BHARATI VIDYAPEETH'S

INSTITUTE OF COMPUTER APPLICATIONS & MANAGEMENT

(Affiliated to Guru Gobind Singh Indraprastha University,Approved by AICTE, New Delhi)

Python Programming (MCA-106)

Practical File

Submitted To:

Dr. Rakhee

(Assistant Professor)

Submitted By:

Abhijeet Rana
01811604422
MCA 2nd Sem, Sec 1

Index

S.No	Problem Description	Date of execution	Date of submission
1.	Implement Python Script to generate first N natural numbers.		
2.	By considering the terms in the Fibonacci sequence whose values do not exceed 1000, find the sum of the even-valued terms		
3.	Write a program which makes use of function to display all such numbers which are divisible by 7 but are not a multiple of 5, between 1000 and 2000.		
4.	Write a function cumulative product to compute cumulative product of a list of numbers.		
5.	Write a function reverse to reverse a list. Without using the reverse function.		
6.	Define a function which generates Fibonacci series up to n numbers using RECURSION.		
7.	With a given tuple (1, 2, 3, 4, 5, 6, 7, 8, 9, 10), write a program to print the first half values in one line and the last half values in one line.		
8.	Write a program to count the numbers of characters in the string and store them in a dictionary data structure.		
9.	Remove spaces from a string using recursion.		
10.	Write a program to compute the number of characters, words and lines in a file.		

1. Implement Python Script to generate first N natural numbers.

Code :

```
def generateNaturalNumbers (n) :
   for i in range (1, n+1) :
         print(i, end=" ")


x = int(input("enter a number : "))
generateNaturalNumbers(x)
```

```
============= RESTART: C:/Users/abhij/Desktop/Python Output/Q1.py =============
enter a number : 22
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
>>>
```

2. By considering the terms in the Fibonacci sequence whose values do not exceed 1000, find the sum of the even-valued terms.

Code :

```
def fibSum():
   first_term = 0
   second_term = 1
   result = 0
   while second_term <= 1000 :
      if second_term % 2 == 0 :
         result += second_term
      temp = second_term
      second_term += first_term
      first_term = temp

   return result


x = fibSum()
```

print(f'Sum of even valued terms of fibonacci sequence whose value do not exceed 1000 is {x}')

Output :

```
============== RESTART: C:/Users/abhij/Desktop/Python Output/Q2.py =============
Sum of even valued terms of fibonacci sequence whose value do not exceed 1000 is 798
```

3. Write a program which makes use of function to display all such numbers which are divisible by 7 but are not a multiple of 5, between 1000 and 2000.

Code :

```
def func ():
   x = 1001
   while ( x % 7 != 0) :
         x += 1
   for i in range (x, 2000, 7) :
       if(i % 5 != 0 ) :
            print( i, end = " ")
print("The required series is as follows : ")
func()
```

Output :

```
============= RESTART: C:/Users/abhij/Desktop/Python Output/Q3.py =============
The required series is as follows :
1001 1008 1022 1029 1036 1043 1057 1064 1071 1078 1092 1099 1106 1113 1127 1134 1141 1148 1162 1169 1176 1183 1197
1204 1211 1218 1232 1239 1246 1253 1267 1274 1281 1288 1302 1309 1316 1323 1337 1344 1351 1358 1372 1379 1386 1
393 1407 1414 1421 1428 1442 1449 1456 1463 1477 1484 1491 1498 1512 1519 1526 1533 1547 1554 1561 1568 1582 15
89 1596 1603 1617 1624 1631 1638 1652 1659 1666 1673 1687 1694 1701 1708 1722 1729 1736 1743 1757 1764 1771 17
78 1792 1799 1806 1813 1827 1834 1841 1848 1862 1869 1876 1883 1897 1904 1911 1918 1932 1939 1946 1953 1967 19
74 1981 1988
```

4. Write a function cumulative_product to compute cumulative product of a list of numbers.

Code :

```
def cummulative_product ( list ):
   res = 1
   for ele in list :
      res *= ele
   return res
```

```
length_of_arr = int(input('Enter the length of the array : '))

print('Enter the elements of the array')

list = list()

for i in range (0, length_of_arr) :

   list.append(int(input()))


print(f'the cummulative product of the list is : {cummulative_product(list)}')
```

Output :

```
============== RESTART: C:/Users/abhij/Desktop/Python Output/Q4.py =============
Enter the length of the array : 5
Enter the elements of the array
1
2
3
4
5
the cummulative product of the list is : 120
```

5. Write a function reverse to reverse a list. Without using the reverse function.

Code :

```
def reverseList(list) :

   n = len(list)

   mid = (n-1)//2

   for i in range (0, mid) :

      temp = list[i]

      list[i] = list[n-1-i ]

      list[ n-1-i ] = temp


list= ['r', 'a', 'b', 'b', 'i', 't']

print( 'original list ', list )

reverseList(list)


print( 'reversed list ', list )
```

Output :

```
============= RESTART: C:/Users/abhij/Desktop/Python Output/Q5.py =============
original list  ['r', 'a', 'b', 'b', 'i', 't']
reversed list  ['t', 'i', 'b', 'b', 'a', 'r']
```

6. Define a function which generates Fibonacci series up to n numbers using RECURSION

Code :

```
def fib (n) :
    if  n == 0 :
        return 0
    if  n == 1 or n == 2  :
        return 1
    return fib(n-1) + fib(n-2)

def printFib(n):
    for i in range(0 ,n) :
        print(fib(i))

x = int(input("Enter a number : "))
print("The fibbonacci terms are as follow : ")
printFib(x)
```

Output:

```
============= RESTART: C:/Users/abhij/Desktop/Python Output/Q6.py =============
Enter a number : 4
The fibbonacci terms are as follow :
0
1
1
2
```

7. With a given tuple (1, 2, 3, 4, 5, 6, 7, 8, 9, 10), write a program to print the first half values in one line and the last half values in one line.

Code :

```
def printTuple (t1 ):
    mid = len(t1)//2
    print(t1[:mid])
    print(t1[mid:])

t1 = (1, 2,3, 4, 5, 6, 7, 8, 9, 10)
printTuple(t1)
```

Output:

```
============= RESTART: C:/Users/abhij/Desktop/Python Output/Q7.py =============
(1, 2, 3, 4, 5)
(6, 7, 8, 9, 10)
>>>
```

8. Write a program to count the numbers of characters in the string and store them in a dictionary data structure

Code :

```
def createMap(str) :
    map = dict()
    for char in str:
        if char in map.keys() :
            map[char] = map[char] + 1
        else:
            map[char] = 1
    return map

str = input("Enter your string : ")
print("The generated map is : ")
map = createMap(str)
print(map)
```

Output :

```
============= RESTART: C:/Users/abhij/Desktop/Python Output/Q8.py =============
Enter your string : This is just for testing purpose
The generated map is :
{'T': 1, 'h': 1, 'i': 3, 's': 5, ' ': 5, 'j': 1, 'u': 2, 't': 3, 'f': 1, 'o': 2, 'r': 2, 'e': 2, 'n': 1, 'g': 1, 'p': 2}
>>>
```

9. Remove spaces from string using recursion.

Code :

```
def stripSpaces( str ):
    if len(str ) == 0 : return ""
    if str[0] == " " :
        return stripSpaces ( str[1:] )
    else:
        return str[0:1] + stripSpaces(str[1:])
str = input("Enter a string : ")
print("String after striping spaces")
print(stripSpaces(str))
```

Ouptut :

```
============= RESTART: C:/Users/abhij/Desktop/Python Output/Q9.py =============
Enter a string : This is also for testing purpose
String after striping spaces
Thisisalsofortestingpurpose
>>>
```