SIKSHA 'O' ANUSANDHAN
DEEMED TO BE UNIVERSITY

Admission Batch: 2019 **Session:** 2021-22

Laboratory Record

Programming in Python (CSE 3142)

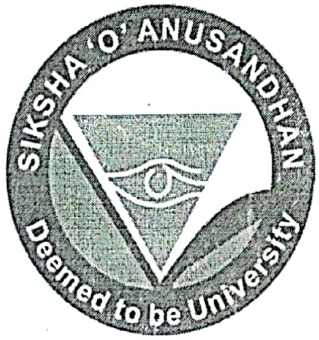
Submitted by

Name: Saswat Mohanty

Registration No.: 1941012407

Branch: Computer Science and Engineering

Semester: 5th Section: 'D'



Department of Computer Science & Engineering

Faculty of Engineering & Technology (ITER)

Jagamohan Nagar, Jagamara, Bhubaneswar, Odisha - 751030

INDEX

Sl. No	Name of Program	Page No	Remarks
1.	Minor Assignment - 1	1-8	good 06/12/21
2.	Minor Assignment - 2	9-21	good 06/12/21
3.	Minor Assignment - 3	22-39	good 06/12/21
4.	Minor Assignment - 5	40-45	good 21/12/21
5.	Minor Assignment - 6	46-57	good 21/12/21
6.	Minor Assignment - 7	58-68	good 21/12/21
7.	Minor Assignment - 8	69-77.	

Minor Assignment - 8

Recursion

Q1) Write a recursive function that converts a number inputted in string form to an integer type. For example, if input string is: '1234' then the recursive function should convert it to 1234 (int type).

Program :-

```
def strtoint(str):
    if len(str) == 0:
        return 0
    return strtoint(str[:-1]) * 10 + ord(str[-1]) - 48
print(strtoint("1234"))
```

Output :- 1234

Q2) Write a recursive function to print the sum of the digits of a given non-negative integer.

Program :-

```
def sum(n):
    if n == 0:
        return 0
    return n%10 + sum(n//10)
print(sum(1234))
```

Output :-

10

Q3) Write a recursive function to calculate the value of 'a' to the power 'b'. For example, if a = 2 and b = 3, the output should be 2**3 = 8.

Program :-

```
def power(n, m):
```

```
if m==0:
    return 1
elif m==1:
    return n
return (n* power (n,m-1))
print (power(2,3))
```

output :-

8

Q4) Write a recursive function to calculate the harmonic sum of first n terms. Note: The harmonic sum is the sum of reciprocals of the positive integers. For example, if n=4, the output should be (1+1/2+1/3+1/4)=2.0833.

Program :-

```
def sum(n):
    if n==1:
        return 1
    return 1/n+(sum(n-1))
print (sum(4))
```

output :-

2.0833333333333333

Q3) write a recursive function to calculate the geometric sum of first n terms with constant ration r, where r lies in the interval (0,1). Note: In mathematics, a geometric series is a series with a constant ratio between successive terms. For example, if n=4 and r=(1/2) then output should be (1+1/2+1/4+1/8)=1.875

Program :-

```
def sum(n,r):
    if n==0:
        return 0
```

```
    else:
        return pow(x,n-1)+sum(n-1,x)
print(sum(4,1/2)).
```

Output :-

1.875

Q6) write a recursive function to calculate the sum of the positive integers of $n+(n-2)+(n-4)\ldots.$ (until $n-x \leq 0$). For example, if $n=6$, then output should be $6+(6-2)+(6-4)+(6-6)=12$.

Program :-

```
def sum(n):
    if (n<1):
        return 0
    return n + sum(n-2)
print(sum(5))
```

Output :-

9

Q7) write a recursive function to print the sums of all subsets of a given array. For example : Input : 1st = [2,3] Output : 0, 2, 3, 5.
Input : 1st = [2,4,5] output : 0, 2, 4, 5, 6, 7, 9, 11.

Program :-

```
def powersetsum(arr, arr1, n):
    if n=0:
        print(sum(arr1))
        return
    arr1.append(arr[n-1])
    powersetsum(arr, arr1, n-1)
    arr1.pop()
    powersetsum(arr, arr1, n-1)
arr=[2,3]
```

```
arr1 = []
powersetsum(arr, arr1, len(arr)).
```

Output :-

```
5
3
2
0
```

Q8) Write a recursive function to check whether a given number is prime or not.

Program :-

```
def primeChecking(i,n):
    if n==i:
        return 0
    else:
        if n%i==0:
            return 1
        else:
            return primeChecking(i+1,n)
if primeChecking(2,9)==0:
    print('It is prime')
else:
    print('It is not prime').
```

Output :-

```
It is not prime.
```

Q9) Write a recursive function that multiplies two positive numbers a and b, and returns the result. Multiplication is to be achieved as a+a+a (b times).

Program :-

```
def mul(a,b):
    if b==0:
```

```
    return 0
  return a + mul(a, b-1)
print(mul(2,3)).
```

Output :-

6

Q10) write a recursive function that takes number n as an input parameter and prints n-digit strictly increasing numbers.

Program :-

```
def findStrictlyIncreasingNum(start, out, n):
  if(n==0):
     print(out, end=" ")
     return
  for i in range(start, 10):
     str1 = out + str(i)
     findStrictlyIncreasingNum(i+1, str1, n-1).
```

Q11) write a recursive function that generates all binary strings of n-bit length.

Program :-

```
def gen(n, str=''):
  if n>0:
     gen(n-1, str+'0')
     gen(n-1, str+'1')
  else:
     print(str)
gen(3)
```

Output :-

```
000     011     110
001     100     111
010     101
```

Q12) Write a recursive function that takes two strings as input parameters and prints all interleaving strings of the given two strings preserving their order of occurence. For example, interleaving of strings 'AB' and 'CD' will generate the strings: 'ABCD', 'ACBD', 'ACDB', 'CDAB', 'CADB' and 'CABD'.

Program :-

```
def combine(str1, str2, res, i, j, lst):
    if i == len(str1) and j == len(str2):
        lst.append(res)
        return
    if i < len(str1):
        combine(str1, str2, res+str1[i], i+1, j, lst)
    if j < len(str2):
        combine(str1, str2, res+str2[j], i, j+1, lst)
str1 = 'AB'
str2 = 'CD'
lst = []
combine(str1, str2, '', 0, 0, lst)
print(lst).
```

Output :-

['ABCD', 'ACBD', 'ACDB', 'CABD', 'CADB', 'CDAB']

Q13) Write a recursive function that inserts the element x at every k^{th} position in the given list, and returns the modified list. For example, if we wish to insert element 50 at every 3rd position (counting 0, 1, 2, 3) in the list [1, 2, 3, 4, 5, 6, 7], the output list will be [1, 2, 3, 50, 4, 5, 6, 50, 7].

Program :-

```
def insrd(lst, x, k, lst2=[]):
    if len(lst)>k-1:
        lst2=lst[:k]
        lst2.append(x)
        return lst2+insrd(lst[k:],x,k)
    elif len(lst)>0:
        return lst2+lst
    else:
        return lst2
lst=[1,2,3,4,5,6,7]
x=50
k=3
print(insrd(lst,x,k))
```

Output :-

[1, 2, 3, 50, 4, 5, 6, 50, 7]

Q14) Write a recursive function that deletes every k^{th} element, and returns the modified list. For example, if we wish to delete every 3rd element from the list [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11], the output list will be [1, 2, 4, 5, 7, 8, 10, 11].

Program :-

```
def delete(lst, curr, n):
    ri=curr+n
    if(ri>=len(lst)+1):
        return
    lst.remove
    delete(lst,ri,n)
```

```
lst = [1,2,3,4,5,6,7,8,9,10,11]
delete(lst,0,3)
print(lst)
```

Output :-

```
[1,2,4,5,7,8,10,11]
```

Q15) Write a recursive function that recursively removes adjacent duplicates from a given list, and returns the modified list. For example, removing adjacent duplicates recursively from the list [1,2,4,4,5,7,7,7,8,8,9,7] will yield list [1,2,5,9,7].

Program :-

```
def remove(lst,i):
    if i==len(lst)-1:
        return
    if not lst:
        return
    if lst[i]==lst[i+1]:
        tmp = lst[i]
        while(i<len(lst) and lst[i]==tmp):
            lst.pop(i)
        if not i-1:
            remove(lst,i-1)
        else:
            remove(lst,0)
    else:
        remove(lst,i+1)
inp = [1,2,4,4,5,7,7,7,8,8,9,7]
remove(inp,0)
print(inp)
```

Output :-

[1, 2, 5, 9, 7]

Q16) Write a recursive function that takes two numbers as input parameters and computes their greatest common divisor.

Program :-

```
def gcd(a,b):
  if b==0:
     return a
  return gcd(b, a%b)
print(gcd(5,20))
```

Output :-

5.