Gautam Buddha University

Python Programming lab file

Name- Bhavika Sharma
Roll no.- 215/UCF/021
Course code- Cs383
Btech CSE 3rd year

Submitted to

Jyoti Kaurav ma'am

Experiment 1

Aim: To show uses of the operators in Python.

Code:

```
a = 40
b = 70
# arithemetic operators
print("Arithematic operators")
print("+: ", a+b)
print("-: ", a-b)
print("*: ", a*b)
print("/: ", a/b)
print("**: ", a**2)
print("//: ", a//b)
print("+: ", a+b)
print("%: ", a % b)
print("\n")
# relational operators
print("relational operators")
print(a > b)
print(a >= b)
print(a <= b)
print(a < b)
print(a == b)
print(a != b)
print("\n")
# logical operators
print("logical operators")
print(a and b)
print(a or b)
print(not a)
print("\n")
# bitwise operators
print("bitwise operators")
print(a & b)
print(a ^ b)
print(a | b)
print("\n")
# identity operators
print("identity operators")
print(a is b)
print(a is not b)
```

Output:

```
Arithematic operators
+:  110
-:  -30
*:  2800
/:  0.5714285714285714
**:  1600
//:  0
+:  110
%:  40


relational operators
False
False
True
True
False
True


logical operators
70
40
False


bitwise operators
0
110
110


identity operators
False
True
```

Experiment 2

Aim: To display the star pattern using for loops.

Code:

```
num = int(input("enter a number"))
for i in range(num):
    for j in range(1, num-(i+1)):
        print('*', end='')
    print()
```

Output:

```
enter a number 7
* * * * *
* * * *
* * *
* *
*
```

Experiment 3

Aim: To perform using operations using inbuilt methods.

Code:

```
a = " Hello Gautam Buddha University"
print("String operations\n")
print("Original String : ", a)
print("Coverts all to lowercase : ", a.lower())
print("converts all to uppercase : ", a.upper())
print("Swaps the cases of alphabets : ", a.swapcase())
print("Checks whether the string is alphabet or not : ", a.isalpha())
print("Checks whether the string is digit or not : ", a.isdigit())
print("First index of the letter in the given string : ", a.index("r"))
print("Counts the letter it occoured: ", a.count("a"))
print("Splits the string using delimeter : ", a.split(" "))
print("Replaces the world with the given string : ", a.replace("Hello", "Hi"))
print("Checks whether the string end with given word : ", a.endswith("d"))
```

Output:

```
String operations

Original String :   Hello Gautam Buddha University
Coverts all to lowercase :   hello gautam buddha university
converts all to uppercase :   HELLO GAUTAM BUDDHA UNIVERSITY
Checks whether the string is digit or not :  False
First index of the letter in the given string :  26
Counts the letter it occoured:  3
Splits the string using delimeter :  ['', 'Hello', 'Gautam', 'Buddha', 'University']
Replaces the world with the given string :   Hi Gautam Buddha University
Checks whether the string end with given word :  False
```

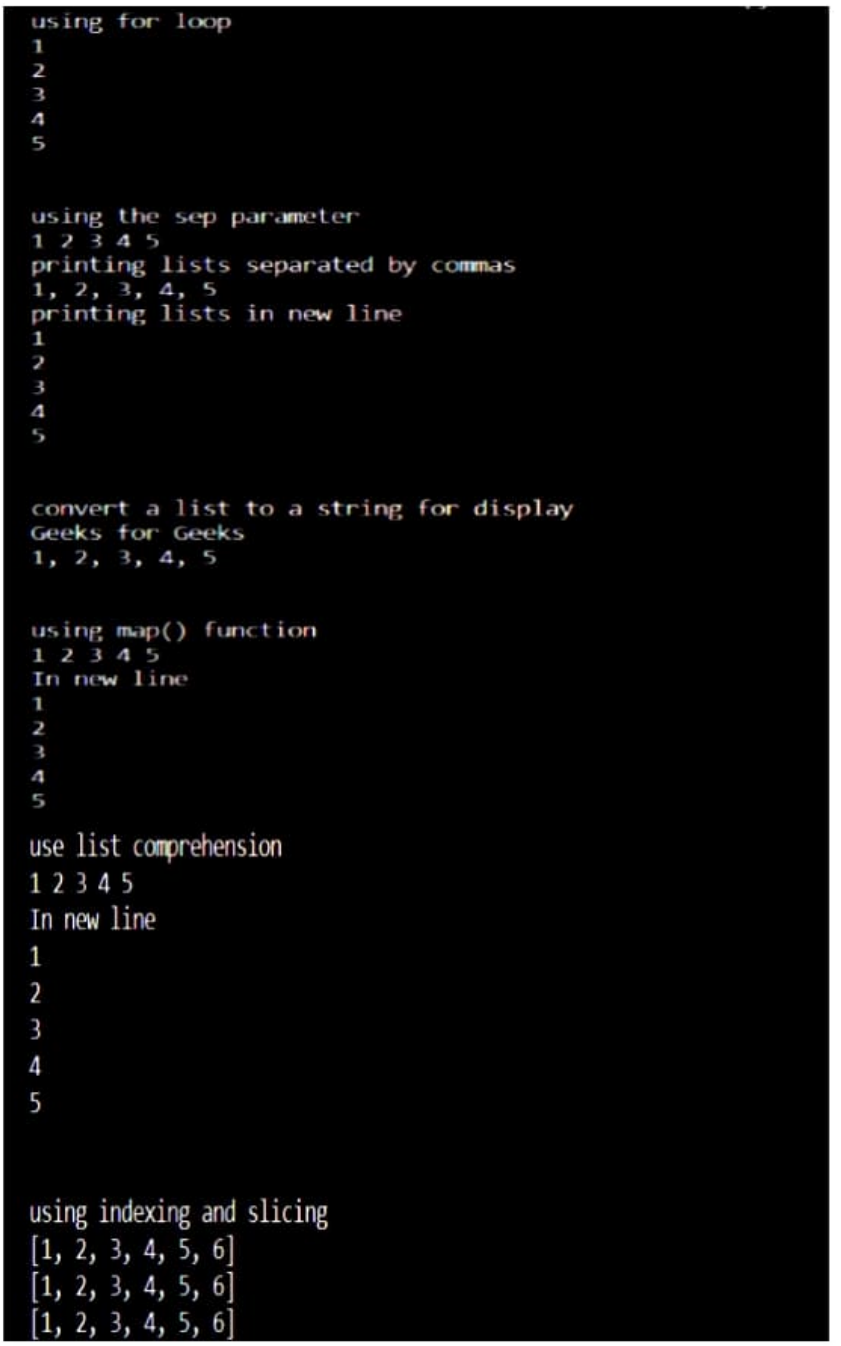
Experiment 4

Aim: Program to print list in six different ways.

Code:

```
a = [1, 2, 3, 4, 5]

# printing the list using loop
print("using for loop")
for x in range(len(a)):
    print(a[x])
print("\n")

# Using the sep parameter in print()
# printing the list using * operator separated by comma
print("using the sep parameter")
print(*a)
# printing the list using * and sep operator
print("printing lists separated by commas")
print(*a, sep=", ")
print("printing lists in new line")
print(*a, sep="\n")
print("\n")

# convert a list to a string for display
print("convert a list to a string for display")
a = ["Geeks", "for", "Geeks"]
# print the list using join function()
print(' '.join(a))
# print the list by converting a list of
# integers to string
a = [1, 2, 3, 4, 5]
print(str(a)[1:-1])
print("\n")

# using map() function
print("using map() function")
a = [1, 2, 3, 4, 5]
print(' '.join(map(str, a)))
print("In new line")
print('\n'.join(map(str, a)))
print("\n")
# Using list comprehension
print("use list comprehension")
a = [1, 2, 3, 4, 5]
[print(i, end=' ') for i in a]
print("\nIn new line")
[print(i) for i in a]
print("\n")

# Using Indexing and slicing
print("using indexing and slicing")
list = [1, 2, 3, 4, 5, 6]
# method 1
print(list[:])
# method 2
print(list[0:])
# method 3
print(list[0:len(list)])
```

Output:


using for loop
1
2
3
4
5
using the sep parameter
1 2 3 4 5
printing lists separated by commas
1, 2, 3, 4, 5
printing lists in new line
1
2
3
4
5
convert a list to a string for display
Geeks for Geeks
1, 2, 3, 4, 5
using map() function
1 2 3 4 5
In new line
1
2
3
4
5
use list comprehension
1 2 3 4 5
In new line
1
2
3
4
5
using indexing and slicing
[1, 2, 3, 4, 5, 6]
[1, 2, 3, 4, 5, 6]
[1, 2, 3, 4, 5, 6]

Experiment 5

Aim: To implement the stack and queue using a list.

Code:

```
class queue_implemntation:
    def __init__(self,list1=[]):
        self.list1=list1

    def enqueue(self,n):
        self.list1.append(n)
    def display(self):
        if(len(self.list1)==0):
            print("empty queue")
        else:
            for i in range(len(self.list1)-1,-1,-1):
                print(self.list1[i],end=" ")
            print("\n")

    def dequeue(self):
        if(len(self.list1)==0):
            print("empty queue")
        else:
            self.list1.pop(0)


class stack_implementation:
    def __init__(self,list1=[]):
        self.list1=list1
    def push(self,n):
        self.list1.append(n)
    def display_stack(self):
        for i in range(len(self.list1)-1,-1,-1):
            print(self.list1[i],end=" ")
        print("\n")
    def pop_stack(self):
        if(len(self.list1)==0):
            print("Stack is empty")
        else:
            self.list1.pop()
```

```
obj1=stack_implementation()
obj1.push(1)
obj1.display_stack()
obj1.push(2)
obj1.push(3)
obj1.display_stack()
obj1.pop_stack()
obj1.display_stack()


obj2=queue_implemntation()
obj2.enqueue(3)
obj2.enqueue(4)
obj2.enqueue(1)
obj2.enqueue(69)
obj2.display()
obj2.dequeue()
obj2.display()
```

Output:

```
1

3 2 1

2 1

69 1 4 3

69 1 4
```

Experiment 6

Aim: To use the dictionary and make a dictionary of faculty and students and store them separately in lists.

Code:

```
faculty_dict = {
    'faculty_id1': {'name': 'ABC', 'department': 'Computer Science'},
    'faculty_id2': {'name': 'EFG', 'department': 'Mathematics'},
    # Add more faculty members as needed
}

students_dict = {
    'student_id1': {'name': 'PQR', 'major': 'Physics'},
    'student_id2': {'name': 'XYZ', 'major': 'History'},
    # Add more students as needed
}

# Store dictionaries in lists
faculty_list = list(faculty_dict.values())
students_list = list(students_dict.values())

# Example: Accessing information
print("Faculty List:")
for faculty in faculty_list:
    print(f"Name: {faculty['name']}, Department: {faculty['department']}")

print("\nStudents List:")
for student in students_list:
    print(f"Name: {student['name']}, Major: {student['major']}")
```

Output:

```
Faculty List:
Name: ABC, Department: Computer Science
Name: EFG, Department: Mathematics

Students List:
Name: PQR, Major: Physics
Name: XYZ, Major: History
```

Experiment 7

Aim: To show the use of lambda expression.

Code:

```
A=lambda x:x+6
print(A(6))

list1=list(map(int,input().split()))
print(list1)

y=lambda x,z: z if A(z)+3>x else 6

print(y(6,3))
```

Output:

```
12
1 2 3 4 5 6 7 8 9 0
[1, 2, 3, 4, 5, 6, 7, 8, 9, 0]
3
```

Experiment 8

Aim: To demonstrate the use of File writing and reading in text file.

Code:

```
# File Writing
with open("file1.txt", "w") as f:
    while (1 == 1):
        line = input("enter the lines : ")
        f.write(line)
        f.write("\n")
        choice = input("are you done(Y/N) : ")
        if(choice.lower() == "y"):
            break
        else:
            pass
    f.close()
    print("written in file successfully")
with open("file1.txt", "r") as g:
    print("Reading the file\n")
    print(g.read())
```

Output:

```
enter the lines : Hii I learn C
are you done(Y/N) : n
enter the lines : I also learn Python
are you done(Y/N) : y
written in file successfully
Reading the file

Hii I learn C
I also learn Python
```

Experiment 9

Aim: To demonstrate error handling.

Code:

```
try:
    a = int(input("enter the number : "))
    print(a/2)
    print(a/0)
except (ArithmeticError, ValueError):
    print("An error Occoured\n")
```

Output:

```
enter the number : 7
3.5
An error Occoured
```

```
enter the number : a
An error Occoured
```

Experiment 10

Aim: To demonstrate Multiple inheritance using classes

Code:

```
class Employee:
    def __init__(self, name):
        self.name = name

    def shwow(self):
        print(f"the name is {self.name}")


class Dancer:
    def __init__(self, dance):
        self.dance = dance

    def show(self):
        print(f"the dance is {self.dance}")


class DancerEmployee(Employee, Dancer):
    def __init__(self, name, dance):
        self.name = name
        self.dance = dance


o = DancerEmployee("ABC", "Kathak")
print(o.name)
print(o.dance)
o.show()
print(DancerEmployee.mro())
```

Output:

```
ABC
Kathak
The dance is Kathak
[<class '__main__.DancerEmployee'>, <class '__main__.Employee'>, <class '__main__.Dancer'>, <class 'object'>]
```

Experiment 11

Aim: To use NumPy and Pandas to generate a list.

Code:

```
import pandas as pd
import numpy as np

# Creating empty series
ser = pd.Series()
print("Pandas Series: ", ser)

# simple array
data = np.array(['g', 'e', 'e', 'k', 's'])

ser = pd.Series(data)
print("Pandas Series:\n", ser)
```

Output:

```
Pandas Series:  Series([], dtype: object)
Pandas Series:
 0    g
1    e
2    e
3    k
4    s
dtype: object
```