Introduction To Python Programming

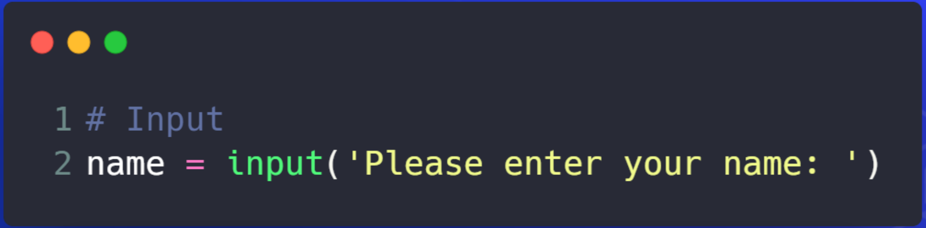
Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
  - Numbers & Math
  - Boolean & Comparison and Logic
  - Strings
  - Lists
  - Tuples
  - Sets
  - Dictionaries
- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

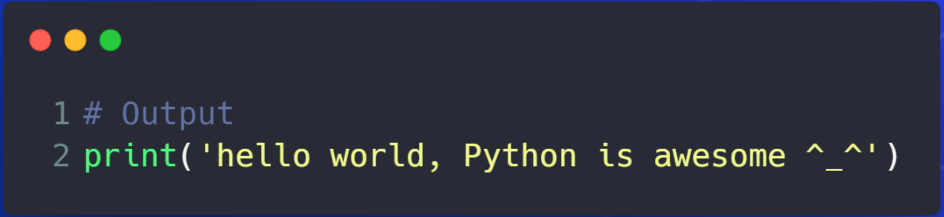
Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- **Input & Output**
- Variables
- Data types
  - Numbers & Math
  - Boolean & Comparison and Logic
  - Strings
  - Lists
  - Tuples
  - Sets
  - Dictionaries
- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, …)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Input & Output



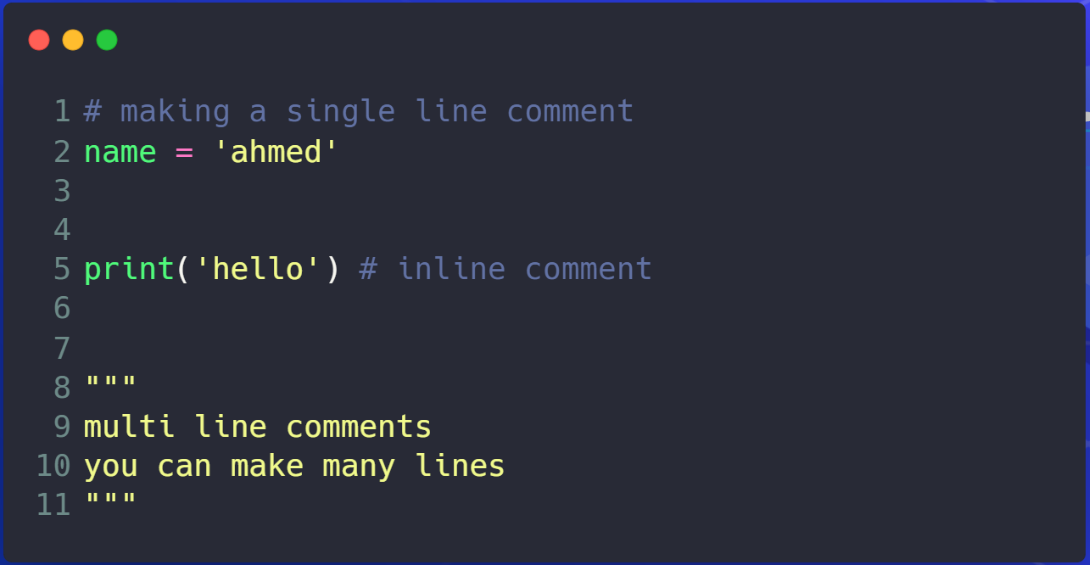
```python
# Input
name = input('Please enter your name: ')
```



```python
# Output
print('hello world, Python is awesome ^_^')
```

Input & Output

Comments

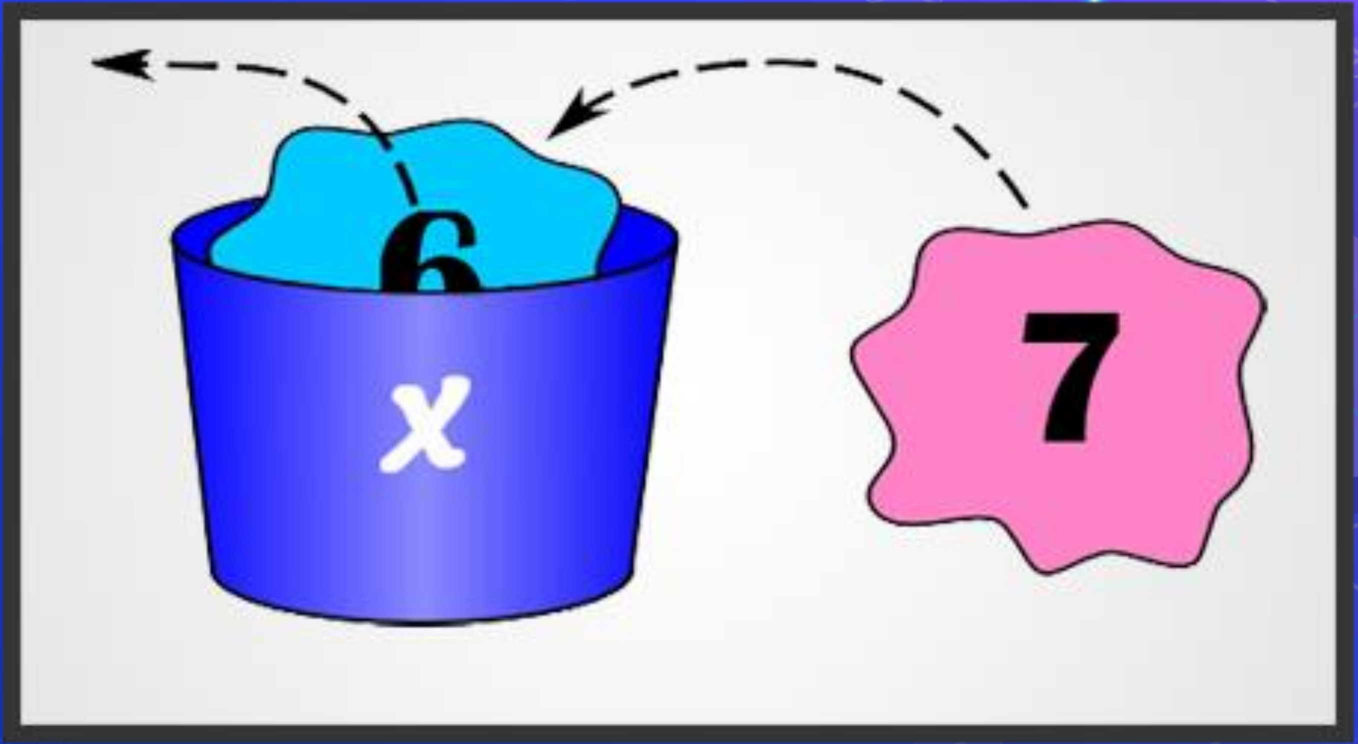


```python
# making a single line comment
name = 'ahmed'


print('hello') # inline comment


"""
multi line comments
you can make many lines
"""
```

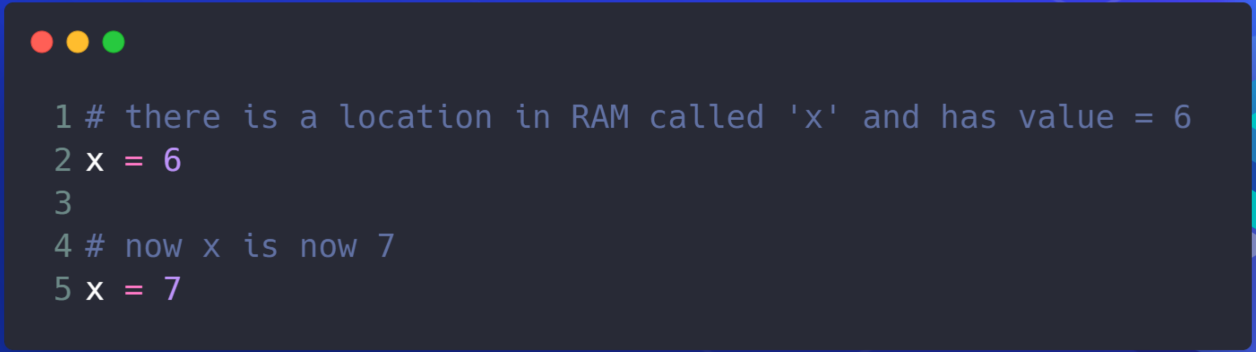
Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- **Variables**
- Data types
  - Numbers & Math
  - Boolean & Comparison and Logic
  - Strings
  - Lists
  - Tuples
  - Sets
  - Dictionaries
- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, …)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Variables

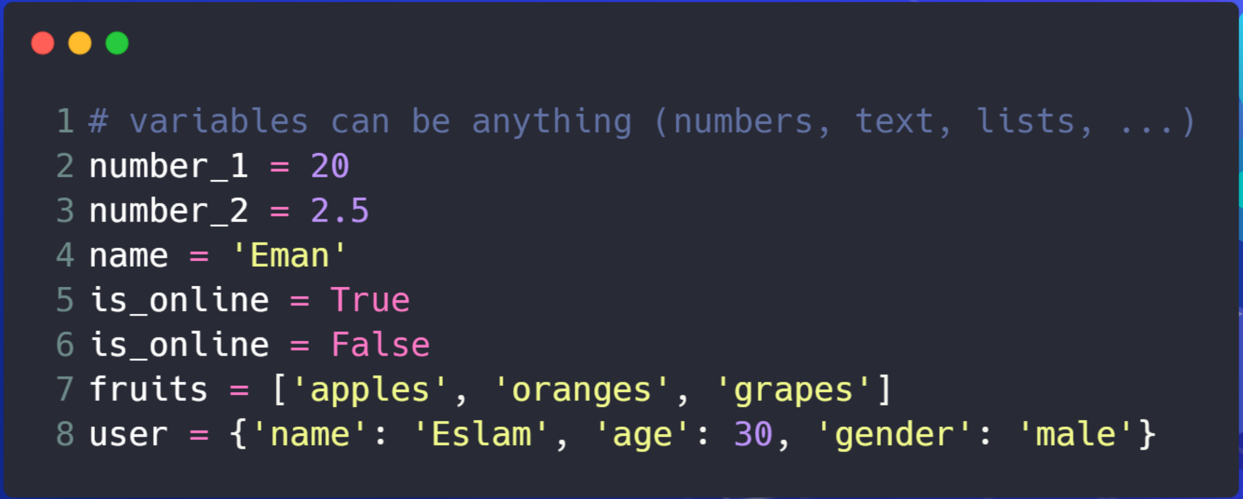


Variables



```
# there is a location in RAM called 'x' and has value = 6
x = 6

# now x is now 7
x = 7
```

Variables



```python
# variables can be anything (numbers, text, lists, ...)
number_1 = 20
number_2 = 2.5
name = 'Eman'
is_online = True
is_online = False
fruits = ['apples', 'oranges', 'grapes']
user = {'name': 'Eslam', 'age': 30, 'gender': 'male'}
```

Variables

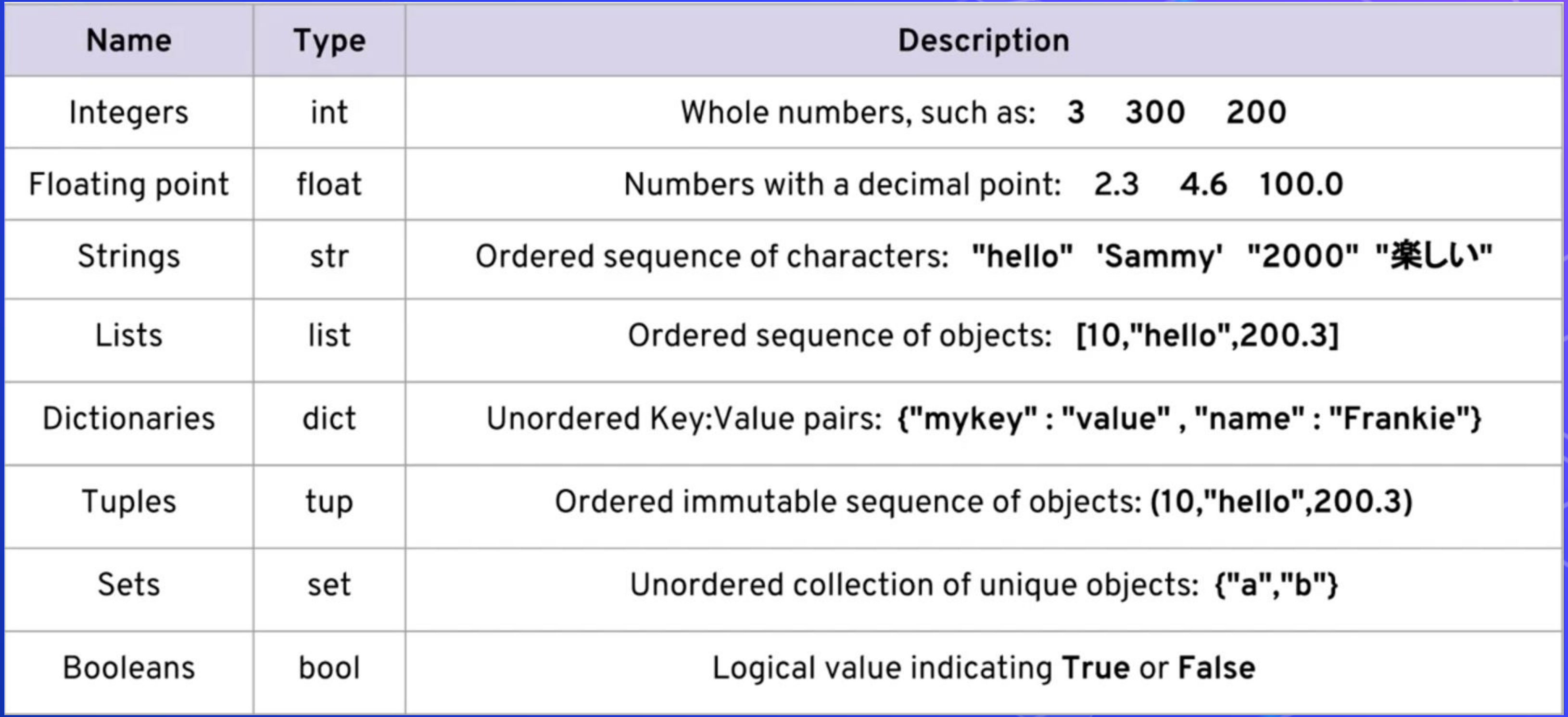
Rules for variables assignment

- Names cannot start with numbers.
- There can be no spaces, use _ instead.
- Can’t use these symbols :”’<>/?|\()!@#$%^&*~-+=[],.
- Avoid using words that have special meaning in Python like list.
- For best practice user lowercase letters.
- Use easy and meaningful name and related to the problem.

Introduction to Python programming Course Outline

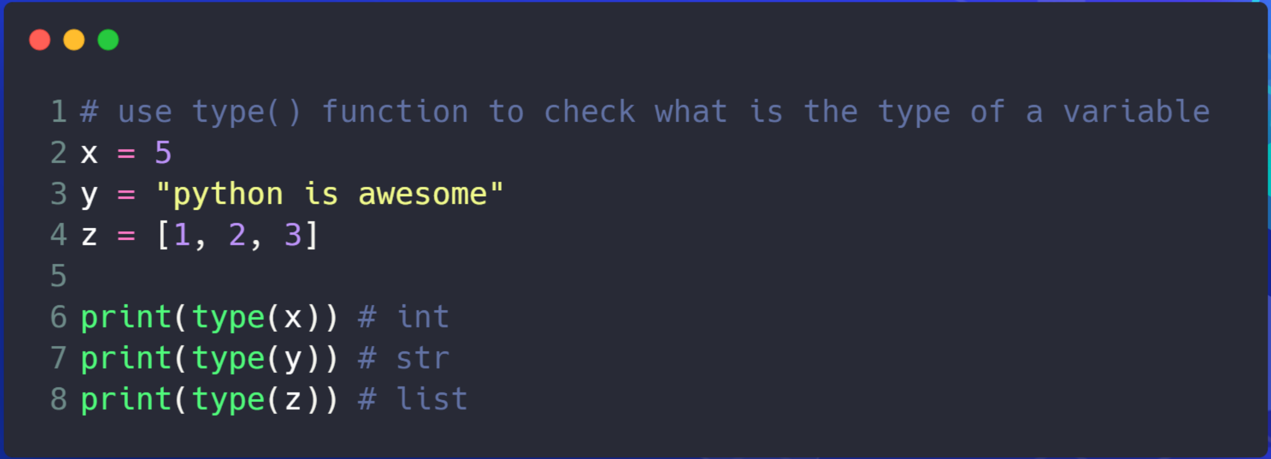
- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
  - Numbers & Math
  - Boolean & Comparison and Logic
  - Strings
  - Lists
  - Tuples
  - Sets
  - Dictionaries
- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Data types



Name	Type	Description
Integers	int	Whole numbers, such as: **3 300 200**
Floating point	float	Numbers with a decimal point: **2.3 4.6 100.0**
Strings	str	Ordered sequence of characters: **"hello" 'Sammy' "2000" "楽しい"**
Lists	list	Ordered sequence of objects: **[10,"hello",200.3]**
Dictionaries	dict	Unordered Key:Value pairs: **{"mykey" : "value" , "name" : "Frankie"}**
Tuples	tup	Ordered immutable sequence of objects: **(10,"hello",200.3)**
Sets	set	Unordered collection of unique objects: **{"a","b"}**
Booleans	bool	Logical value indicating **True** or **False**

Data types

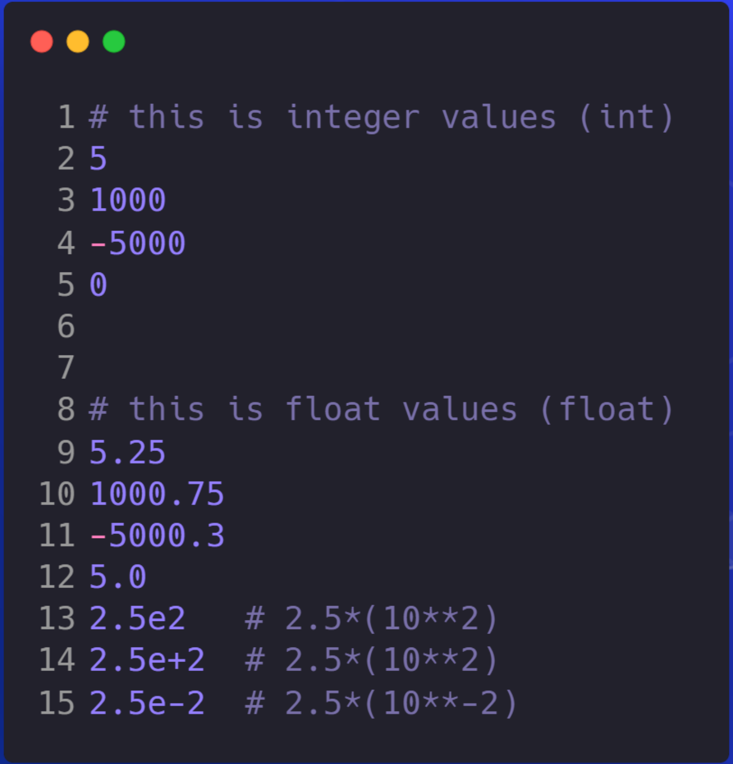


```python
# use type() function to check what is the type of a variable
x = 5
y = "python is awesome"
z = [1, 2, 3]

print(type(x)) # int
print(type(y)) # str
print(type(z)) # list
```

Introduction to Python programming Course Outline

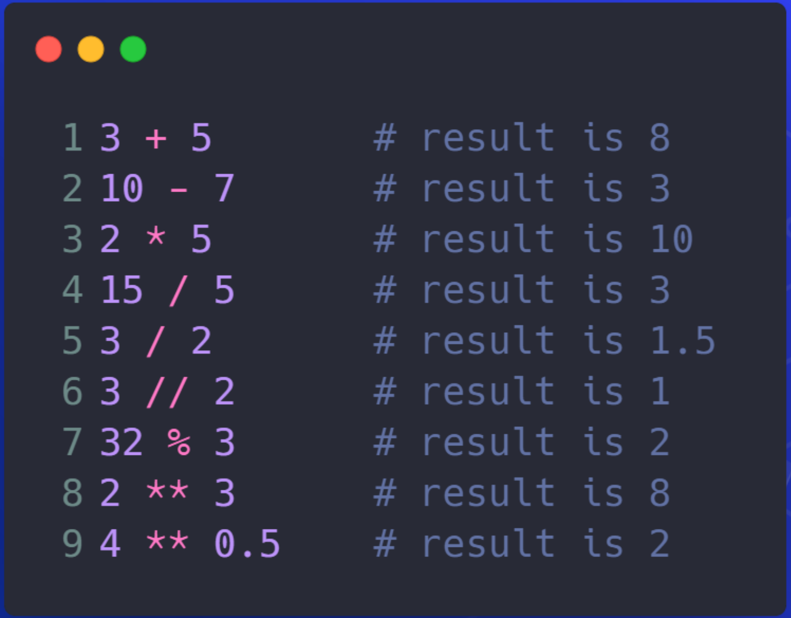
- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
  - **Numbers & Math**
  - Boolean & Comparison and Logic
  - Strings
  - Lists
  - Tuples
  - Sets
  - Dictionaries
- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, …)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Numbers & Math



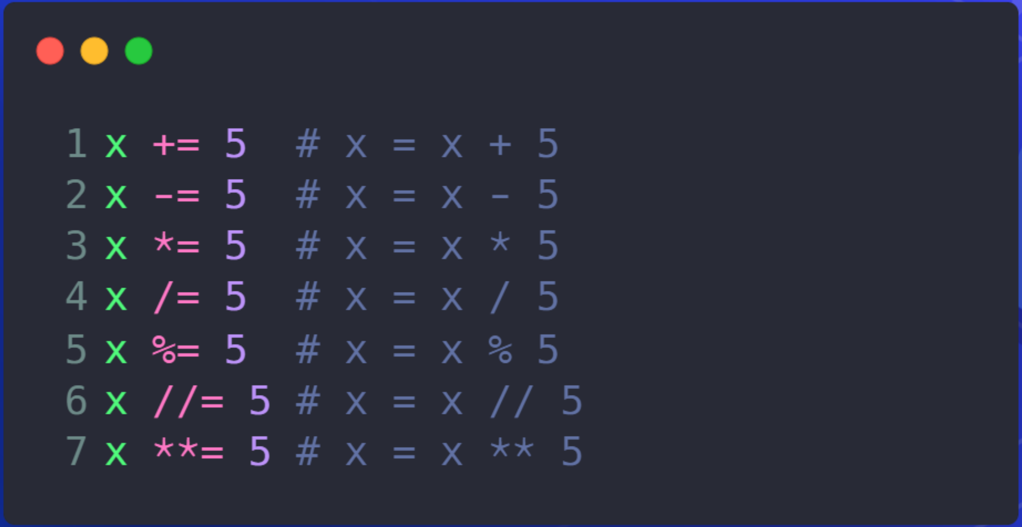
```
# this is integer values (int)
5
1000
-5000
0


# this is float values (float)
5.25
1000.75
-5000.3
5.0
2.5e2   # 2.5*(10**2)
2.5e+2  # 2.5*(10**2)
2.5e-2  # 2.5*(10**-2)
```

Numbers & Math



```
3 + 5          # result is 8
10 - 7         # result is 3
2 * 5          # result is 10
15 / 5         # result is 3
3 / 2          # result is 1.5
3 // 2         # result is 1
32 % 3         # result is 2
2 ** 3         # result is 8
4 ** 0.5       # result is 2
```

Numbers & Math

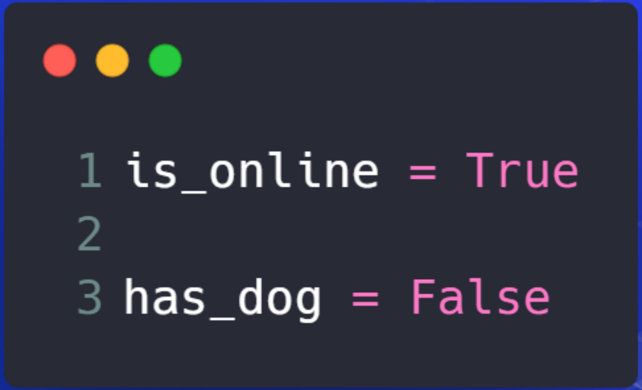


```
x += 5  # x = x + 5
x -= 5  # x = x - 5
x *= 5  # x = x * 5
x /= 5  # x = x / 5
x %= 5  # x = x % 5
x //= 5 # x = x // 5
x **= 5 # x = x ** 5
```

Introduction to Python programming Course Outline

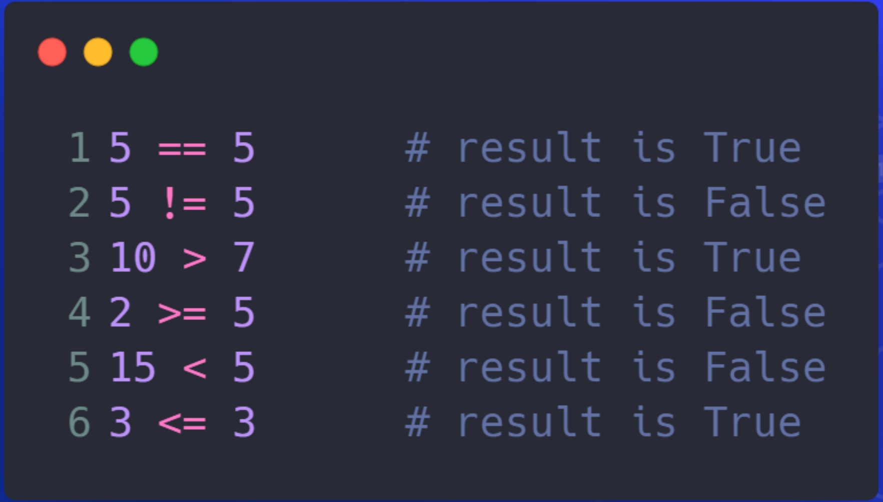
- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
  - Numbers & Math
  - **Boolean & Comparison and Logic**
  - Strings
  - Lists
  - Tuples
  - Sets
  - Dictionaries
- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, …)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Boolean & Comparison and Logic



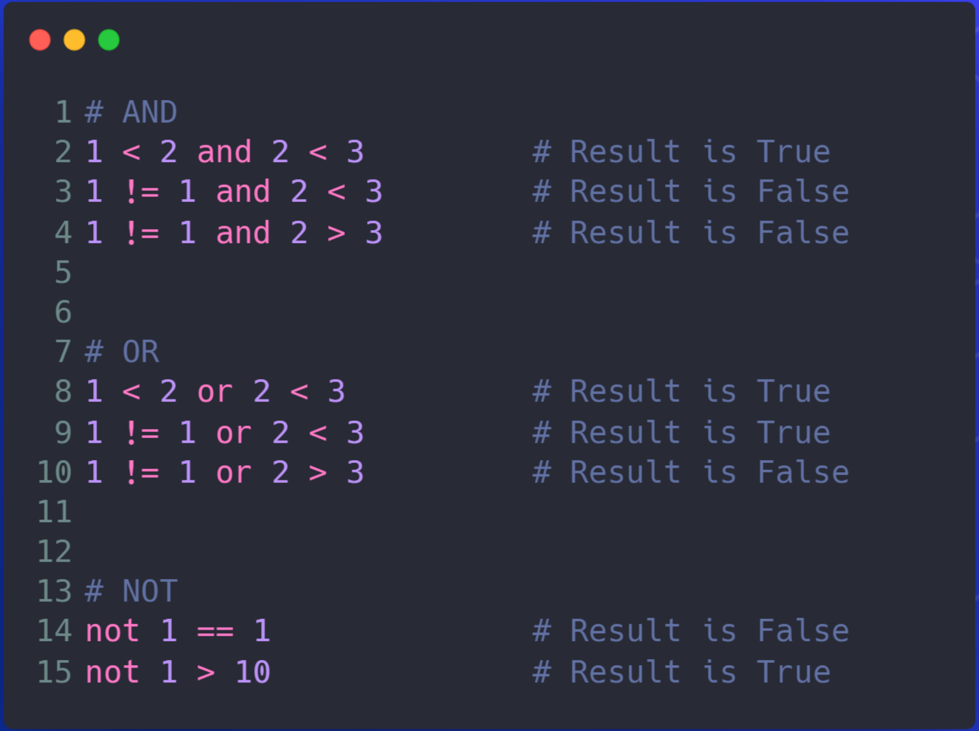
```
is_online = True

has_dog = False
```

Boolean & Comparison and Logic



```
5 == 5      # result is True
5 != 5      # result is False
10 > 7      # result is True
2 >= 5      # result is False
15 < 5      # result is False
3 <= 3      # result is True
```

Boolean & Comparison and Logic

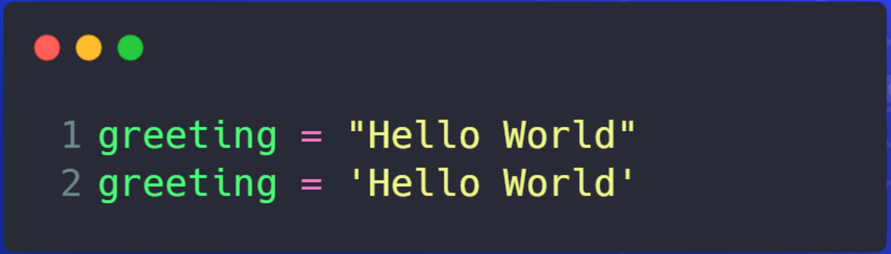


```
# AND
1 < 2 and 2 < 3            # Result is True
1 != 1 and 2 < 3           # Result is False
1 != 1 and 2 > 3           # Result is False


# OR
1 < 2 or 2 < 3             # Result is True
1 != 1 or 2 < 3            # Result is True
1 != 1 or 2 > 3            # Result is False


# NOT
not 1 == 1                 # Result is False
not 1 > 10                 # Result is True
```

Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
  - Numbers & Math
  - Boolean & Comparison and Logic
  - **Strings**
  - Lists
  - Tuples
  - Sets
  - Dictionaries
- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, …)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Strings

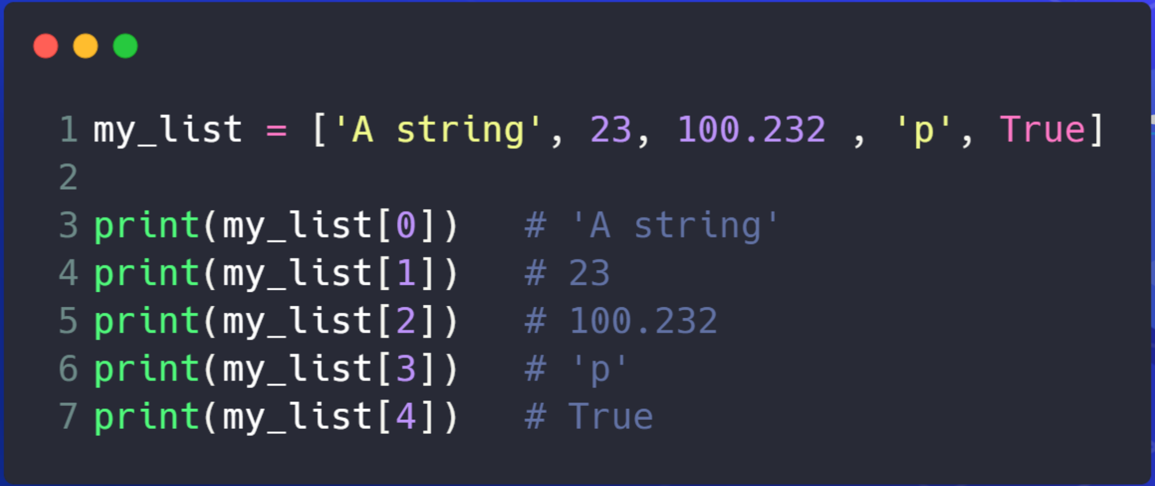


```
greeting = "Hello World"
greeting = 'Hello World'
```

Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
  - Numbers & Math
  - Boolean & Comparison and Logic
  - Strings
  - Lists
  - Tuples
  - Sets
  - Dictionaries
- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, …)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Lists



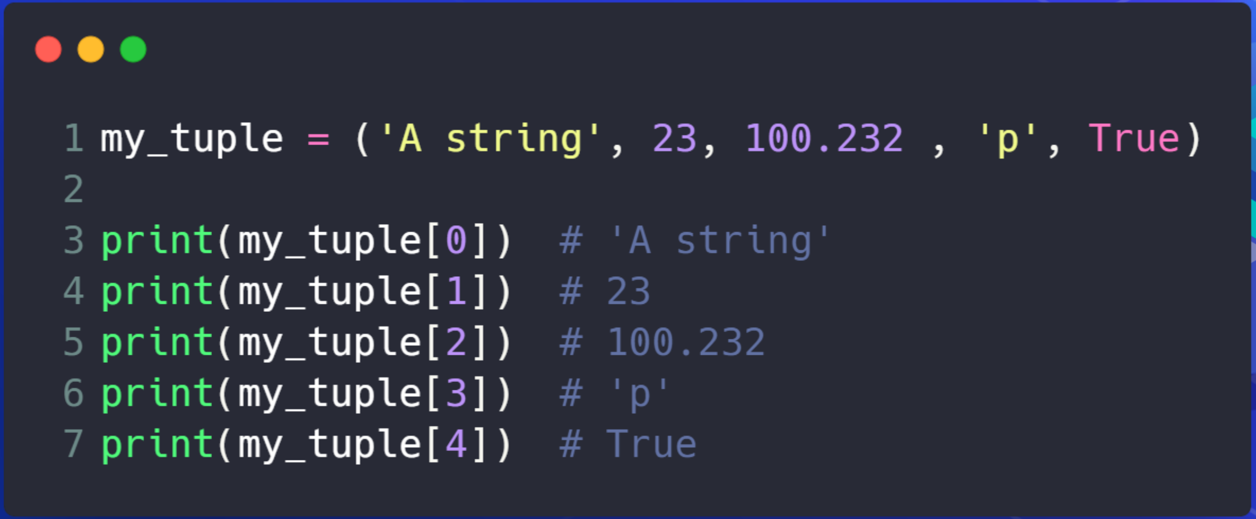
```
my_list = ['A string', 23, 100.232 , 'p', True]

print(my_list[0])   # 'A string'
print(my_list[1])   # 23
print(my_list[2])   # 100.232
print(my_list[3])   # 'p'
print(my_list[4])   # True
```

Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
  - Numbers & Math
  - Boolean & Comparison and Logic
  - Strings
  - Lists
  - <u>Tuples</u>
  - Sets
  - Dictionaries
- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Tuples (faster and immutable lists)

Used when you have immutable values and need faster processing on them



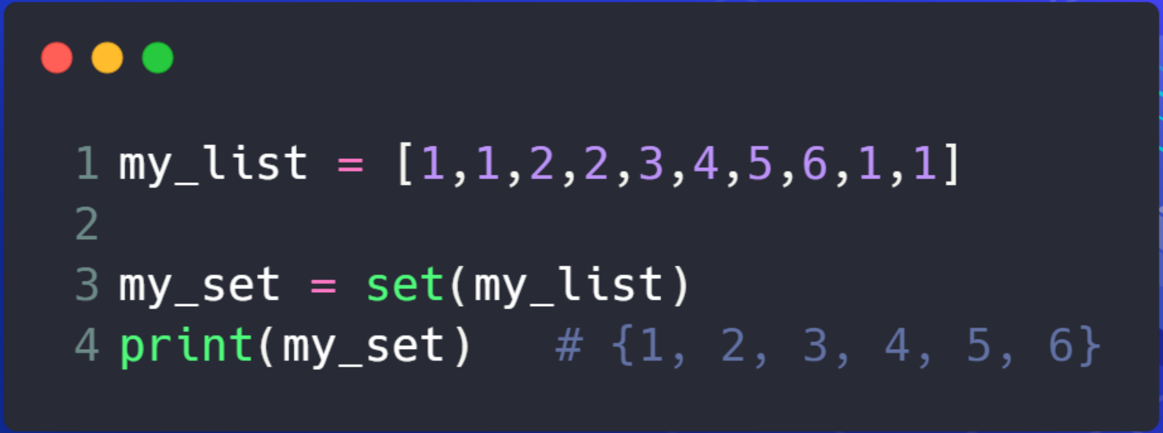
```python
my_tuple = ('A string', 23, 100.232 , 'p', True)

print(my_tuple[0])  # 'A string'
print(my_tuple[1])  # 23
print(my_tuple[2])  # 100.232
print(my_tuple[3])  # 'p'
print(my_tuple[4])  # True
```

Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
  - Numbers & Math
  - Boolean & Comparison and Logic
  - Strings
  - Lists
  - Tuples
  - <u>Sets</u>
  - Dictionaries
- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, …)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Sets (unique lists)

Used for intersections & union operations

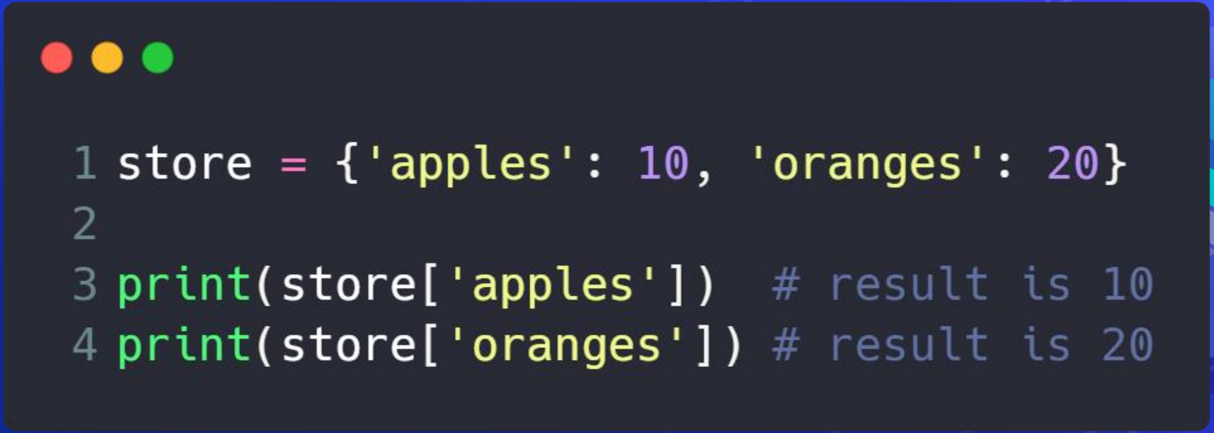


```python
my_list = [1,1,2,2,3,4,5,6,1,1]

my_set = set(my_list)
print(my_set)   # {1, 2, 3, 4, 5, 6}
```

Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
  - Numbers & Math
  - Boolean & Comparison and Logic
  - Strings
  - Lists
  - Tuples
  - Sets
  - <u>Dictionaries</u>
- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Dictionaries

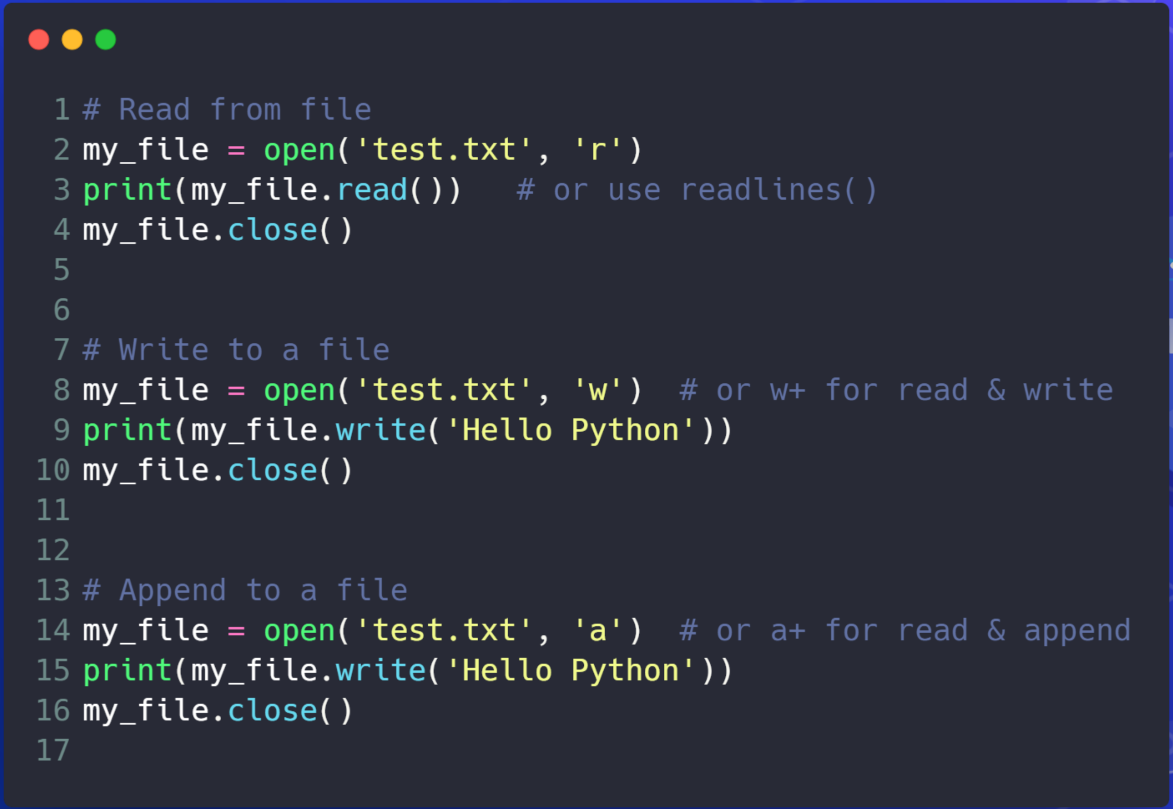


```python
store = {'apples': 10, 'oranges': 20}

print(store['apples'])  # result is 10
print(store['oranges']) # result is 20
```

Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
  - Numbers & Math
  - Boolean & Comparison and Logic
  - Strings
  - Lists
  - Tuples
  - Sets
  - Dictionaries
- <u>**File Handling**</u>
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, …)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

File Handling

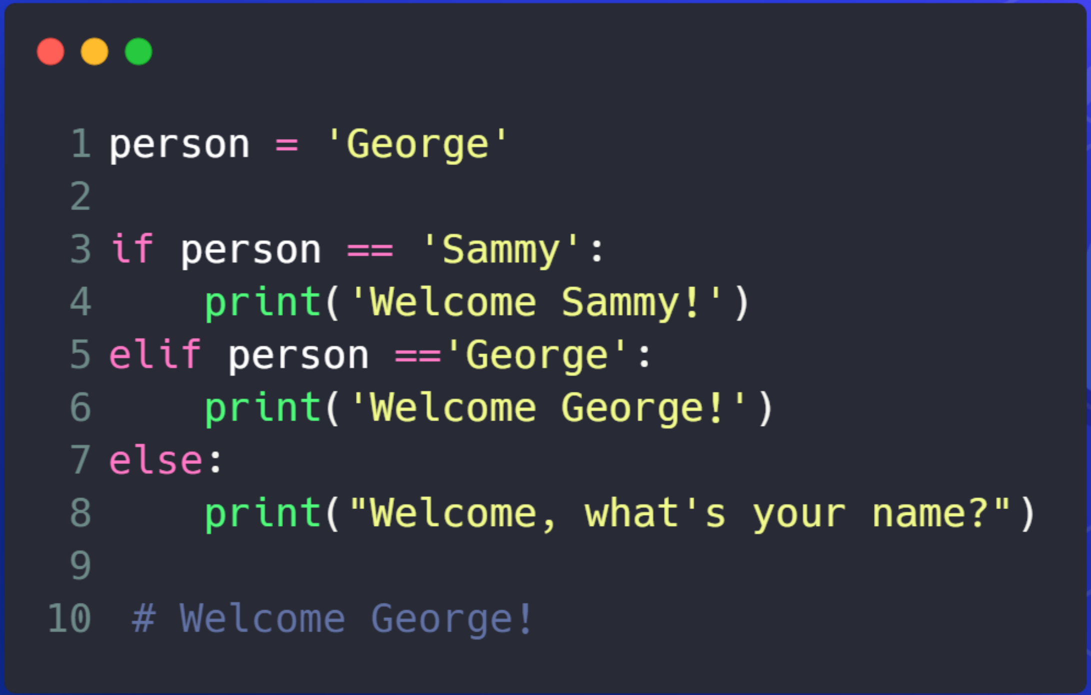


```python
# Read from file
my_file = open('test.txt', 'r')
print(my_file.read())    # or use readlines()
my_file.close()


# Write to a file
my_file = open('test.txt', 'w')  # or w+ for read & write
print(my_file.write('Hello Python'))
my_file.close()


# Append to a file
my_file = open('test.txt', 'a')  # or a+ for read & append
print(my_file.write('Hello Python'))
my_file.close()

```

Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
  - Numbers & Math
  - Boolean & Comparison and Logic
  - Strings
  - Lists
  - Tuples
  - Sets
  - Dictionaries
- File Handling
- **If Conditions**
- For Loops
- Built-in functions & Operators (zip, enumerate, range, …)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

If Conditions

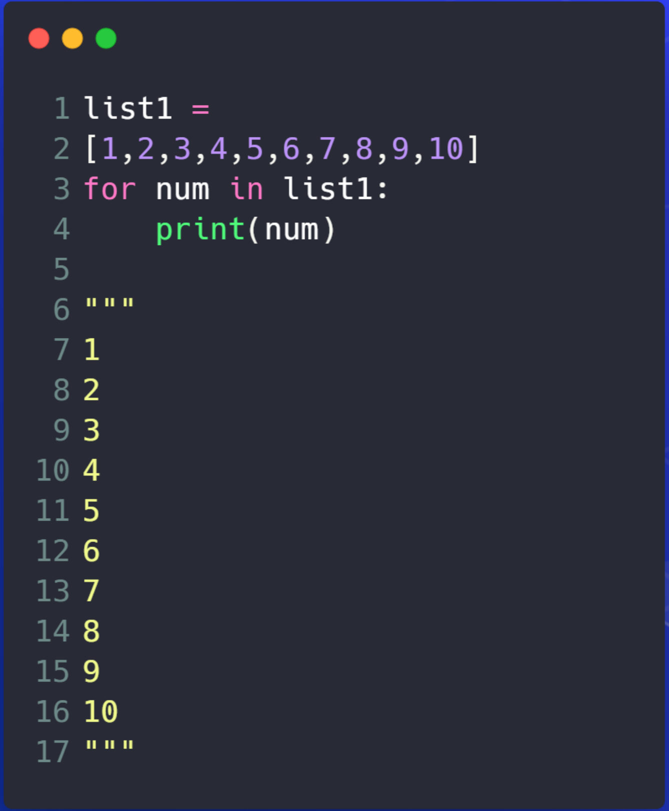


```python
person = 'George'

if person == 'Sammy':
    print('Welcome Sammy!')
elif person =='George':
    print('Welcome George!')
else:
    print("Welcome, what's your name?")

 # Welcome George!
```

Introduction to Python programming Course Outline

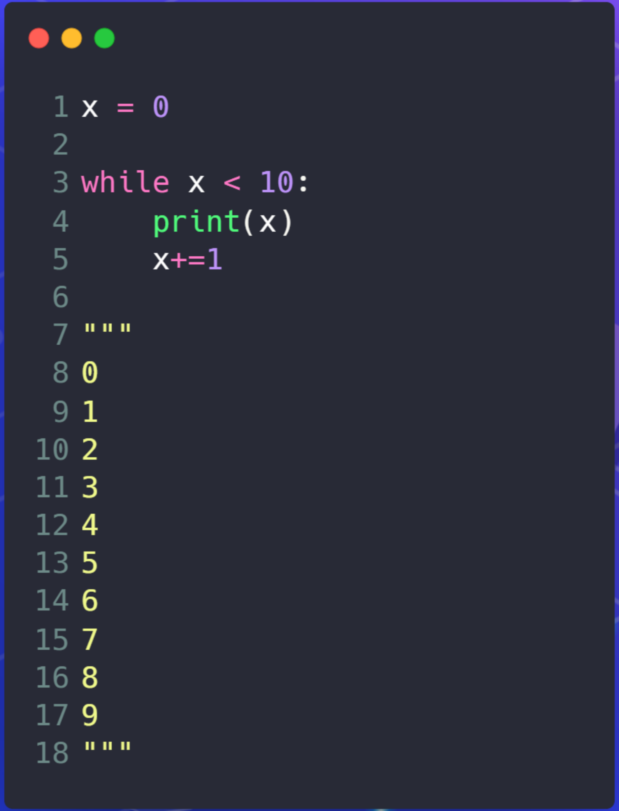
- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
  - Numbers & Math
  - Boolean & Comparison and Logic
  - Strings
  - Lists
  - Tuples
  - Sets
  - Dictionaries
- File Handling
- If Conditions
- **For Loops**
- Built-in functions & Operators (zip, enumerate, range, …)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

For Loops



```
list1 =
[1,2,3,4,5,6,7,8,9,10]
for num in list1:
    print(num)

"""
1
2
3
4
5
6
7
8
9
10
"""
```

While Loops



```
x = 0

while x < 10:
    print(x)
    x+=1

"""
0
1
2
3
4
5
6
7
8
9
"""
```

Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
  - Numbers & Math
  - Boolean & Comparison and Logic
  - Strings
  - Lists
  - Tuples
  - Sets
  - Dictionaries
- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

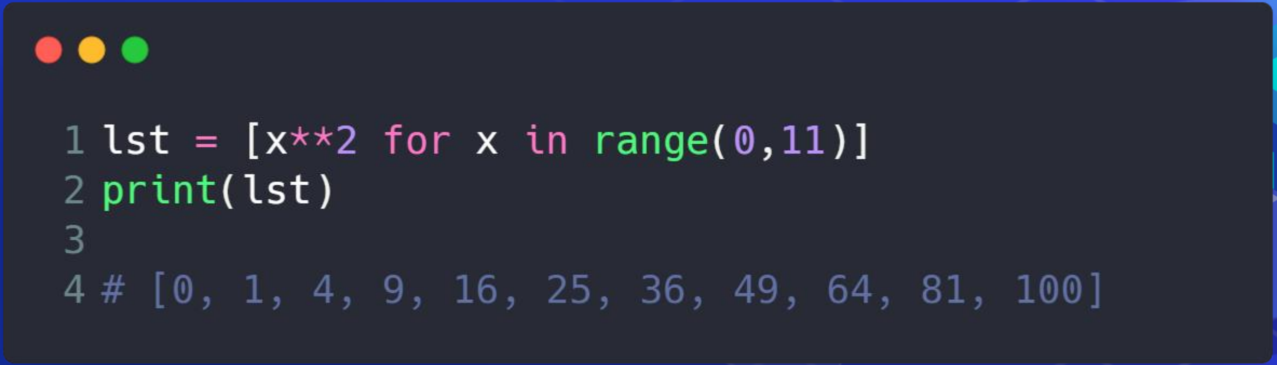
Built-in functions & Operators

- Range
- Enumerate
- Zip
- In
- ...

Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
  - Numbers & Math
  - Boolean & Comparison and Logic
  - Strings
  - Lists
  - Tuples
  - Sets
  - Dictionaries
- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- **List Comprehensions**
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

List Comprehensions

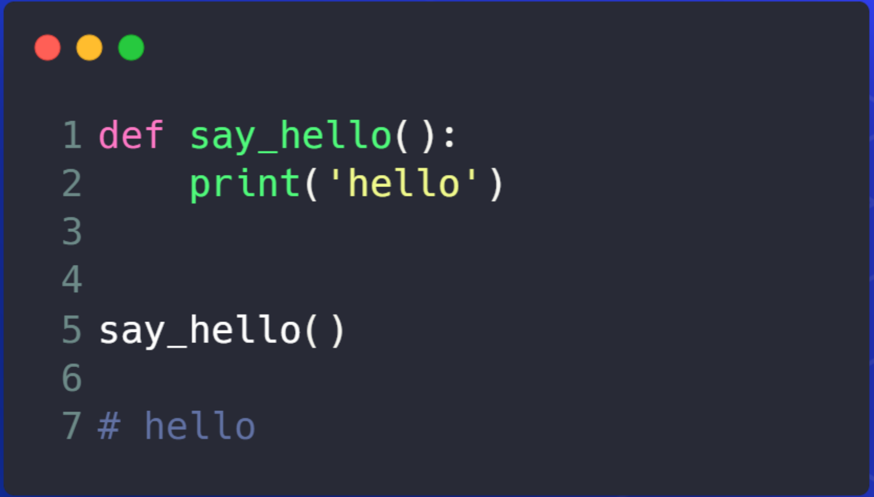


```
lst = [x**2 for x in range(0,11)]
print(lst)

# [0, 1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
```

Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
  - Numbers & Math
  - Boolean & Comparison and Logic
  - Strings
  - Lists
  - Tuples
  - Sets
  - Dictionaries
- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, …)
- List Comprehensions
- **Functions**
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- Modules & Packages

Functions



```python
def say_hello():
    print('hello')


say_hello()

# hello
```

Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
  - Numbers & Math
  - Boolean & Comparison and Logic
  - Strings
  - Lists
  - Tuples
  - Sets
  - Dictionaries
- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- **Lambda Expressions**
- **Map, Filter, Reduce**
- Variables Scope
- Modules & Packages

Lambda Expressions & Map, Filter, Reduce

- Map
- Filter
- Reduce
- Lambda Expression

Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
  - Numbers & Math
  - Boolean & Comparison and Logic
  - Strings
  - Lists
  - Tuples
  - Sets
  - Dictionaries
- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, ...)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- **Variables Scope**
- Modules & Packages

Variables Scope

- Local
- Global

Introduction to Python programming Course Outline

- Intro to Computer Science
- Environment Setup (Anaconda)
- Command Line
- Conda & pip package managers
- Jupyter Notebook
- Input & Output
- Variables
- Data types
  - Numbers & Math
  - Boolean & Comparison and Logic
  - Strings
  - Lists
  - Tuples
  - Sets
  - Dictionaries
- File Handling
- If Conditions
- For Loops
- Built-in functions & Operators (zip, enumerate, range, …)
- List Comprehensions
- Functions
- Lambda Expressions
- Map, Filter, Reduce
- Variables Scope
- **Modules & Packages**

Modules & Packages

- Module is a file which contains various Python functions and global variables. It is simply just .py extension file which has python executable code.

- Package is a collection of modules.

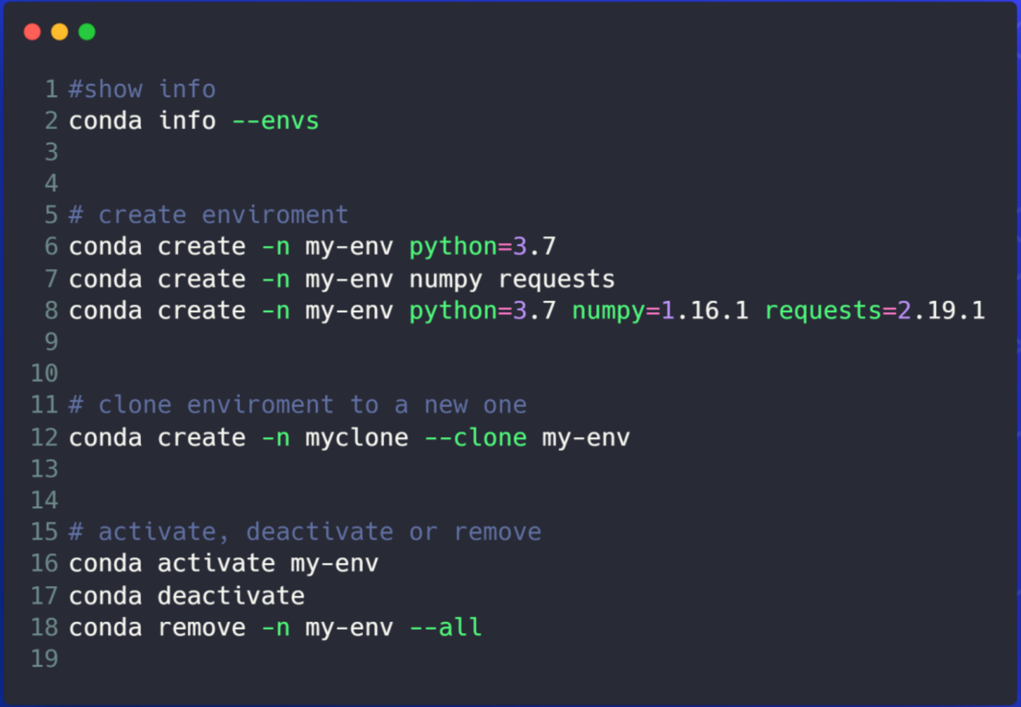
- Library is a collection of packages.

- Framework is a collection of libraries.

Modules & Packages

- Random
- OS
- Datetime
- Install external code
- How to make your custom module
- How to make your custom package

Conda enviroments

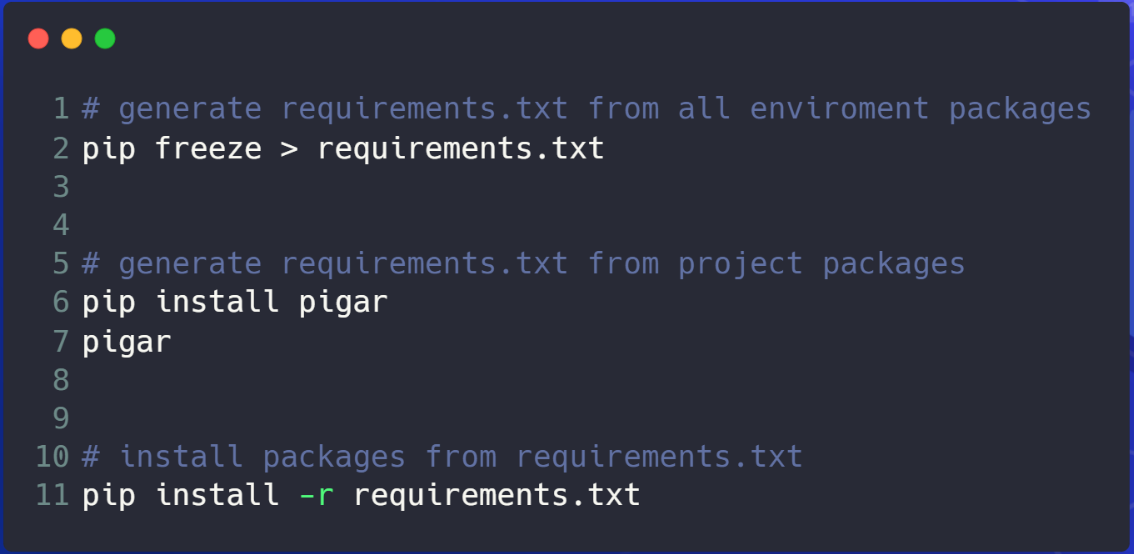


```
#show info
conda info --envs


# create enviroment
conda create -n my-env python=3.7
conda create -n my-env numpy requests
conda create -n my-env python=3.7 numpy=1.16.1 requests=2.19.1


# clone enviroment to a new one
conda create -n myclone --clone my-env


# activate, deactivate or remove
conda activate my-env
conda deactivate
conda remove -n my-env --all

```

Don’t forget to add jupyter kernel for this newly created enviroment

https://janakiev.com/blog/jupyter-virtual-envs/

Conda enviroments



```
# generate requirements.txt from all enviroment packages
pip freeze > requirements.txt


# generate requirements.txt from project packages
pip install pigar
pigar


# install packages from requirements.txt
pip install -r requirements.txt
```

Time for Thanos.py >_

Questions ?!

Thanks!

>_ Live long and prosper