MIT World Peace University

Data Science for Cybersecurity and Forensics
Third Year B. Tech, Semester 6

Basics of Python

Assignment 1

Prepared By

Krishnaraj Thadesar
Cyber Security and Forensics
Batch A1, PA 10

April 15, 2024

Contents

1 Aim

To implement basic programs in Python.

2 Objectives

1. To learn about the basics of Python programming language.
2. To understand the basic syntax and data types in Python.
3. To implement basic programs in Python.

3 Platform

Operating System: Arch Linux X8664
IDEs or Text Editors Used: Visual Studio Code
Compilers or Interpreters: Python 3.10.1

4 Code

1. Python Program to Find a Series in an Array Consisting of Characters

[28]:
```
array = "wijwoifjasbabcjasidf"

# find a series of elements (a, b, c) in an array

for i in range(len(array)):
if array[i] == 'a':
if array[i+1] == 'b':
if array[i+2] == 'c':
print('found at', i)
break
else:
print('not found')
```

```
found at 11
```

2. Python program to find the occurrence of a particular number in an array

[4]:
```
def find_number_in_array(array, number):
for i in range(len(array)):
if array[i] == number:
return i
return -1

# example
array = [1, 2, 3, 4, 5, 6, 7, 8, 9]
number = 7
```

```
print(find_number_in_array(array, number))
```

```
6
```

3. Find the union and intersection of two arrays in Python

[5]:

```
def find_union_and_intersection(array1, array2):
union = []
intersection = []
for i in range(len(array1)):
if array1[i] in array2:
intersection.append(array1[i])
union.append(array1[i])
for i in range(len(array2)):
if array2[i] not in array1:
union.append(array2[i])
return union, intersection

# example
array1 = [1, 2, 3, 4, 5, 6, 7, 8, 9]
array2 = [2, 4, 6, 8, 10, 12, 14, 16]
print(find_union_and_intersection(array1, array2))
```

```
([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 14, 16], [2, 4, 6, 8])
```

4. Create number variables (int, float and complex) and print their types and values in Python

[6]:

```
# creating number variables
a = 1
b = 2.1
print(type(a))
print(type(b))
```

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

5. Python program to print a string, extract characters from the string

[7]:

```
def extract_characters(string):
characters = []
for i in range(len(string)):
characters.append(string[i])
return characters

# example
string = "Hello World!"
print(extract_characters(string))
```

```
['H', 'e', 'l', 'l', 'o', ' ', 'W', 'o', 'r', 'l', 'd', '!']
```

6. Python Program to print words with their length of a string

[9]:

```
def print_words_with_length(string):
words = string.split()
for i in range(len(words)):
print(words[i], ":", len(words[i]))

# example
string = "Data science with Cyber Security"
print_words_with_length(string)
```

```
Data : 4
science : 7
with : 4
Cyber : 5
Security : 8
```

7. Python Print EVEN length words

[10]:

```
def print_even_length_words(string):
words = string.split()
for i in range(len(words)):
if len(words[i]) % 2 == 0:
print(words[i])

# example
string = "Data science with Cyber Security"
print_even_length_words(string)
```

```
Data
with
Security
```

8. Read contents of the file using readline() method in Python

[16]:

```
!touch test.txt
! echo "Hello World!" > test.txt
```

[13]:

```
with open("test.txt", "r") as file:
data = file.read()
print(data)
```

```
Hello World!
```

9. Read contents of a file using readline() method and manipulating it in Python

[17]:

```
with open("test.txt", "r") as file:
# using readline
data = file.readline()
print(data)
```

```
Hello World!
```

10. Copy odd lines of one file to another file in Python

[21]:

```
! touch oddeven.txt
! echo "1 2 3 \n 4 5 6 \n 7 8 9" > oddeven.txt
```

[22]:

```
def print_odd_lines(file):
with open(file, "r") as file:
data = file.readlines()
for i in range(len(data)):
if i % 2 == 0:
print(data[i])

# example
file = "oddeven.txt"
print_odd_lines(file)
```

```
1 2 3

7 8 9
```

11. Python program for Linear Search

[23]:

```
def linear_search(array, number):
for i in range(len(array)):
if array[i] == number:
return i
return -1

# example
array = [1, 2, 3, 4, 5, 6, 7, 8, 9]
number = 7
print(linear_search(array, number))
```

```
6
```

12. Python program to print list elements in different ways

[24]:

```
print("directly")
lst = [1, 2, 3, 4, 5, 6, 7, 8, 9]
# print list elements directly
print(lst)
print("for loop")
# print list elements using for loop
for i in range(len(lst)):
print(lst[i])
```

```
# print list elements using while loop

print("while loop")
i = 0
while i < len(lst):
print(lst[i])
i += 1

print("list comprehension")
# print list elements using list comprehension
print([lst[i] for i in range(len(lst))])
```

```
[1, 2, 3, 4, 5, 6, 7, 8, 9]
1
2
3
4
5
6
7
8
9
1
2
3
4
5
6
7
8
9
    [1, 2, 3, 4, 5, 6, 7, 8, 9]
```

13. Python Program for Adding, removing elements in the list

[25]:

```
# program to add and remove elements
lst = [1, 2, 3, 4, 5, 6, 7, 8, 9]
print("before adding element")
print(lst)
# adding element to list
lst.append(10)
print("after adding element")
print(lst)

# removing element from list
lst.remove(10)
print("after removing element")
print(lst)
```

```
before adding element
    [1, 2, 3, 4, 5, 6, 7, 8, 9]
after adding element
    [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
after removing element
    [1, 2, 3, 4, 5, 6, 7, 8, 9]
```

5 Conclusion

In this assignment, we have implemented various Python programs to understand the basic concepts of Python programming. We have implemented programs to find the number in an array, find the union and intersection of two arrays, create number variables and print their types and values, extract characters from a string, print words with their length of a string, print EVEN length words, read contents of the file using readline() method, read contents of a file using readline() method and manipulating it, copy odd lines of one file to another file, linear search, print list elements in different ways, and adding, removing elements in the list.