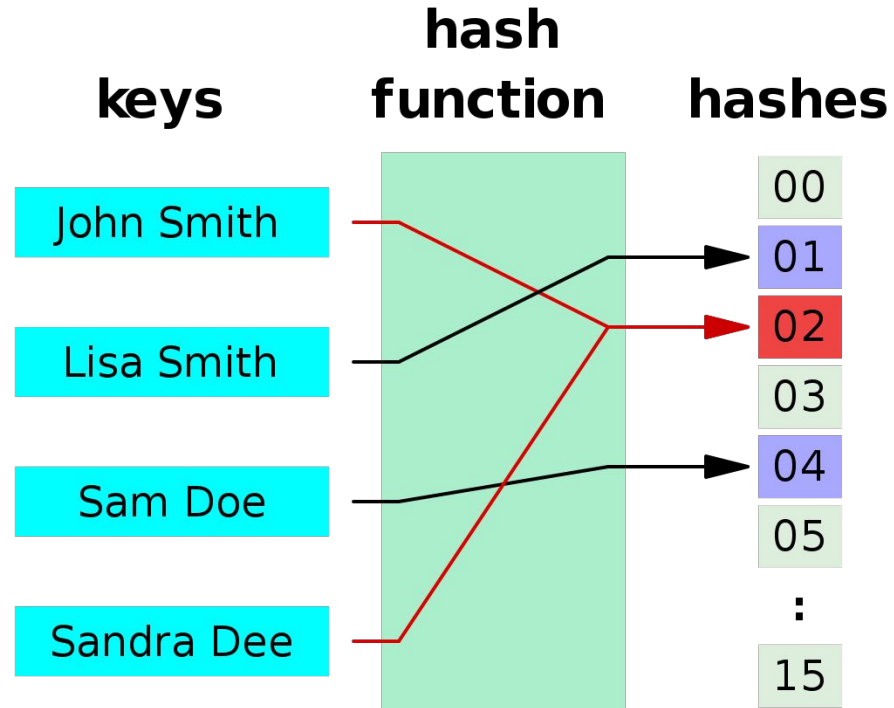
# Версии Python

- Python 2 вышел 2010 году последняя версия 2.7.16 - исправлялись только баги(ошибки) с января 2020 года поддержка прекращена.
- Python 3 появился в 2008, является актуальной версией языка. Текущая стабильная версия 3.8.5 -> в предрелиз 3.9, в разработке 3.10
  - Python 3 не гарантирует совместимости кода с Python 2

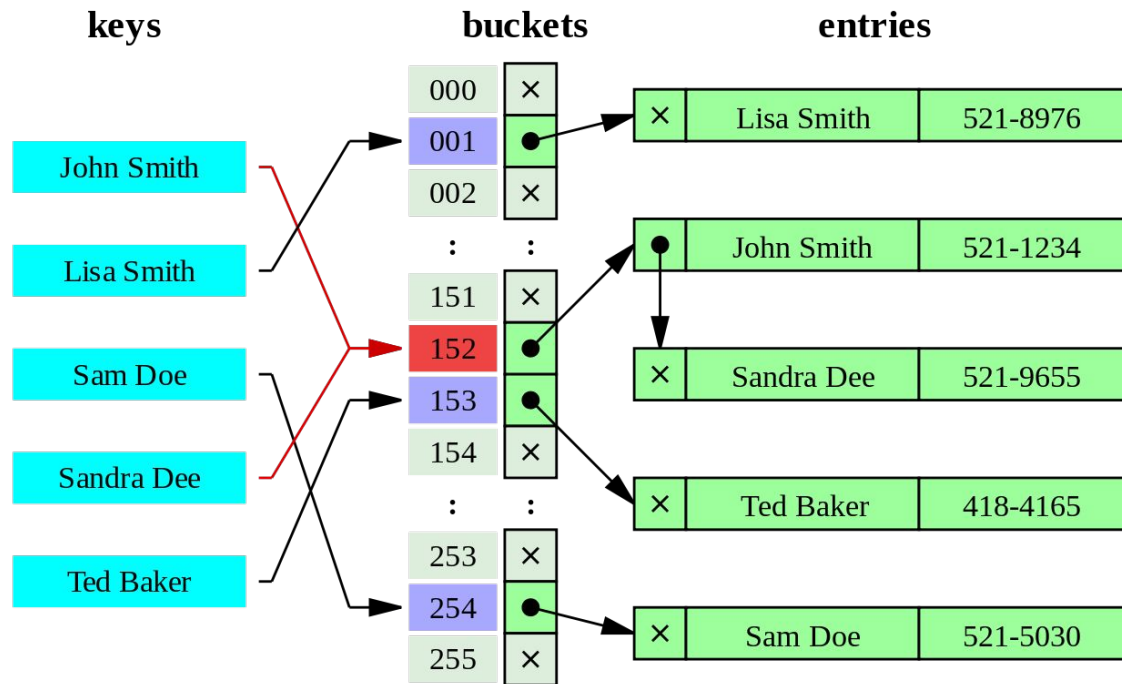
# План занятия

- Общее представление
- Hash function
- Словарь - Dict
  - definition
- Множество - Set
- Практика

# Hash function

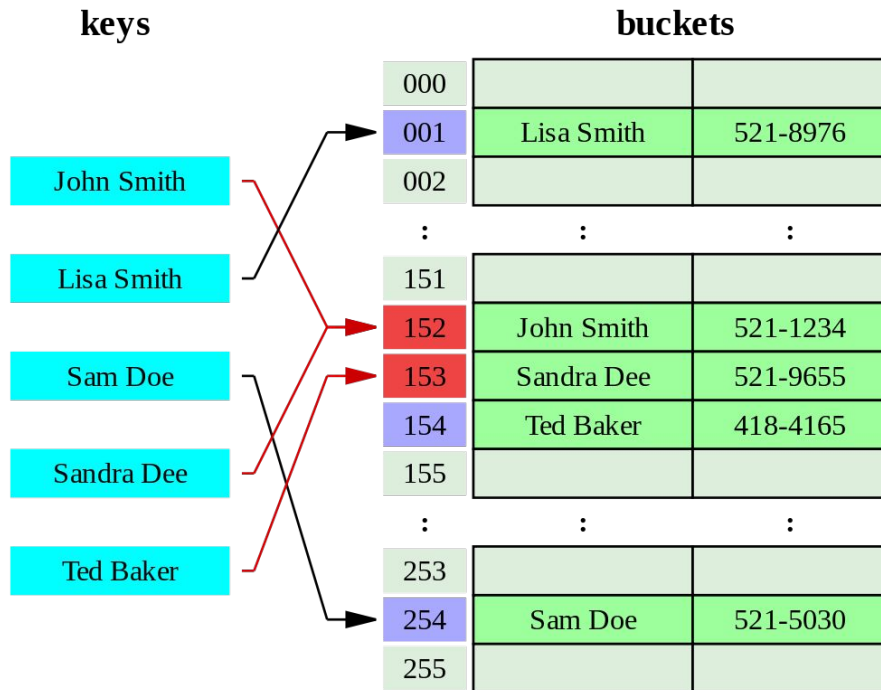


# Hash function (коллизия)





# Hash function (коллизия)



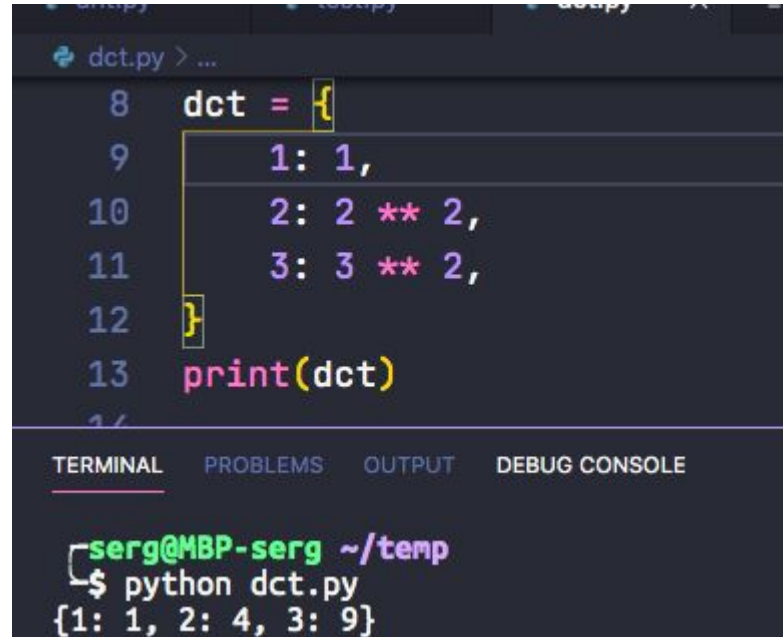
# Hash table (python)



The image shows a code editor window with a dark theme. The top part displays a Python script named `dct.py` with three lines of code: `hash_table = {}`, `hash_table["key1"] = "value"`, and `print(hash_table)`. The bottom part shows a terminal window with the command `python dct.py` being executed, resulting in the output `{'key1': 'value'}`.

```
dct.py > ...  
1 hash_table = {}  
2 hash_table["key1"] = "value"  
3 print(hash_table)  
  
TERMINAL PROBLEMS 2 OUTPUT DEBUG CONSOLE  
serg@MBP-serg ~/temp  
$ python dct.py  
{'key1': 'value'}  
serg@MBP-serg ~/temp  
$
```

# Python **dict** obj definition 1



The image shows a code editor window with a dark theme. The editor displays a Python script named `dct.py` with the following code:

```
8  dct = {  
9      1: 1,  
10     2: 2 ** 2,  
11     3: 3 ** 2,  
12 }  
13 print(dct)
```

Below the code editor, there is a terminal window. The terminal shows the command `python dct.py` being executed, and the output is `{1: 1, 2: 4, 3: 9}`.

```
serg@MBP-serg ~/temp  
$ python dct.py  
{1: 1, 2: 4, 3: 9}
```

# Python **dict** obj definition 2

```
dct.py > ...
14
15  dct = {i: i**2 for i in range(1,4)}
16  print(dct)
17
```

---

TERMINAL   PROBLEMS   OUTPUT   DEBUG CONSOLE

```
serg@MBP-serg ~/temp
$ python dct.py
{1: 1, 2: 4, 3: 9}
```

# Python **dict** obj definition 3

```
21  dct = {}
22  for i in range(1, 4):
23      key = i
24      value = i ** 2
25      dct[key] = value
26  print(dct)
```

TERMINAL

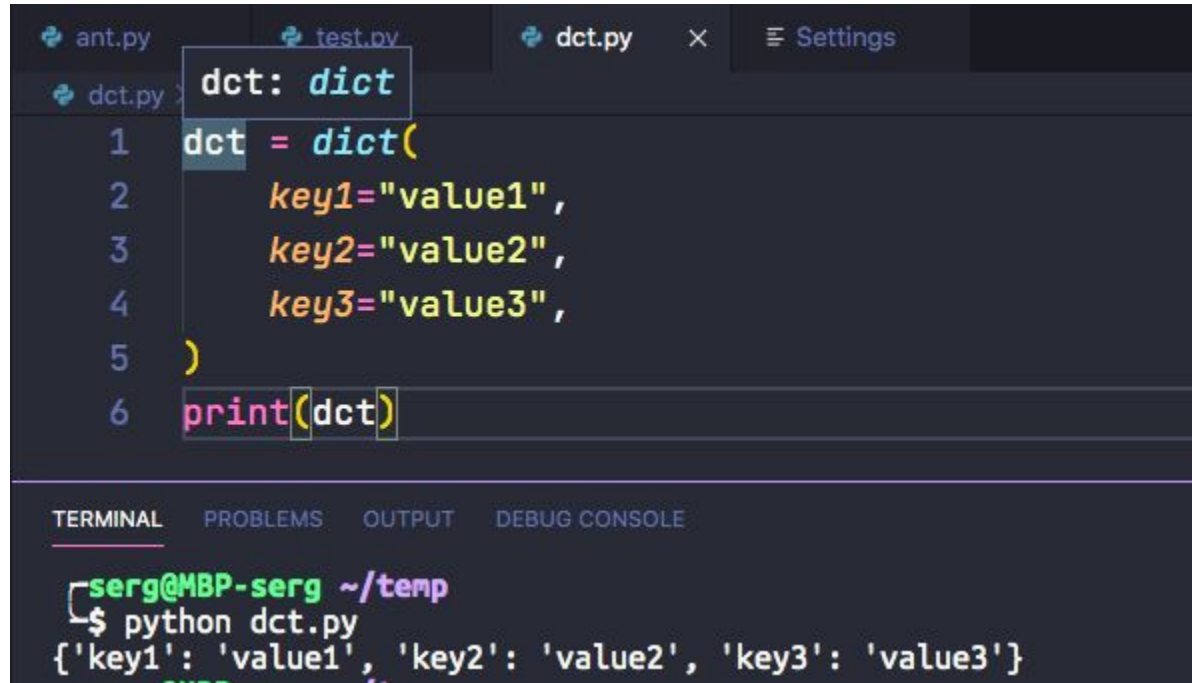
PROBLEMS

OUTPUT

DEBUG CONSOLE

```
serg@MBP-serg ~/temp
$ python dct.py
{1: 1, 2: 4, 3: 9}
```

# Python **dict** obj definition 3



The image shows a code editor with three tabs: `ant.py`, `test.py`, and `dct.py`. The `dct.py` tab is active, displaying the following Python code:

```
dct: dict
1  dct = dict(
2      key1="value1",
3      key2="value2",
4      key3="value3",
5  )
6  print(dct)
```

Below the code editor, the **TERMINAL** panel shows the execution of the script:

```
serg@MBP-serg ~/temp
$ python dct.py
{'key1': 'value1', 'key2': 'value2', 'key3': 'value3'}
```

# Python **dict** obj definition 4

```
>>> a = dict(one=1, two=2, three=3)
>>> b = {'one': 1, 'two': 2, 'three': 3}
>>> c = dict(zip(['one', 'two', 'three'], [1, 2, 3]))
>>> d = dict([('two', 2), ('one', 1), ('three', 3)])
>>> e = dict({'three': 3, 'one': 1, 'two': 2})
>>> f = dict({'one': 1, 'three': 3}, two=2)
>>> a == b == c == d == e == f
True
```

<https://docs.python.org/3/library/stdtypes.html#typesmapping>

# Dict (проход)

```
dct.py > ...
1  dct = {i: i**2 for i in range(1,4)}
2  print("-----")
3  for k in dct:
4      print(k, dct[k])
5  print("-----")
6  for k in dct.keys():
7      print(k, dct[k])
8  print("-----")
9  for v in dct.values():
10     print(v)
11 print("-----")
12 for k, v in dct.items():
13     print(k, v)
```

serg@MBP-serg ~/temp  
\$ python dct.py

```
-----
1 1
2 4
3 9
-----
1 1
2 4
3 9
-----
1
4
9
-----
1 1
2 4
3 9
```

<https://docs.python.org/3/library/stdtypes.html#dict>



# Dict - ключи

- immutable objects
- hashable
- comparable  $\Rightarrow$  obj1 == obj2

# Dict - ключи

Class	Description	Immutable?
<b>bool</b>	Boolean value	✓
<b>int</b>	integer (arbitrary magnitude)	✓
<b>float</b>	floating-point number	✓
<b>list</b>	mutable sequence of objects	
<b>tuple</b>	immutable sequence of objects	✓
<b>str</b>	character string	✓
<b>set</b>	unordered set of distinct objects	
<b>frozenset</b>	immutable form of set class	✓
<b>dict</b>	associative mapping (aka dictionary)	

# Dict - ключи

```
In [2]: d = {[1,2,3]: 1}
```

---

```
TypeError                                Traceback (most recent call last)
<ipython-input-2-7e7847a84a44> in <module>()
----> 1 d = {[1,2,3]: 1}
```

```
TypeError: unhashable type: 'list'
```

```
In [3]: d = {(1,2,3): 1}
```

```
In [4]: d
```

```
Out[4]: {(1, 2, 3): 1}
```

# Множество (Set)

```
In [13]: s1 = set((1,2,3))
```

```
In [14]: s2 = {1,2,3}
```

```
In [15]: s1
```

```
Out[15]: {1, 2, 3}
```

```
In [16]: s2
```

```
Out[16]: {1, 2, 3}
```

```
In [17]: s1 == s2
```

```
Out[17]: True
```

<https://docs.python.org/3/tutorial/datastructures.html#sets>

# Множество (Set)

```
In [31]: 1 in s1
```

```
Out[31]: True
```

```
In [32]: s2.add(4)
```

```
In [33]: s2
```

```
Out[33]: {1, 2, 3, 4}
```

```
In [34]: s1 ^ s2
```

```
Out[34]: {4}
```

```
In [35]: s1 | s2
```

```
Out[35]: {1, 2, 3, 4}
```

```
In [36]: s1 & s2
```

```
Out[36]: {1, 2, 3}
```

# Практическая Часть

1. [Словари](#)