Data Science with Python and R

Moinul Islam

9/15/2022

Table of contents

Preface

This book contains the basic Python and R programming skills to learn data science. You can dive into the amazing data science world by using Python and R. Learn how to effectively analyze and visualize your data. No coding experience or skills needed. Data Science is about data gathering, analysis and decision-making. Data Science is used in many industries in the world today, e.g. banking, consultancy, healthcare, and manufacturing. Data science is not a one-step process such that will get to learned it in a short time. It's passes from many stages and every element is important. One should always follow the proper steps to reach the ladder. Every step has its value and it counts in your model.

Figure 1: Data Science

1 Basic Python

1.1 Prerequisites

- There are three things we need to run the code in this lecture:
  - Python 3.10 (https://www.python.org/downloads/)
  - PyCharm community or professional (https://www.jetbrains.com/pycharm/)
  - A handful of other packages

1.2 Variables

- Let's start using a variable
- Variable names can contain only letters, numbers and underscores
- Space are not allowed in variable name
- Variables name should be short and descriptive

```
message = "Hello world!"
print(message)
```

```
Hello world!
```

1.3 Strings

- A *string* is a series of characters
- In python, anything inside a quotes is considered a string
- We can use single or double quotes around the strings

```
message = "Python is a programming language"
print(message)
```

```
Python is a programming language
```

```
message = 'I told my friend, "Python is a programming language"'
print(message)
```

I told my friend, "Python is a programming language"

1.3.1 Change case of string

- A lower case string can be changed to a title

```
name = "moinul islam"
print(name.title())
```

Moinul Islam

- We can also change the string to all upper or all lower

```
name = "moinul islam"
print(name.upper())
```

MOINUL ISLAM

1.4 Numbers

- Numbers are used quite often in python
  - Integers: We can add (+), multiply (*), exponent (**), and divide (/) integers in Python

```
print(2+3)
```

5

```
print(3-2)
```

1

```
print(2*3)
```

  6

```
print(3**3)
```

  27

```
print(3/2)
```

  1.5

  - Python supports the order of operation too.

```
print(2 + 3*4)
```

  14

```
print((2+3)*4)
```

  20

  - Floats: Python calls any number with a decimal point a *float*

```
print(2*0.2)
```

  0.4

  - Integers and floats: When we divide two numbers, even if they are integers, you will always get a float

```
print(4/2)
```

  2.0

- Underscore in numbers
  - When you are writing long numbers, you can group digits using underscores

```
universe_age = 14_000_000_000
print(universe_age)
```

```
14000000000
```

- Multiple assignments
  - We can assign the value to more than one variables using just a single line

```
x, y, z = 12, 3, 5
print(x, y, z)
print(z)
```

```
12 3 5

5
```

- Constants
  - A *constant* is a variable whose value stay the same throughout the life of a program.
  - Python does not have built in command for constant
  - However, a variable name with all capital letters treated as constant

```
ELON_MASK  = 1_000_000_000_000
print(ELON_MASK)
```

```
1000000000000
```

1.5 Comments

- In Python, hash (#) indicates a comment.
- Anything following a # mark in your code is ignored by Python

```
# Say hello to your friends
print("Hello friends")
```

```
Hello friends
```

1.6 Lists

- List the elements of a variable

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles)
```

['trek', 'cannondale', 'redline', 'specialized']

- Access an element from a list

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0])
```

trek

1.6.1 Neat outcome

- You can format the element "trek" even more neatly by using title() method

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0].title())
```

Trek

1.6.2 Print string

- You can print the strings that you are interested
- It starts counting from zero

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[1])
```

cannondale

- Python has special syntax to call the last item of a list

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[-1])
```

specialized

1.7 Functions

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
message = f"my first bicycle was a {bicycles[0].title()}"
print(message)
```

```
my first bicycle was a Trek
```

- Modifying elements in a list

2 Introduction to Python

2.1 Prerequisites

- There are three things we need to run the code in this lecture:
  - Python 3.10 (https://www.python.org/downloads/)
  - PyCharm community or professional (https://www.jetbrains.com/pycharm/)
  - A handful of other packages

2.2 Variables

- Let's start using a variable
- Variable names can contain only letters, numbers and underscores
- Space are not allowed in variable name
- Variables name should be short and descriptive

```
message = "Hello world!"
print(message)
```

```
Hello world!
```

2.3 Strings

- A *string* is a series of characters
- In python, anything inside a quotes is considered a string
- We can use single or double quotes around the strings

```
message = "Python is a programming language"
print(message)
```

```
Python is a programming language
```

```python
message = 'I told my friend, "Python is a programming language"'
print(message)
```

I told my friend, “Python is a programming language”

2.3.1 Change case of string

- A lower case string can be changed to a title

```python
name = "moinul islam"
print(name.title())
```

Moinul Islam

- We can also change the string to all upper or all lower

```python
name = "moinul islam"
print(name.upper())
```

MOINUL ISLAM

2.4 Numbers

- Numbers are used quite often in python
  - Integers: We can add (+), multiply (*), exponent (**), and divide (/) integers in Python

```python
print(2+3)
```

5

```python
print(3-2)
```

1

```python
print(2*3)
```

```
6
```

```python
print(3**3)
```

```
27
```

```python
print(3/2)
```

```
1.5
```

- Python supports the order of operation too.

```python
print(2 + 3*4)
```

```
14
```

```python
print((2+3)*4)
```

```
20
```

- Floats: Python calls any number with a decimal point a *float*

```python
print(2*0.2)
```

```
0.4
```

- Integers and floats: When we divide two numbers, even if they are integers, you will always get a float

```python
print(4/2)
```

```
2.0
```

- Underscore in numbers
  - When you are writing long numbers, you can group digits using underscores

```python
universe_age = 14_000_000_000
print(universe_age)
```

```
14000000000
```

- Multiple assignments
  - We can assign the value to more than one variables using just a single line

```
x, y, z = 12, 3, 5
print(x, y, z)
print(z)
```

```
12 3 5

5
```

- Constants
  - A *constant* is a variable whose value stay the same throughout the life of a program.
  - Python does not have built in command for constant
  - However, a variable name with all capital letters treated as constant

```
ELON_MASK  = 1_000_000_000_000
print(ELON_MASK)
```

```
1000000000000
```

2.5 Comments

- In Python, hash (#) indicates a comment.
- Anything following a # mark in your code is ignored by Python

```
# Say hello to your friends
print("Hello friends")
```

```
Hello friends
```

2.6 Lists

- List the elements of a variable

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles)
```

['trek', 'cannondale', 'redline', 'specialized']

- Access an element from a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0])
```

trek

2.6.1 Neat outcome

- You can format the element "trek" even more neatly by using title() method

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0].title())
```

Trek

2.6.2 Print string

- You can print the strings that you are interested
- It starts counting from zero

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[1])
```

cannondale

- Python has special syntax to call the last item of a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[-1])
```

specialized

2.7 Functions

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
message = f"my first bicycle was a {bicycles[0].title()}"
print(message)
```

```
my first bicycle was a Trek
```

- Modifying elements in a list

3 Introduction to Python

3.1 Prerequisites

- There are three things we need to run the code in this lecture:
  - Python 3.10 (https://www.python.org/downloads/)
  - PyCharm community or professional (https://www.jetbrains.com/pycharm/)
  - A handful of other packages

3.2 Variables

- Let's start using a variable
- Variable names can contain only letters, numbers and underscores
- Space are not allowed in variable name
- Variables name should be short and descriptive

```
message = "Hello world!"
print(message)
```

```
Hello world!
```

3.3 Strings

- A *string* is a series of characters
- In python, anything inside a quotes is considered a string
- We can use single or double quotes around the strings

```
message = "Python is a programming language"
print(message)
```

```
Python is a programming language
```

```python
message = 'I told my friend, "Python is a programming language"'
print(message)
```

I told my friend, “Python is a programming language”

3.3.1 Change case of string

- A lower case string can be changed to a title

```python
name = "moinul islam"
print(name.title())
```

Moinul Islam

- We can also change the string to all upper or all lower

```python
name = "moinul islam"
print(name.upper())
```

MOINUL ISLAM

3.4 Numbers

- Numbers are used quite often in python
  - Integers: We can add (+), multiply (*), exponent (**), and divide (/) integers in Python

```python
print(2+3)
```

5

```python
print(3-2)
```

1

```python
print(2*3)
```

    6

```
print(3**3)
```

    27

```
print(3/2)
```

    1.5

  - Python supports the order of operation too.

```
print(2 + 3*4)
```

    14

```
print((2+3)*4)
```

    20

  - Floats: Python calls any number with a decimal point a *float*

```
print(2*0.2)
```

    0.4

  - Integers and floats: When we divide two numbers, even if they are integers, you will always get a float

```
print(4/2)
```

    2.0

- Underscore in numbers
  - When you are writing long numbers, you can group digits using underscores

```
universe_age = 14_000_000_000
print(universe_age)
```

```
14000000000
```

- Multiple assignments
    - We can assign the value to more than one variables using just a single line

```
x, y, z = 12, 3, 5
print(x, y, z)
print(z)
```

```
12 3 5
5
```

- Constants
    - A *constant* is a variable whose value stay the same throughout the life of a program.
    - Python does not have built in command for constant
    - However, a variable name with all capital letters treated as constant

```
ELON_MASK  = 1_000_000_000_000
print(ELON_MASK)
```

```
1000000000000
```

3.5 Comments

- In Python, hash (#) indicates a comment.
- Anything following a # mark in your code is ignored by Python

```
# Say hello to your friends
print("Hello friends")
```

```
Hello friends
```

3.6 Lists

- List the elements of a variable

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles)
```

```
['trek', 'cannondale', 'redline', 'specialized']
```

- Access an element from a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0])
```

```
trek
```

3.6.1 Neat outcome

- You can format the element "trek" even more neatly by using title() method

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0].title())
```

```
Trek
```

3.6.2 Print string

- You can print the strings that you are interested
- It starts counting from zero

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[1])
```

```
cannondale
```

- Python has special syntax to call the last item of a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[-1])
```

```
specialized
```

3.7 Functions

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
message = f"my first bicycle was a {bicycles[0].title()}"
print(message)
```

```
my first bicycle was a Trek
```

- Modifying elements in a list

4 Introduction to Python

4.1 Prerequisites

- There are three things we need to run the code in this lecture:
  - Python 3.10 (https://www.python.org/downloads/)
  - PyCharm community or professional (https://www.jetbrains.com/pycharm/)
  - A handful of other packages

4.2 Variables

- Let's start using a variable
- Variable names can contain only letters, numbers and underscores
- Space are not allowed in variable name
- Variables name should be short and descriptive

```
message = "Hello world!"
print(message)
```

```
Hello world!
```

4.3 Strings

- A *string* is a series of characters
- In python, anything inside a quotes is considered a string
- We can use single or double quotes around the strings

```
message = "Python is a programming language"
print(message)
```

```
Python is a programming language
```

```python
message = 'I told my friend, "Python is a programming language"'
print(message)
```

I told my friend, “Python is a programming language”

4.3.1 Change case of string

- A lower case string can be changed to a title

```python
name = "moinul islam"
print(name.title())
```

Moinul Islam

- We can also change the string to all upper or all lower

```python
name = "moinul islam"
print(name.upper())
```

MOINUL ISLAM

4.4 Numbers

- Numbers are used quite often in python
    - Integers: We can add (+), multiply (*), exponent (**), and divide (/) integers in Python

```python
print(2+3)
```

5

```python
print(3-2)
```

1

```python
print(2*3)
```

```
6
```

```python
print(3**3)
```

```
27
```

```python
print(3/2)
```

```
1.5
```

  - Python supports the order of operation too.

```python
print(2 + 3*4)
```

```
14
```

```python
print((2+3)*4)
```

```
20
```

  - Floats: Python calls any number with a decimal point a *float*

```python
print(2*0.2)
```

```
0.4
```

  - Integers and floats: When we divide two numbers, even if they are integers, you will always get a float

```python
print(4/2)
```

```
2.0
```

- Underscore in numbers
  - When you are writing long numbers, you can group digits using underscores

```python
universe_age = 14_000_000_000
print(universe_age)
```

```
14000000000
```

- Multiple assignments
  - We can assign the value to more than one variables using just a single line

```python
x, y, z = 12, 3, 5
print(x, y, z)
print(z)
```

```
12 3 5

5
```

- Constants
  - A *constant* is a variable whose value stay the same throughout the life of a program.
  - Python does not have built in command for constant
  - However, a variable name with all capital letters treated as constant

```python
ELON_MASK  = 1_000_000_000_000
print(ELON_MASK)
```

```
1000000000000
```

4.5 Comments

- In Python, hash (#) indicates a comment.
- Anything following a # mark in your code is ignored by Python

```python
# Say hello to your friends
print("Hello friends")
```

```
Hello friends
```

4.6 Lists

- List the elements of a variable

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles)
```

```
['trek', 'cannondale', 'redline', 'specialized']
```

- Access an element from a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0])
```

```
trek
```

4.6.1 Neat outcome

- You can format the element "trek" even more neatly by using title() method

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0].title())
```

```
Trek
```

4.6.2 Print string

- You can print the strings that you are interested
- It starts counting from zero

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[1])
```

```
cannondale
```

- Python has special syntax to call the last item of a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[-1])
```

```
specialized
```

4.7 Functions

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
message = f"my first bicycle was a {bicycles[0].title()}"
print(message)
```

```
my first bicycle was a Trek
```

- Modifying elements in a list

5 Introduction to Python

5.1 Prerequisites

- There are three things we need to run the code in this lecture:
  - Python 3.10 (https://www.python.org/downloads/)
  - PyCharm community or professional (https://www.jetbrains.com/pycharm/)
  - A handful of other packages

5.2 Variables

- Let's start using a variable
- Variable names can contain only letters, numbers and underscores
- Space are not allowed in variable name
- Variables name should be short and descriptive

```
message = "Hello world!"
print(message)
```

```
Hello world!
```

5.3 Strings

- A *string* is a series of characters
- In python, anything inside a quotes is considered a string
- We can use single or double quotes around the strings

```
message = "Python is a programming language"
print(message)
```

```
Python is a programming language
```

```python
message = 'I told my friend, "Python is a programming language"'
print(message)
```

I told my friend, "Python is a programming language"

5.3.1 Change case of string

- A lower case string can be changed to a title

```python
name = "moinul islam"
print(name.title())
```

Moinul Islam

- We can also change the string to all upper or all lower

```python
name = "moinul islam"
print(name.upper())
```

MOINUL ISLAM

5.4 Numbers

- Numbers are used quite often in python
  - Integers: We can add (+), multiply (*), exponent (**), and divide (/) integers in Python

```python
print(2+3)
```

5

```python
print(3-2)
```

1

```python
print(2*3)
```

```
6
```

```python
print(3**3)
```

```
27
```

```python
print(3/2)
```

```
1.5
```

- Python supports the order of operation too.

```python
print(2 + 3*4)
```

```
14
```

```python
print((2+3)*4)
```

```
20
```

- Floats: Python calls any number with a decimal point a *float*

```python
print(2*0.2)
```

```
0.4
```

- Integers and floats: When we divide two numbers, even if they are integers, you will always get a float

```python
print(4/2)
```

```
2.0
```

- Underscore in numbers
  - When you are writing long numbers, you can group digits using underscores

```python
universe_age = 14_000_000_000
print(universe_age)
```

```
14000000000
```

- Multiple assignments
  - We can assign the value to more than one variables using just a single line

```
x, y, z = 12, 3, 5
print(x, y, z)
print(z)
```

```
12 3 5

5
```

- Constants
  - A *constant* is a variable whose value stay the same throughout the life of a program.
  - Python does not have built in command for constant
  - However, a variable name with all capital letters treated as constant

```
ELON_MASK  = 1_000_000_000_000
print(ELON_MASK)
```

```
1000000000000
```

5.5 Comments

- In Python, hash (#) indicates a comment.
- Anything following a # mark in your code is ignored by Python

```
# Say hello to your friends
print("Hello friends")
```

```
Hello friends
```

5.6 Lists

- List the elements of a variable

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles)
```

['trek', 'cannondale', 'redline', 'specialized']

- Access an element from a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0])
```

trek

5.6.1 Neat outcome

- You can format the element "trek" even more neatly by using title() method

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0].title())
```

Trek

5.6.2 Print string

- You can print the strings that you are interested
- It starts counting from zero

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[1])
```

cannondale

- Python has special syntax to call the last item of a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[-1])
```

specialized

5.7 Functions

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
message = f"my first bicycle was a {bicycles[0].title()}"
print(message)
```

```
my first bicycle was a Trek
```

- Modifying elements in a list

6 Introduction to Python

6.1 Prerequisites

- There are three things we need to run the code in this lecture:
    - Python 3.10 (https://www.python.org/downloads/)
    - PyCharm community or professional (https://www.jetbrains.com/pycharm/)
    - A handful of other packages

6.2 Variables

- Let's start using a variable
- Variable names can contain only letters, numbers and underscores
- Space are not allowed in variable name
- Variables name should be short and descriptive

```python
message = "Hello world!"
print(message)
```

```
Hello world!
```

6.3 Strings

- A *string* is a series of characters
- In python, anything inside a quotes is considered a string
- We can use single or double quotes around the strings

```python
message = "Python is a programming language"
print(message)
```

```
Python is a programming language
```

```
message = 'I told my friend, "Python is a programming language"'
print(message)
```

I told my friend, “Python is a programming language”

6.3.1 Change case of string

- A lower case string can be changed to a title

```
name = "moinul islam"
print(name.title())
```

Moinul Islam

- We can also change the string to all upper or all lower

```
name = "moinul islam"
print(name.upper())
```

MOINUL ISLAM

6.4 Numbers

- Numbers are used quite often in python
  - Integers: We can add (+), multiply (*), exponent (**), and divide (/) integers in Python

```
print(2+3)
```

5

```
print(3-2)
```

1

```
print(2*3)
```

```
6
```

```python
print(3**3)
```

```
27
```

```python
print(3/2)
```

```
1.5
```

  - Python supports the order of operation too.

```python
print(2 + 3*4)
```

```
14
```

```python
print((2+3)*4)
```

```
20
```

  - Floats: Python calls any number with a decimal point a *float*

```python
print(2*0.2)
```

```
0.4
```

  - Integers and floats: When we divide two numbers, even if they are integers, you will always get a float

```python
print(4/2)
```

```
2.0
```

- Underscore in numbers
  - When you are writing long numbers, you can group digits using underscores

```python
universe_age = 14_000_000_000
print(universe_age)
```

14000000000

- Multiple assignments
  - We can assign the value to more than one variables using just a single line

```
x, y, z = 12, 3, 5
print(x, y, z)
print(z)
```

```
12 3 5
```

```
5
```

- Constants
  - A *constant* is a variable whose value stay the same throughout the life of a program.
  - Python does not have built in command for constant
  - However, a variable name with all capital letters treated as constant

```
ELON_MASK  = 1_000_000_000_000
print(ELON_MASK)
```

```
1000000000000
```

6.5 Comments

- In Python, hash (#) indicates a comment.
- Anything following a # mark in your code is ignored by Python

```
# Say hello to your friends
print("Hello friends")
```

```
Hello friends
```

6.6 Lists

- List the elements of a variable

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles)
```

```
['trek', 'cannondale', 'redline', 'specialized']
```

- Access an element from a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0])
```

```
trek
```

6.6.1 Neat outcome

- You can format the element "trek" even more neatly by using title() method

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0].title())
```

```
Trek
```

6.6.2 Print string

- You can print the strings that you are interested
- It starts counting from zero

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[1])
```

```
cannondale
```

- Python has special syntax to call the last item of a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[-1])
```

```
specialized
```

6.7 Functions

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
message = f"my first bicycle was a {bicycles[0].title()}"
print(message)
```

my first bicycle was a Trek

- Modifying elements in a list

7 Introduction to Python

7.1 Prerequisites

- There are three things we need to run the code in this lecture:
  - Python 3.10 (https://www.python.org/downloads/)
  - PyCharm community or professional (https://www.jetbrains.com/pycharm/)
  - A handful of other packages

7.2 Variables

- Let's start using a variable
- Variable names can contain only letters, numbers and underscores
- Space are not allowed in variable name
- Variables name should be short and descriptive

```
message = "Hello world!"
print(message)
```

```
Hello world!
```

7.3 Strings

- A *string* is a series of characters
- In python, anything inside a quotes is considered a string
- We can use single or double quotes around the strings

```
message = "Python is a programming language"
print(message)
```

```
Python is a programming language
```

```python
message = 'I told my friend, "Python is a programming language"'
print(message)
```

I told my friend, “Python is a programming language”

7.3.1 Change case of string

- A lower case string can be changed to a title

```python
name = "moinul islam"
print(name.title())
```

Moinul Islam

- We can also change the string to all upper or all lower

```python
name = "moinul islam"
print(name.upper())
```

MOINUL ISLAM

7.4 Numbers

- Numbers are used quite often in python
  - Integers: We can add (+), multiply (*), exponent (**), and divide (/) integers in Python

```python
print(2+3)
```

5

```python
print(3-2)
```

1

```python
print(2*3)
```

    6

```python
print(3**3)
```

    27

```python
print(3/2)
```

    1.5

  - Python supports the order of operation too.

```python
print(2 + 3*4)
```

    14

```python
print((2+3)*4)
```

    20

  - Floats: Python calls any number with a decimal point a *float*

```python
print(2*0.2)
```

    0.4

  - Integers and floats: When we divide two numbers, even if they are integers, you will always get a float

```python
print(4/2)
```

    2.0

- Underscore in numbers
  - When you are writing long numbers, you can group digits using underscores

```python
universe_age = 14_000_000_000
print(universe_age)
```

    14000000000

- Multiple assignments
  - We can assign the value to more than one variables using just a single line

```
x, y, z = 12, 3, 5
print(x, y, z)
print(z)
```

    12 3 5

    5

- Constants
  - A *constant* is a variable whose value stay the same throughout the life of a program.
  - Python does not have built in command for constant
  - However, a variable name with all capital letters treated as constant

```
ELON_MASK  = 1_000_000_000_000
print(ELON_MASK)
```

1000000000000

7.5 Comments

- In Python, hash (#) indicates a comment.
- Anything following a # mark in your code is ignored by Python

```
# Say hello to your friends
print("Hello friends")
```

Hello friends

7.6 Lists

- List the elements of a variable

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles)
```

```
['trek', 'cannondale', 'redline', 'specialized']
```

- Access an element from a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0])
```

```
trek
```

7.6.1 Neat outcome

- You can format the element "trek" even more neatly by using title() method

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0].title())
```

```
Trek
```

7.6.2 Print string

- You can print the strings that you are interested
- It starts counting from zero

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[1])
```

```
cannondale
```

- Python has special syntax to call the last item of a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[-1])
```

```
specialized
```

7.7 Functions

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
message = f"my first bicycle was a {bicycles[0].title()}"
print(message)
```

```
my first bicycle was a Trek
```

- Modifying elements in a list

8 Introduction to Python

8.1 Prerequisites

- There are three things we need to run the code in this lecture:
  - Python 3.10 (https://www.python.org/downloads/)
  - PyCharm community or professional (https://www.jetbrains.com/pycharm/)
  - A handful of other packages

8.2 Variables

- Let's start using a variable
- Variable names can contain only letters, numbers and underscores
- Space are not allowed in variable name
- Variables name should be short and descriptive

```
message = "Hello world!"
print(message)
```

```
Hello world!
```

8.3 Strings

- A *string* is a series of characters
- In python, anything inside a quotes is considered a string
- We can use single or double quotes around the strings

```
message = "Python is a programming language"
print(message)
```

```
Python is a programming language
```

```python
message = 'I told my friend, "Python is a programming language"'
print(message)
```

I told my friend, “Python is a programming language”

8.3.1 Change case of string

- A lower case string can be changed to a title

```python
name = "moinul islam"
print(name.title())
```

Moinul Islam

- We can also change the string to all upper or all lower

```python
name = "moinul islam"
print(name.upper())
```

MOINUL ISLAM

8.4 Numbers

- Numbers are used quite often in python
  - Integers: We can add (+), multiply (*), exponent (**), and divide (/) integers in Python

    ```python
    print(2+3)
    ```

    5

    ```python
    print(3-2)
    ```

    1

    ```python
    print(2*3)
    ```

```
6
```

```python
print(3**3)
```

```
27
```

```python
print(3/2)
```

```
1.5
```

- Python supports the order of operation too.

```python
print(2 + 3*4)
```

```
14
```

```python
print((2+3)*4)
```

```
20
```

- Floats: Python calls any number with a decimal point a *float*

```python
print(2*0.2)
```

```
0.4
```

- Integers and floats: When we divide two numbers, even if they are integers, you will always get a float

```python
print(4/2)
```

```
2.0
```

- Underscore in numbers
  - When you are writing long numbers, you can group digits using underscores

```python
universe_age = 14_000_000_000
print(universe_age)
```

14000000000

- Multiple assignments
  - We can assign the value to more than one variables using just a single line

```
x, y, z = 12, 3, 5
print(x, y, z)
print(z)
```

```
12 3 5

5
```

- Constants
  - A *constant* is a variable whose value stay the same throughout the life of a program.
  - Python does not have built in command for constant
  - However, a variable name with all capital letters treated as constant

```
ELON_MASK  = 1_000_000_000_000
print(ELON_MASK)
```

```
1000000000000
```

8.5 Comments

- In Python, hash (#) indicates a comment.
- Anything following a # mark in your code is ignored by Python

```
# Say hello to your friends
print("Hello friends")
```

```
Hello friends
```

8.6 Lists

- List the elements of a variable

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles)
```

    ['trek', 'cannondale', 'redline', 'specialized']

- Access an element from a list

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0])
```

    trek

8.6.1 Neat outcome

- You can format the element "trek" even more neatly by using title() method

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0].title())
```

    Trek

8.6.2 Print string

- You can print the strings that you are interested
- It starts counting from zero

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[1])
```

    cannondale

- Python has special syntax to call the last item of a list

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[-1])
```

    specialized

8.7 Functions

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
message = f"my first bicycle was a {bicycles[0].title()}"
print(message)
```

```
my first bicycle was a Trek
```

- Modifying elements in a list

9 Introduction to Python

9.1 Prerequisites

- There are three things we need to run the code in this lecture:
  - Python 3.10 (https://www.python.org/downloads/)
  - PyCharm community or professional (https://www.jetbrains.com/pycharm/)
  - A handful of other packages

9.2 Variables

- Let's start using a variable
- Variable names can contain only letters, numbers and underscores
- Space are not allowed in variable name
- Variables name should be short and descriptive

```
message = "Hello world!"
print(message)
```

```
Hello world!
```

9.3 Strings

- A *string* is a series of characters
- In python, anything inside a quotes is considered a string
- We can use single or double quotes around the strings

```
message = "Python is a programming language"
print(message)
```

```
Python is a programming language
```

```python
message = 'I told my friend, "Python is a programming language"'
print(message)
```

I told my friend, "Python is a programming language"

9.3.1 Change case of string

- A lower case string can be changed to a title

```python
name = "moinul islam"
print(name.title())
```

Moinul Islam

- We can also change the string to all upper or all lower

```python
name = "moinul islam"
print(name.upper())
```

MOINUL ISLAM

9.4 Numbers

- Numbers are used quite often in python
  - Integers: We can add (+), multiply (*), exponent (**), and divide (/) integers in Python

```python
print(2+3)
```

5

```python
print(3-2)
```

1

```python
print(2*3)
```

6

```
print(3**3)
```

27

```
print(3/2)
```

1.5

- Python supports the order of operation too.

```
print(2 + 3*4)
```

14

```
print((2+3)*4)
```

20

- Floats: Python calls any number with a decimal point a *float*

```
print(2*0.2)
```

0.4

- Integers and floats: When we divide two numbers, even if they are integers, you will always get a float

```
print(4/2)
```

2.0

- Underscore in numbers
  - When you are writing long numbers, you can group digits using underscores

```
universe_age = 14_000_000_000
print(universe_age)
```

```
14000000000
```

- Multiple assignments
  - We can assign the value to more than one variables using just a single line

```python
x, y, z = 12, 3, 5
print(x, y, z)
print(z)
```

```
12 3 5

5
```

- Constants
  - A *constant* is a variable whose value stay the same throughout the life of a program.
  - Python does not have built in command for constant
  - However, a variable name with all capital letters treated as constant

```python
ELON_MASK  = 1_000_000_000_000
print(ELON_MASK)
```

```
1000000000000
```

9.5 Comments

- In Python, hash (#) indicates a comment.
- Anything following a # mark in your code is ignored by Python

```python
# Say hello to your friends
print("Hello friends")
```

```
Hello friends
```

9.6 Lists

- List the elements of a variable

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles)
```

['trek', 'cannondale', 'redline', 'specialized']

- Access an element from a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0])
```

trek

9.6.1 Neat outcome

- You can format the element "trek" even more neatly by using title() method

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0].title())
```

Trek

9.6.2 Print string

- You can print the strings that you are interested
- It starts counting from zero

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[1])
```

cannondale

- Python has special syntax to call the last item of a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[-1])
```

specialized

9.7 Functions

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
message = f"my first bicycle was a {bicycles[0].title()}"
print(message)
```

```
my first bicycle was a Trek
```

- Modifying elements in a list

10 Introduction to Python

10.1 Prerequisites

- There are three things we need to run the code in this lecture:
  - Python 3.10 (https://www.python.org/downloads/)
  - PyCharm community or professional (https://www.jetbrains.com/pycharm/)
  - A handful of other packages

10.2 Variables

- Let's start using a variable
- Variable names can contain only letters, numbers and underscores
- Space are not allowed in variable name
- Variables name should be short and descriptive

```
message = "Hello world!"
print(message)
```

```
Hello world!
```

10.3 Strings

- A *string* is a series of characters
- In python, anything inside a quotes is considered a string
- We can use single or double quotes around the strings

```
message = "Python is a programming language"
print(message)
```

```
Python is a programming language
```

```
message = 'I told my friend, "Python is a programming language"'
print(message)
```

I told my friend, “Python is a programming language”

10.3.1 Change case of string

- A lower case string can be changed to a title

```
name = "moinul islam"
print(name.title())
```

Moinul Islam

- We can also change the string to all upper or all lower

```
name = "moinul islam"
print(name.upper())
```

MOINUL ISLAM

10.4 Numbers

- Numbers are used quite often in python
  - Integers: We can add (+), multiply (*), exponent (**), and divide (/) integers in Python

```
print(2+3)
```

5

```
print(3-2)
```

1

```
print(2*3)
```

```
6
```

```python
print(3**3)
```

```
27
```

```python
print(3/2)
```

```
1.5
```

- Python supports the order of operation too.

```python
print(2 + 3*4)
```

```
14
```

```python
print((2+3)*4)
```

```
20
```

- Floats: Python calls any number with a decimal point a *float*

```python
print(2*0.2)
```

```
0.4
```

- Integers and floats: When we divide two numbers, even if they are integers, you will always get a float

```python
print(4/2)
```

```
2.0
```

- Underscore in numbers
  - When you are writing long numbers, you can group digits using underscores

```python
universe_age = 14_000_000_000
print(universe_age)
```

```
14000000000
```

- Multiple assignments
  - We can assign the value to more than one variables using just a single line

```python
x, y, z = 12, 3, 5
print(x, y, z)
print(z)
```

```
12 3 5
```

```
5
```

- Constants
  - A *constant* is a variable whose value stay the same throughout the life of a program.
  - Python does not have built in command for constant
  - However, a variable name with all capital letters treated as constant

```python
ELON_MASK  = 1_000_000_000_000
print(ELON_MASK)
```

```
1000000000000
```

10.5 Comments

- In Python, hash (#) indicates a comment.
- Anything following a # mark in your code is ignored by Python

```python
# Say hello to your friends
print("Hello friends")
```

```
Hello friends
```

10.6 Lists

- List the elements of a variable

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles)
```

```
['trek', 'cannondale', 'redline', 'specialized']
```

- Access an element from a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0])
```

```
trek
```

10.6.1 Neat outcome

- You can format the element "trek" even more neatly by using title() method

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0].title())
```

```
Trek
```

10.6.2 Print string

- You can print the strings that you are interested
- It starts counting from zero

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[1])
```

```
cannondale
```

- Python has special syntax to call the last item of a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[-1])
```

```
specialized
```

10.7 Functions

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
message = f"my first bicycle was a {bicycles[0].title()}"
print(message)
```

```
my first bicycle was a Trek
```

- Modifying elements in a list

11 Introduction to Python

11.1 Prerequisites

- There are three things we need to run the code in this lecture:
  - Python 3.10 (https://www.python.org/downloads/)
  - PyCharm community or professional (https://www.jetbrains.com/pycharm/)
  - A handful of other packages

11.2 Variables

- Let's start using a variable
- Variable names can contain only letters, numbers and underscores
- Space are not allowed in variable name
- Variables name should be short and descriptive

```
message = "Hello world!"
print(message)
```

```
Hello world!
```

11.3 Strings

- A *string* is a series of characters
- In python, anything inside a quotes is considered a string
- We can use single or double quotes around the strings

```
message = "Python is a programming language"
print(message)
```

```
Python is a programming language
```

```
message = 'I told my friend, "Python is a programming language"'
print(message)
```

I told my friend, “Python is a programming language”

11.3.1 Change case of string

- A lower case string can be changed to a title

```
name = "moinul islam"
print(name.title())
```

Moinul Islam

- We can also change the string to all upper or all lower

```
name = "moinul islam"
print(name.upper())
```

MOINUL ISLAM

11.4 Numbers

- Numbers are used quite often in python
  - Integers: We can add (+), multiply (*), exponent (**), and divide (/) integers in Python

```
print(2+3)
```

5

```
print(3-2)
```

1

```
print(2*3)
```

    6

```
print(3**3)
```

    27

```
print(3/2)
```

    1.5

  - Python supports the order of operation too.

```
print(2 + 3*4)
```

    14

```
print((2+3)*4)
```

    20

  - Floats: Python calls any number with a decimal point a *float*

```
print(2*0.2)
```

    0.4

  - Integers and floats: When we divide two numbers, even if they are integers, you will always get a float

```
print(4/2)
```

    2.0

- Underscore in numbers
  - When you are writing long numbers, you can group digits using underscores

```
universe_age = 14_000_000_000
print(universe_age)
```

```
14000000000
```

- Multiple assignments
  - We can assign the value to more than one variables using just a single line

```
x, y, z = 12, 3, 5
print(x, y, z)
print(z)
```

```
12 3 5

5
```

- Constants
  - A *constant* is a variable whose value stay the same throughout the life of a program.
  - Python does not have built in command for constant
  - However, a variable name with all capital letters treated as constant

```
ELON_MASK  = 1_000_000_000_000
print(ELON_MASK)
```

```
1000000000000
```

11.5 Comments

- In Python, hash (#) indicates a comment.
- Anything following a # mark in your code is ignored by Python

```
# Say hello to your friends
print("Hello friends")
```

```
Hello friends
```

11.6 Lists

- List the elements of a variable

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles)
```

['trek', 'cannondale', 'redline', 'specialized']

- Access an element from a list

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0])
```

trek

11.6.1 Neat outcome

- You can format the element "trek" even more neatly by using title() method

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0].title())
```

Trek

11.6.2 Print string

- You can print the strings that you are interested
- It starts counting from zero

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[1])
```

cannondale

- Python has special syntax to call the last item of a list

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[-1])
```

specialized

11.7 Functions

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
message = f"my first bicycle was a {bicycles[0].title()}"
print(message)
```

```
my first bicycle was a Trek
```

- Modifying elements in a list

12 Introduction to Python

12.1 Prerequisites

- There are three things we need to run the code in this lecture:
  - Python 3.10 (https://www.python.org/downloads/)
  - PyCharm community or professional (https://www.jetbrains.com/pycharm/)
  - A handful of other packages

12.2 Variables

- Let's start using a variable
- Variable names can contain only letters, numbers and underscores
- Space are not allowed in variable name
- Variables name should be short and descriptive

```
message = "Hello world!"
print(message)
```

```
Hello world!
```

12.3 Strings

- A *string* is a series of characters
- In python, anything inside a quotes is considered a string
- We can use single or double quotes around the strings

```
message = "Python is a programming language"
print(message)
```

```
Python is a programming language
```

```python
message = 'I told my friend, "Python is a programming language"'
print(message)
```

I told my friend, "Python is a programming language"

12.3.1 Change case of string

- A lower case string can be changed to a title

```python
name = "moinul islam"
print(name.title())
```

Moinul Islam

- We can also change the string to all upper or all lower

```python
name = "moinul islam"
print(name.upper())
```

MOINUL ISLAM

12.4 Numbers

- Numbers are used quite often in python
  - Integers: We can add (+), multiply (*), exponent (**), and divide (/) integers in Python

```python
print(2+3)
```

5

```python
print(3-2)
```

1

```python
print(2*3)
```

6

```python
print(3**3)
```

27

```python
print(3/2)
```

1.5

  - Python supports the order of operation too.

```python
print(2 + 3*4)
```

14

```python
print((2+3)*4)
```

20

  - Floats: Python calls any number with a decimal point a *float*

```python
print(2*0.2)
```

0.4

  - Integers and floats: When we divide two numbers, even if they are integers, you will always get a float

```python
print(4/2)
```

2.0

- Underscore in numbers
  - When you are writing long numbers, you can group digits using underscores

```python
universe_age = 14_000_000_000
print(universe_age)
```

14000000000

- Multiple assignments
  - We can assign the value to more than one variables using just a single line

```python
x, y, z = 12, 3, 5
print(x, y, z)
print(z)
```

```
12 3 5

5
```

- Constants
  - A *constant* is a variable whose value stay the same throughout the life of a program.
  - Python does not have built in command for constant
  - However, a variable name with all capital letters treated as constant

```python
ELON_MASK  = 1_000_000_000_000
print(ELON_MASK)
```

```
1000000000000
```

12.5 Comments

- In Python, hash (#) indicates a comment.
- Anything following a # mark in your code is ignored by Python

```python
# Say hello to your friends
print("Hello friends")
```

```
Hello friends
```

12.6 Lists

- List the elements of a variable

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles)
```

['trek', 'cannondale', 'redline', 'specialized']

- Access an element from a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0])
```

trek

12.6.1 Neat outcome

- You can format the element "trek" even more neatly by using title() method

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0].title())
```

Trek

12.6.2 Print string

- You can print the strings that you are interested
- It starts counting from zero

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[1])
```

cannondale

- Python has special syntax to call the last item of a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[-1])
```

specialized

12.7 Functions

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
message = f"my first bicycle was a {bicycles[0].title()}"
print(message)
```

```
my first bicycle was a Trek
```

- Modifying elements in a list

13 Introduction to Python

13.1 Prerequisites

- There are three things we need to run the code in this lecture:
  - Python 3.10 (https://www.python.org/downloads/)
  - PyCharm community or professional (https://www.jetbrains.com/pycharm/)
  - A handful of other packages

13.2 Variables

- Let's start using a variable
- Variable names can contain only letters, numbers and underscores
- Space are not allowed in variable name
- Variables name should be short and descriptive

```python
message = "Hello world!"
print(message)
```

```
Hello world!
```

13.3 Strings

- A *string* is a series of characters
- In python, anything inside a quotes is considered a string
- We can use single or double quotes around the strings

```python
message = "Python is a programming language"
print(message)
```

```
Python is a programming language
```

```
message = 'I told my friend, "Python is a programming language"'
print(message)
```

I told my friend, "Python is a programming language"

13.3.1 Change case of string

- A lower case string can be changed to a title

```
name = "moinul islam"
print(name.title())
```

Moinul Islam

- We can also change the string to all upper or all lower

```
name = "moinul islam"
print(name.upper())
```

MOINUL ISLAM

13.4 Numbers

- Numbers are used quite often in python
  - Integers: We can add (+), multiply (*), exponent (**), and divide (/) integers in Python

```
print(2+3)
```

5

```
print(3-2)
```

1

```
print(2*3)
```

6

```
print(3**3)
```

27

```
print(3/2)
```

1.5

- Python supports the order of operation too.

```
print(2 + 3*4)
```

14

```
print((2+3)*4)
```

20

- Floats: Python calls any number with a decimal point a *float*

```
print(2*0.2)
```

0.4

- Integers and floats: When we divide two numbers, even if they are integers, you will always get a float

```
print(4/2)
```

2.0

- Underscore in numbers
  - When you are writing long numbers, you can group digits using underscores

```
universe_age = 14_000_000_000
print(universe_age)
```

```
14000000000
```

- Multiple assignments
    - We can assign the value to more than one variables using just a single line

```
x, y, z = 12, 3, 5
print(x, y, z)
print(z)
```

```
12 3 5

5
```

- Constants
    - A *constant* is a variable whose value stay the same throughout the life of a program.
    - Python does not have built in command for constant
    - However, a variable name with all capital letters treated as constant

```
ELON_MASK  = 1_000_000_000_000
print(ELON_MASK)
```

```
1000000000000
```

13.5 Comments

- In Python, hash (#) indicates a comment.
- Anything following a # mark in your code is ignored by Python

```
# Say hello to your friends
print("Hello friends")
```

```
Hello friends
```

13.6 Lists

- List the elements of a variable

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles)
```

    ['trek', 'cannondale', 'redline', 'specialized']

- Access an element from a list

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0])
```

    trek

13.6.1 Neat outcome

- You can format the element "trek" even more neatly by using title() method

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0].title())
```

    Trek

13.6.2 Print string

- You can print the strings that you are interested
- It starts counting from zero

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[1])
```

    cannondale

- Python has special syntax to call the last item of a list

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[-1])
```

    specialized

13.7 Functions

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
message = f"my first bicycle was a {bicycles[0].title()}"
print(message)
```

```
my first bicycle was a Trek
```

- Modifying elements in a list

14 Introduction to Python

14.1 Prerequisites

- There are three things we need to run the code in this lecture:
  - Python 3.10 (https://www.python.org/downloads/)
  - PyCharm community or professional (https://www.jetbrains.com/pycharm/)
  - A handful of other packages

14.2 Variables

- Let's start using a variable
- Variable names can contain only letters, numbers and underscores
- Space are not allowed in variable name
- Variables name should be short and descriptive

```
message = "Hello world!"
print(message)
```

```
Hello world!
```

14.3 Strings

- A *string* is a series of characters
- In python, anything inside a quotes is considered a string
- We can use single or double quotes around the strings

```
message = "Python is a programming language"
print(message)
```

```
Python is a programming language
```

```python
message = 'I told my friend, "Python is a programming language"'
print(message)
```

```
I told my friend, "Python is a programming language"
```

14.3.1 Change case of string

- A lower case string can be changed to a title

```python
name = "moinul islam"
print(name.title())
```

```
Moinul Islam
```

- We can also change the string to all upper or all lower

```python
name = "moinul islam"
print(name.upper())
```

```
MOINUL ISLAM
```

14.4 Numbers

- Numbers are used quite often in python
  - Integers: We can add (+), multiply (*), exponent (**), and divide (/) integers in Python

```python
print(2+3)
```

```
5
```

```python
print(3-2)
```

```
1
```

```python
print(2*3)
```

```
6
```

```python
print(3**3)
```

```
27
```

```python
print(3/2)
```

```
1.5
```

  - Python supports the order of operation too.

```python
print(2 + 3*4)
```

```
14
```

```python
print((2+3)*4)
```

```
20
```

  - Floats: Python calls any number with a decimal point a *float*

```python
print(2*0.2)
```

```
0.4
```

  - Integers and floats: When we divide two numbers, even if they are integers, you will always get a float

```python
print(4/2)
```

```
2.0
```

- Underscore in numbers
  - When you are writing long numbers, you can group digits using underscores

```python
universe_age = 14_000_000_000
print(universe_age)
```

```
14000000000
```

- Multiple assignments
  - We can assign the value to more than one variables using just a single line

```python
x, y, z = 12, 3, 5
print(x, y, z)
print(z)
```

```
12 3 5

5
```

- Constants
  - A *constant* is a variable whose value stay the same throughout the life of a program.
  - Python does not have built in command for constant
  - However, a variable name with all capital letters treated as constant

```python
ELON_MASK  = 1_000_000_000_000
print(ELON_MASK)
```

```
1000000000000
```

14.5 Comments

- In Python, hash (#) indicates a comment.
- Anything following a # mark in your code is ignored by Python

```python
# Say hello to your friends
print("Hello friends")
```

```
Hello friends
```

14.6 Lists

- List the elements of a variable

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles)
```

['trek', 'cannondale', 'redline', 'specialized']

- Access an element from a list

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0])
```

trek

14.6.1 Neat outcome

- You can format the element "trek" even more neatly by using title() method

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0].title())
```

Trek

14.6.2 Print string

- You can print the strings that you are interested
- It starts counting from zero

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[1])
```

cannondale

- Python has special syntax to call the last item of a list

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[-1])
```

specialized

14.7 Functions

```
bicycles = ["trek", "cannondale", "redline", "specialized"]
message = f"my first bicycle was a {bicycles[0].title()}"
print(message)
```

```
my first bicycle was a Trek
```

- Modifying elements in a list

15 Introduction to Python

15.1 Prerequisites

- There are three things we need to run the code in this lecture:
  - Python 3.10 (https://www.python.org/downloads/)
  - PyCharm community or professional (https://www.jetbrains.com/pycharm/)
  - A handful of other packages

15.2 Variables

- Let's start using a variable
- Variable names can contain only letters, numbers and underscores
- Space are not allowed in variable name
- Variables name should be short and descriptive

```
message = "Hello world!"
print(message)
```

```
Hello world!
```

15.3 Strings

- A *string* is a series of characters
- In python, anything inside a quotes is considered a string
- We can use single or double quotes around the strings

```
message = "Python is a programming language"
print(message)
```

```
Python is a programming language
```

```
message = 'I told my friend, "Python is a programming language"'
print(message)
```

```
I told my friend, “Python is a programming language”
```

15.3.1 Change case of string

- A lower case string can be changed to a title

```
name = "moinul islam"
print(name.title())
```

```
Moinul Islam
```

- We can also change the string to all upper or all lower

```
name = "moinul islam"
print(name.upper())
```

```
MOINUL ISLAM
```

15.4 Numbers

- Numbers are used quite often in python
  - Integers: We can add (+), multiply (*), exponent (**), and divide (/) integers in Python

  ```
  print(2+3)
  ```

  ```
  5
  ```

  ```
  print(3-2)
  ```

  ```
  1
  ```

  ```
  print(2*3)
  ```

6

```
print(3**3)
```

27

```
print(3/2)
```

1.5

- Python supports the order of operation too.

```
print(2 + 3*4)
```

14

```
print((2+3)*4)
```

20

- Floats: Python calls any number with a decimal point a *float*

```
print(2*0.2)
```

0.4

- Integers and floats: When we divide two numbers, even if they are integers, you will always get a float

```
print(4/2)
```

2.0

- Underscore in numbers
  - When you are writing long numbers, you can group digits using underscores

```
universe_age = 14_000_000_000
print(universe_age)
```

14000000000

- Multiple assignments
  - We can assign the value to more than one variables using just a single line

```
x, y, z = 12, 3, 5
print(x, y, z)
print(z)
```

```
12 3 5
```

```
5
```

- Constants
  - A *constant* is a variable whose value stay the same throughout the life of a program.
  - Python does not have built in command for constant
  - However, a variable name with all capital letters treated as constant

```
ELON_MASK  = 1_000_000_000_000
print(ELON_MASK)
```

```
1000000000000
```

15.5 Comments

- In Python, hash (#) indicates a comment.
- Anything following a # mark in your code is ignored by Python

```
# Say hello to your friends
print("Hello friends")
```

```
Hello friends
```

15.6 Lists

- List the elements of a variable

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles)
```

['trek', 'cannondale', 'redline', 'specialized']

- Access an element from a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0])
```

trek

15.6.1 Neat outcome

- You can format the element "trek" even more neatly by using title() method

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[0].title())
```

Trek

15.6.2 Print string

- You can print the strings that you are interested
- It starts counting from zero

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[1])
```

cannondale

- Python has special syntax to call the last item of a list

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
print(bicycles[-1])
```

specialized

15.7 Functions

```python
bicycles = ["trek", "cannondale", "redline", "specialized"]
message = f"my first bicycle was a {bicycles[0].title()}"
print(message)
```

```
my first bicycle was a Trek
```

- Modifying elements in a list

References