Week 1

Python Programming

Python Syntax

1. **Hello World**

   #!/bin/python3

   print("Hello, World!")

   Ans: `Hello, World!`

2. **Comments**

   #This is a comment.

   print("Hello, World!")

   Ans: `Hello, World!`

3. **Docstrings**

   """This is a

   multiline docstring."""

   print("Hello, World!")

   Ans: `Hello, World!`

Python Variables

1. **Create variable**

   x = 5

   y = "John"

   print(x)

   print(y)

   Ans:
   `5`
   `John`

2. **Output both text and a variable**

```
x = "awesome"

print("Python is " + x)
```

Ans:
```
Python is awesome
```

3. **Add a variable to another variable**

```
x = "Python is "

y = "awesome"

z = x + y

print(z)
```

Ans:
```
Python is awesome
```

Python Numbers

1. **Verify the type of an object**

```
x = 1
y = 2.8
z = 1j
print(type(x))
print(type(y))
print(type(z))
```

Ans:

```
<class 'int'>
<class 'float'>
<class 'complex'>
```

2. **Create integers**

```
x = 1
y = 35656222554887711
z = -3255522
print(type(x))
print(type(y))
print(type(z))
```

Ans:
```
<class 'int'>
<class 'int'>
<class 'int'>
```

3. **Create floating point numbers**

```
x = 1.10
y = 1.0
z = -35.59
```

```
print(type(x))
print(type(y))
print(type(z))
```
Ans:
```
<class 'float'>
<class 'float'>
<class 'float'>
```

4. **Create scientific numbers with an “e” to indicate the power of 10**

```
x = 35e3
y = 12E4
z = -87.7e100
print(type(x))
print(type(y))
print(type(z))
```
Ans:

```
<class 'float'>
<class 'float'>
<class 'float'>
```

5. **Create complex numbers**

```
x = 3+5j
y = 5j
z = -5j
print(type(x))
print(type(y))
print(type(z))
```
Ans:

```
<class 'complex'>
<class 'complex'>
<class 'complex'>
```

Python Casting

1. **Casting Integer**

```
x = int(1)
y = int(2.8)
z = int("3")
print(x)
print(y)
print(z)
```
Ans:

```
1
2
3
```

2. **Casting Floats**

```
x = float(1)
y = float(2.8)
z = float("3")
w = float("4.2")
print(x)
print(y)
print(z)
print(w)
```

Ans:

```
1.0
2.8
3.0
4.2
```

3. **Casting Strings**

```
x = str("s1")
y = str(2)
z = str(3.0)
print(x)
print(y)
print(z)
```

Ans:

```
s1
2
3.0
```

Python Strings

1. **Get Character at position 1 of a string**

```
a = "Hello, World!"
print(a[1])
```

Ans: `e`

2. **Substring. Get the characters from position 2 to 5**

```
b = "Hello, World!"
print(b[2:5])
```

Ans: `llo`

3. **Remove whitespace from the beginning or at the end of a string**

   a = " Hello, World! "

   print(a.strip())

   Ans: `Hello, World!`

4. **Return the length of a string**

   a = "Hello, World!"

   print(len(a))

   Ans: `13`

5. **Convert a string of lower case**

   a = "Hello, World!"

   print(a.lower())

   Ans: `hello, world!`

6. **Convert a string to upper case**

   a = "Hello, World!"

   print(a.upper())

   Ans: `HELLO, WORLD!`

7. **Replace a character in string**

   a = "Hello, World!"

   print(a.replace("H", "J"))

   Ans: `Jello, World!`

8. **Split a string into substring**

   a = "Hello, World!"

   b = a.split(",")

   print(b)

   Ans: `['Hello', ' World!']`

Python Operators

1. **Addition operator**
   x = 5
   y = 3
   print(x + y)
   Ans: 8

2. **Subtraction operator**
   x = 5
   y = 3
   print(x - y)
   Ans: 2

3. **Multiplication operator**
   x = 5
   y = 3
   print(x * y)
   Ans: 15
4. **Division operator**
   x = 12
   y = 3
   print(x / y)
   Ans: 4
5. **Modulus operator**
   x = 5
   y = 2
   print(x % y)
   Ans: 1
6. **Assignment operator**
   x = 5
   print(x)
   Ans: 5

Python Lists

1. **Create a list**

   thislist = ["apple", "banana", "cherry"]

   print(thislist)

   Ans: `['apple', 'banana', 'cherry']`

2. **Access list items**

   thislist = ["apple", "banana", "cherry"]

   print(thislist[1])

Ans: `banana`

3. **Change the value of a list item**

```
thislist = ["apple", "banana", "cherry"]
thislist[1] = "blackcurrant"
print(thislist)
```

Ans: `['apple', 'blackcurrant', 'cherry']`

4. **Get the length of a list**

```
thislist = ["apple", "banana", "cherry"]
print(len(thislist))
```

Ans: `3`

5. **Add and item to the end of list**

```
thislist = ["apple", "banana", "cherry"]
thislist.append("orange")
print(thislist)
```

Ans: `['apple', 'banana', 'cherry', 'orange']`

6. **Add an item at specified index**

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

Ans: `['apple', 'orange', 'banana', 'cherry']`

7. **Remove an item**

```
thislist = ["apple", "banana", "cherry"]
thislist.remove("banana")
print(thislist)
```

Ans: `['apple', 'cherry']`

8. **Empty a list**

```
thislist = ["apple", "banana", "cherry"]
thislist.clear()
print(thislist)
```

Ans: `[]`

Python Tuples

1. **Create a tuple**

   thistuple = ("apple", "banana", "cherry")

   print(thistuple)

   Ans: `('apple', 'banana', 'cherry')`

2. **Access tuple items**

   thistuple = ("apple", "banana", "cherry")

   print(thistuple[1])

   Ans: `banana`

3. **Change tuple values**

   thistuple = ("apple", "banana", "cherry")

   thistuple[1] = "blackcurrant"

   # the value is still the same:

   print(thistuple)

   Ans: `('apple', 'banana', 'cherry')`

4. **Get the length of a tuple**

   thistuple = ("apple", "banana", "cherry")

   print(len(thistuple))

   Ans: `3`

5. **Delete a tuple**

   thistuple = ("apple", "banana", "cherry")

   del thistuple

   print(thistuple) #this will raise an error because the tuple no longer exists

   Ans:
   ```
   Traceback (most recent call last):
     File "demo_tuple_del.py", line 3, in <module>
       print(thistuple) #this will raise an error because the tuple
   no longer exists
   NameError: name 'thistuple' is not defined5
   ```

6. **Using the tuple() constructor to create a tuple**

   thistuple = tuple(("apple", "banana", "cherry"))

   print(thistuple)

   Ans: `('apple', 'banana', 'cherry')`

Python Sets

1. **Create a set**

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

   print(thisset)

   Ans: `{'cherry', 'banana', 'apple'}`

2. **Add an item to a set**

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

   thisset.add("orange")

   print(thisset)

   Ans: `{'orange', 'cherry', 'banana', 'apple'}`

3. **Add multiple items to a set**

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

   thisset.update(["orange", "mango", "grapes"])

   print(thisset)

   Ans: `{'orange', 'mango', 'cherry', 'grapes', 'banana', 'apple'}`

Python Dictionaries

1. **Create a directory**

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

Ans: `{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}`

2. **Access item from dictionary**

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

Ans: `Mustang`

3. **Change the value of a specific item in a dictionary**

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

Ans: `{'brand': 'Ford', 'model': 'Mustang', 'year': 2018}`

4. **Get the length of a dictionary**

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

Ans: `3`

5. **Add an item to a dictionary**

```
thisdict =    {
 "brand": "Ford",
 "model": "Mustang",
 "year": 1964
```

```
}
thisdict["color"] = "red"
print(thisdict)
```

Ans: `{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}`

6. **Remove an item from a dictionary**

```
thisdict =    {
  "brand": "Ford",
  "model": "Mustang",
  "year": 1964
}
thisdict.pop("model")
print(thisdict)
```

7. **Using the dict() constructor to create a dictionary**

```
thisdict = dict(brand="Ford", model="Mustang", year=1964)
# note that keywords are not string literals
# note the use of equals rather than colon for the assignment
print(thisdict)
```