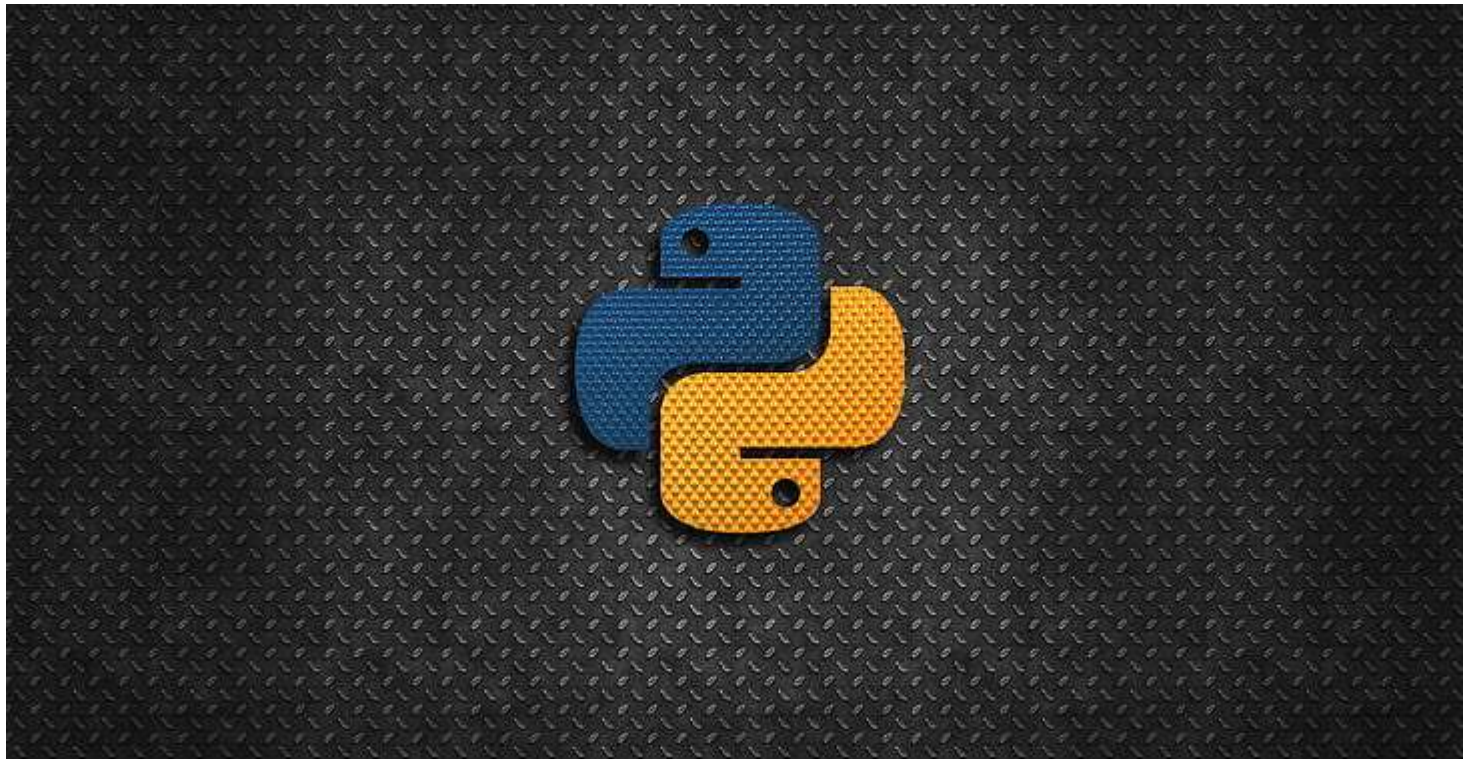

PYTHON

BASICS

Intro and adv

- Python is a popular programming language. It was created by Guido van Rossum, and released in 1991.
- Python can be used on a server to create web applications.
- Python can connect to database systems. It can also read and modify files.

Advantages

- Simple and easy to learn syntax
- Extensive libraries and frameworks
- Open source and free to use
- Large, active community support
- Versatile—used in web, data science, AI, automation, and more
- Python Syntax-print("Hello, World!")

  Hello, World!

Python Variables

- Variables are containers for storing data values.
- A variable can have a short name (like x and y) or a more descriptive name (age, carname, total_volume).
- Rules for Python variables
- A variable name must start with a letter or the underscore character
- A variable name cannot start with a number
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- Variable names are case-sensitive (age, Age and AGE are three different variables)
- A variable name cannot be any of the Python keywords

- Python Variables - Assign Multiple Values
- Exmpl- x, y, z = "Orange", "Banana", "Cherry"
  print(x)
  print(y)
  print(z)
- Python - Output Variables
- Exmple-x = "Python is awesome"
  print(x)

Datatypes

Text Type:	`str`
Numeric Types:	`int`, `float`, `complex`
Sequence Types:	`list`, `tuple`,
Mapping Type:	`dict`
Set Types:	`set`,
Boolean Type:	`bool`

Strings

- Strings in python are surrounded by either single quotation marks, or double quotation marks.
- Examples:
- print('Hello')
- print("He is called 'Johnny’”)
- a = """Lorem ipsum dolor sit amet,
  consectetur adipiscing elit,
  sed do eiusmod tempor incididunt
  ut labore et dolore magna aliqua."""
  print(a)

String- Slicing

- Slice From the Start, (start number 0 and doesn’t include last number example:
  - b = "Hello, World!"
    print(b[:5])----------→ ‘Hello’
- Slice To the End(Get the characters from position , and all the way to the end: example:
  - b = "Hello, World!"
    print(b[2:])----------→llo, World!
-

Modify string

- Uppercase:
  - Eg:
  - a = "Hello, World!"
  - print(a.upper()) ----------→HELLO, WORLD!
- Lower Case
  - Eg:
  - a = "Hello, World!"
  - print(a.lower())----------→hello, world!
- Replace String
  - a = "Hello, World!"
  - print(a.replace("H", "J"))----→Jello, World!

- Split String
  - Eg:
  - a = "Hello, World!"
  - b = a.split(",")
  - print(b)--------→['Hello', ' World!’]
- String Concatenation
  - Eg:
  - a = "Hello"
  - b = "World"
  - c = a + b
  - print(c)----------→HelloWorld

Lists

- Lists are used to store multiple items in a single variable.
- built-in data types in Python used to store collections of data
- List items are ordered, changeable, and allow duplicate values
- Lists are created using square brackets:
  - Eg:
  - thislist = ["apple", "banana", "cherry"]
  - print(thislist)----------→['apple', 'banana', 'cherry']

- Access List Items
- List items are indexed and you can access them by referring to the index number:
  - EG:
  - thislist = ["apple", "banana", "cherry"]
  - print(thislist[1])----------→banana
- Range of Indexes
- You can specify a range of indexes by specifying where to start and where to end the range.
  - Eg:
  - thislist = ["apple", "banana", "cherry", "orange", "kiwi", "melon", "mango"]
  - print(thislist[2:5])----------→ ['cherry', 'orange', 'kiwi']

- Append Items
  - Eg:
  - thislist = ["apple", "banana", "cherry"]

    thislist.append("orange")

    print(thislist)--------→['apple', 'banana', 'cherry', 'orange’]

  Insert Items

  Eg:

  thislist = ["apple", "banana", "cherry"]
  thislist.insert(1, "orange")
  print(thislist)---------→['apple', 'orange', 'banana', 'cherry']

- Remove List Items
  - Eg:
  - thislist = ["apple", "banana", "cherry"]
    thislist.remove("banana")
    print(thislist)----------→['apple', 'cherry', 'banana', 'kiwi’]
- The pop() method removes the specified index.
  - Eg:
  - thislist = ["apple", "banana", "cherry"]
    thislist.pop(1)
    print(thislist)----------→['apple', 'cherry']

- Join Two Lists
  - Eg:

```
list1 = ["a", "b", "c"]
list2 = [1, 2, 3]
list3 = list1 + list2
print(list3)--------→['a', 'b', 'c', 1, 2, 3]
```

Tuple

- Tuple is one of 4 built-in data types in Python used to store collections of data
- A tuple is a collection which is ordered and unchangeable.
- Tuples are written with round brackets.
- Tuples allow duplicate values
  - Eg:
  - thistuple = ("apple", "banana", "cherry")
  - print(thistuple)---------→('apple', 'banana', 'cherry')

Change Tuple Values

- convert the tuple into a list, change the list, and convert the list back into a tuple
  - Eg:

```
x = ("apple", "banana", "cherry")
y = list(x)
y[1] = "kiwi"
x = tuple(y)
print(x)----------------→("apple", "kiwi", "cherry")
```

- Join Two Tuples
- To join two or more tuples you can use the + operator:
  - Eg:

```
tuple1 = ("a", "b" , "c")
tuple2 = (1, 2, 3)
tuple3 = tuple1 + tuple2
print(tuple3)----------→('a', 'b', 'c', 1, 2, 3)
```

Set

- built-in data types in Python used to store collections of data
- A set is a collection which is *unordered, unchangeable,* and *unindexed*.
  - (*items* are unchangeable, but you can remove and add new items.)

  Eg:

  thisset = {"apple", "banana", "cherry"}

  print(thisset)----------→{'apple', 'cherry', 'banana'}

- Access Set Items
- You cannot access items in a set by referring to an index or a key.
- But you can loop through the set items using a for loop,

  Eg:

```
thisset = {"apple", "banana", "cherry"}
for x in thisset:
  print(x)----------→        apple
                            banana
                            cherry
```

- or ask if a specified value is present in a set, by using the in keyword.

  Eg:

```
thisset = {"apple", "banana", "cherry"}
print("banana" in thisset)----------→ True
```

- Add Items:
  - Eg:

  thisset = {"apple", "banana", "cherry"}

  thisset.add("orange")

  print(thisset)----------→{'orange', 'cherry', 'apple', 'banana’}
- Remove Item
  - remove(), or the discard()

- Loop Items
- You can loop through the set items by using a for loop:
  - Eg:

```
thisset = {"apple", "banana", "cherry"}
for x in thisset:
  print(x)----------→      banana
                          apple
                          cherry
```

Dictionary

- ordered, changeable and do not allow duplicates

Eg:

```
thisdict =    {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
print(thisdict)----------→{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
```

- Accessing Items

```
thisdict =          {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
x = thisdict["model"]
print(x)----------→Mustang
```

- Also can use get()

```
thisdict =          {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
x = thisdict.get("model")
print(x)-------------------→Mustang
```

- Change Values

```
thisdict =    {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
thisdict["year"] = 2018
print(thisdict)----------→{'brand': 'Ford', 'model': 'Mustang', 'year': 2018}
```

If ... Else

```
a = 33
b = 200
if b > a:
    print("b is greater than a")----------→b is greater than a
        #If statement, without indentation (will raise an error)
```

Elif

- if the previous conditions were not true, then try this condition“
  - Eg:

```
a = 33
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")----------→a and b are equal
```

Else

- catches anything which isn't caught by the preceding conditions.
  - Eg:

```
a = 200
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
else:
  print("a is greater than b")
```

----------→a is greater than b

Python Loops

- Python has two primitive loop commands:
- while loops
- for loops

while loop :

we can execute a set of statements as long as a condition is true.

- Eg:

```
i = 1
while i < 6:
  print(i)
  i += 1----------------->   1
                             2
                             3
                             4
                             5
```

- continue statement :

we can stop the current iteration, and continue with the next:

Eg:

```
i = 0
while i < 6:
    i += 1
    if i == 3:
        continue
    print(i)----------→ 1
                        2
                        4
                        5
                        6
```

For Loops

- for loop :
- used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string).
- we can execute a set of statements, once for each item in a list, tuple, set etc
  - Eg:

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)----------------------------------------→   apple
                                                      banana
                                                      cherry
```

- break statement :
- we can stop the loop before it has looped through all the items:
  - Eg:

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
  if x == "banana":
    break----------------------------------→apple
                                                banana
```

  - Eg 2:

```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  if x == "banana":
    break
  print(x)--------------------------------→apple
```

Functions

- function is defined using the def keyword:
- Eg:def my_function():

  print("Hello from a function")

  my_function()----------→Hello from a function

Arguments:

- Information can be passed into functions as arguments.
  - Eg:

```
def my_function(fname):
  print(fname + " Refsnes")

my_function("Emil")
my_function("Tobias")
my_function("Linus")----------→Emil Refsnes
                                Tobias Refsnes
                                Linus Refsnes
```

User Input

- able to ask the user for input.
- The following example asks for your name, and when you enter a name, it gets printed on the screen:
  - print("Enter your name:")
  - name = input()
  - print(f"Hello {name}")----------→Enter your name: *alex(input)*
    Hello alex

- By Using Prompt:
  - Eg:
  - name = input("Enter your name:")
  - print(f"Hello {name}")----------→Enter your name: alex (input)
    Hello alex