GAYATRI VIDYA PARISHAD COLLEGE OF ENGINEERING

(AUTONOMOUS)

Madhurawada, Visakhapatnam-530048

COLLEGE OF ENGINEERING

(AUTONOMOUS)

CERTIFICATE

Certified that this is a bonifide record of practical done by ____ bearing Roll No. ______________of MCA I semester successfully completed Introduction to python programming lab from the department of computer applications during the academic year 2020-21.

No. of Experiments done:

Signature of the Faculty

Signature of Internal Examiner:

Signature of External Examiner:

INDEX

	given number is Armstrong or not.				
f)	Implement Python Script to generate prime numbers series up to n		7		
IV	**Exercise programs on Lists.**				
a)	Finding the sum and average of given numbers using lists.		7		
b)	To display elements of list in reverse order.		7		
c)	Finding the minimum and maximum elements in the lists.		8		
V	**Exercise programs on Strings.**				
a)	Implement Python Script to perform various operations on string using string libraries.		8		
b)	Implement Python Script to check given string is palindrome or not.		8		
c)	Implement python script to accept line of text and find the number of characters, number of vowels and number of blank spaces in it.		9		
VI	**Exercise programs on functions.**				
a)	Define a function max_of_three() that takes three numbers as arguments and returns the largest of them.		9		
b)	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.		9-12		
VII	**Exercise programs on recursion & parameter passing techniques.**				
a)	Define afunction which generates Fibonacci series up ton numbers.		12-13		
b)	Define a function that checks whether the given number is Armstrong.		13		
c)	Implement a python script for Call-by-value and Call-by-reference.		13-14		
d)	Implement a python script for factorial of number by using recursion.		14		
VIII	**Exercise programs on Tuples.**				
a)	Write a program which accepts a sequence of comma-separated numbers from console and		14		

	generate a list and a tuple which contains every number. Suppose the following input is supplied to theprogram: 34,67,55,33, 12, 98. Then, the output should be: ['34', '67', '55', '33', '12', '98'] ('34',67', '55', '33', '12', '98').				
b)	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.		14-15		
IX	**Exercise programs on files.**				
a)	Write Python script to display file contents.		15		
b)	Write Python script to copy file contents from one file to another.		15		
X	**Exercise programs on searching & sorting Techniques.**				
a)	Implement a python script to check the element is in the list or not by using Linear search &Binary search.		16		
b)	Implement a python script to arrange the elements in sorted order using Bubble, Selection, Insertion and Merge sorting techniques.		17-18		

INTRODUCTION TO PYTHON PROGRAMMING

I. Exercise programs on basic control structures & loops.
a) Write a program for checking the given number is even or odd.

```
n=int(input("Enter is a number"))
if(n%2==0):
  print(f"{n} is even")
else:
  print(f"{n} is odd")
```

Output:

```
Enter is a number10
10 is even
Enter is a number9
9 is odd
```

b) Using a for loop, write a program that prints the decimal equivalents of 1/2, 1/3, 1/4 ,.......1/10

```
for i in range(2,11):
   print((1/i), '  ')
```

Output:

```
0.5
0.3333333333333333
0.25
0.2
0.16666666666666666
0.14285714285714285
0.125
0.1111111111111111
0.1
```

c) Write a program for displaying reversal of a number.

```
a=int(input("Enter a number"))
r=0
while(a>0):
  s=a%10
  r=(r*10)+s
  a=a//10
  print(r)
```

Output:

```
Enter a number123
3
32
321
```

d) Write a program for finding biggest number among 3 numbers.

```
a=int(input("Enter a value"))
b=int(input("Enter b value"))
c=int(input("Enter c value"))
if (a>b and a>c):
  print(f"{a} is big")
elif(b>c and b>a):
  print(f"{b} is big")
else:
  print(f"{c} is big")
```

Output:

```
Enter a value90
Enter b value100
Enter c value45
100 is big
```

e) Write a program using a while loop that asks the user for a number, and prints a countdown from that number to zero.

```
n=int(input("Enter values"))
while(n+1>=0):
  print(n)
  n=n-1
```

Output:

```
Enter values5
5
4
3
2
1
0
```

II. Exercise programs on operators & I/O operations.

a) Write a program that takes 2 numbers as command line arguments and prints its sum.

```
import sys
x=int(sys.argv[1])
y=int(sys.argv[2])
sum=x+y
print("sum of two numbers is",sum)
```

Output:

```
sum.py 12 34
sum of two numbers is 46
sum.py 20 30
sum of two numbers is 50
```

b) Implement python script to show the usage of various operators available in python language.

```
a=int(input("enter a number"))
b=int(input("enter a number"))
print(f"addition of a and b{a+b}")
print(f"subtraction of a and b {a-b}")
print(f"subtraction of a and b {a*b}")
print(f"division of a and b {a/b}")
print(f"exponentional of a and b {a**b}")
print(f"modulus of a and b {a%b}")
print(f"floor division of a and b {a//b}")
print(f"a is greater than b {a>b}")
print(f"a is lessthan b {a<b}")
print(f"a is greathan or equal to b {a>=b}")
print(f"a is lessthan or equal to b {a<=b}")
print(f"a is equals to b {a==b}")
print(f"a is not equals to b {a!=b}")
print(f"a and b is {a and b}")
print(f"a and b is{a or b}")
print(f"not a {not a}")
print(f"a binary and b {a&b}")
print(f"a or b {a|b}")
print(f" a xor b {a^b}")
print(f" a one's complement {~a}")
print(f" a leftshift {a<<2}")
print(f" a rightshift {a>>2}")
```

Output:

```
enter a number10
enter a number20
addition of a and b30
subtraction of a and b -10
subtraction of a and b 200
division of a and b 0.5
exponentional of a and b 100000000000000000000
modulus of a and b 10
floor division of a and b 0
a is greater than b False
a is lessthan b True
```

```
a is greathan or equal to b False
a is lessthan or equal to b True
a is equals to b False
a is not equals to b True
a and b is 20
a and b is10
not a False
a binary and b 0
a or b 30
 a xor b 30
 a one's complement -11
 a leftshift 40
 a rightshift 2
```

c) Implement python script to read person’s age from keyboard and display whether he is eligible for voting or not.

```
a=int(input("Enter age"))
if(a>=18):
  print("eligible")
else:
  print("Not eligible")
```

Output:

```
Enter age18
eligible
Enter age10
Not eligible
```

d) Implement python script to check the given year is leap year or not.

```
year=int(input("Enter year"))
if((year%400==0)and(year%4==0)):
 print(f"{year} is leap year")
else:
 print(f"{year} is not a leap year")
```

Output:

```
Enter year700
700 is not a leap year
Enter year1200
1200 is leap year
```

III. Exercise programs on Python Script.

a) Implement Python Script to generate first N natural numbers.

```
a=int(input("Enter Number"))
for a in range(1,a+1):
  print(a)
```

Output:

```
Enter Number6
1
2
3
4
5
6
```

Using While Condition:

```
a=int(input("Enter Number"))
i=0
while(i<a):
  i=i+1
  print(i)
```

Output:

```
Enter Number4
1
2
3
4
```

b) Implement Python Script to check given number is palindrome or not.

```
a=int(input("Enter a Number"))
b=a
r=0
while(a>0):
 s=a%10
r=(r*10)+s
a=a//10
if(r==b):
 print(f"{b} is palindrome")
else:
 print(f"{b} is not a palindrome")
```

Output:

```
Enter a Number121
121 is palindrome
Enter a Number123
123 is not a palindrome
```

c) Implement Python script to print factorial of a number.

```
n=int(input("Enter a number"))
i=1
f=1
for n in range(1,n+1):
 f=f*i
 i=i+1
 print(f)
```

Output:

```
Enter a number4
1
2
6
24
```

d) Implement Python Script to print sum of N natural numbers.

```
n=int(input("Enter a number"))
s=n*(n+1)/2
print(s)
```

Output:

```
Enter a number10
55.0
```

e) Implement Python Script to check given number is Armstrong or not.

```
n=int(input("Enter a number"))
t=n
s=0
while(n!=0):
 r=n%10
s=s+(r*r*r)
n=n//10
if(t==s):
 print(f"{t} is an Armstrong Number")
else:
 print(f"{t} is not an Armstrong Number")
```

Output:

```
Enter a number153
```

```
153 is an Armstrong Number
Enter a number151
151 is not an Armstrong Number
```

f) Implement Python Script to generate prime numbers series up to n.

```
Number = int(input(" Enter any Number: "))
print("Prime numbers between", 1, "and", Number, "are:")
for num in range(1, Number +1):
   if num > 1:
       for i in range(2, num):
           if (num % i) == 0:
               break
       else:
           print(num)
```

Output:

```
Enter any Number: 20
Prime numbers between 1 and 20 are:
2
3
5
7
11
13
17
19
```

IV. Exercise programs on Lists.

a) Finding the sum and average of given numbers using lists.

```
list1=[1,2,3,4,5]
print("sum of list is",sum(list1))
print("Avg of list is",(sum(list1)/len(list1)))
```

Output:

```
sum of list is 15
Avg of list is 3.0
```

b) To display elements of list in reverse order.

```
list1=[1,2,3,4,5]
print("Reverse of the list is" ,(list1[::-1]))
```

Output:

```
Reverse of the list is [5, 4, 3, 2, 1]
```

c) Finding the minimum and maximum elements in the lists.

```
list1=[1,2,3,4,5]
print("minimum element in the list is" ,min(list1))
print("maximum element in the list is",max(list1))
```

Output:

```
minimum element in the list is 1
maximum element in the list is 5
```

V. Exercise programs on Strings.

a) Implement Python Script to perform various operations on string using string libraries.

```
s="welcome to gvp CoLLEGE"
print(s.capitalize())#to captialize the string
print(s.casefold())#to casefold the length of a string
print(len(s))#to find the length of a string
print(s.lower())#to convert to lowercase
print(s.upper())#to convert to uppercase
print(s.split("o"))#to split the given string at given number
```

Output:

```
Welcome to gvp college
welcome to gvp college
22
welcome to gvp college
WELCOME TO GVP COLLEGE
['welc', 'me t', ' gvp C', 'LLEGE']
```

b) Implement Python Script to check given string is palindrome or not.

```
s=input("Enter a string:")
if(s==s[::-1]):
 print("The given string is palindrome")
else:
 print("The given string is not a palindrome")
```

Output:

```
Enter a string:madam
The given string is palindrome

Enter a string:corona
The given string is not a palindrome
```

c) Implement python script to accept line of text and find the number of characters, number of vowels and number of blank spaces in it.

```
s=input("enter text:")
p="abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ"
q="aeiouAEIOU"
c=0
v=0
b=0
for  i  in  s:
    if i in q:
        v=v+1
    if i in p:
        c=c+1
    else:
        b=b+1
print("No of characters are:",c)
print("No of vowels are:",v)
print("No of blank spaces are:",b)
```

Output:

```
enter text:Hello world
No of characters are: 10
No of vowels are: 3
No of blank spaces are: 1
```

VI. Exercise programs on functions.

a) Define a function max_of_three() that takes three numbers as arguments and returns the largest of them.

```
def max_of_three(a,b,c):
        return max(a,b,c)
a=int(input("Enter first number:"))
b=int(input("Enter second number:"))
c=int(input("Enter third number:"))
max_val=max_of_three(a,b,c)
print("Largest of three numbers",max_val)
```

Output:

```
Enter first number:67
Enter second number:45
Enter third number:88
Largest of three numbers 88
```

b) 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.

```
def divisible (num):
    if(num%7==0)and(num%5!=0):
        return num
start=1000
end=2000
for i in range(start,end):
  res=divisible(i)
  if(res!=None):
     print(res)
```

Output:

```
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
1393
1407
1414
1421
1428
1442
1449
1456
1463
1477
1484
1491
1498
1512
1519
1526
1533
1547
1554
1561
1568
1582
1589
1596
1603
1617
1624
1631
1638
1652
1659
1666
1673
1687
1694
1701
1708

```
1722
1729
1736
1743
1757
1764
1771
1778
1792
1799
1806
1813
1827
1834
1841
1848
1862
1869
1876
1883
1897
1904
1911
1918
1932
1939
1946
1953
1967
1974
1981
1988
```

VII. Exercise programs on recursion & parameter passing techniques.

a) Define a function which generates Fibonacci series up to n numbers.

```
def fib(n):
    a = 0
    b = 1
    if n == 1:
        print(a)
    else:
        print(a)
        print(b)
        for i in range(2,n):
            c = a + b
            a = b
            b = c
            print(c)
fib(10)
```

Output:

```
0
1
1
2
3
5
8
13
21
34
```

b) Define a function that checks whether the given number is Armstrong.

```
def armstrong():
        num=int(input("Enter a number"))
        temp=num
        s=0
        c=0
        while temp!=0:
                c=c+1
                temp=int(temp/10)
        temp=num
        while temp!=0:
                rem= temp%10
                p=1
                for i in range(0,c):
                        p=p*rem
                s=s+p
                temp= int(temp/10)
        if(s==num):
                print("armstrong number")
        else:
                print("not a armstrong number")
armstrong()
```

Output:

```
Enter a number:77
not a armstrong number
Enter a number:153
armstrong number
```

c) Implement a python script for Call-by-value and Call-by-reference.

```
def call_by_value(s):
 s="python lab programs"
 return s
s='python'
```

```
res=call_by_value(s)
print(res)
def call_by_reference(l):
    l[0]=50
    l.append(80)
    return l
l=[20,30,40,50]
res=call_by_reference(l)
print(res)
```

Output:

```
 python lab programs
  [50, 30, 40, 50, 80]
```

d) Implement a python script for factorial of number by using recursion.

```
def factorial (num):
 if num==1:
  return num
 else:
  return num*factorial(num-1)
n=int(input("Enter a number:"))
print(factorial(n))
```

Output:

```
Enter a number:10
3628800
```

VIII. Exercise programs on Tuples.

a) Write a program which accepts a sequence of comma-separated numbers from console and generate a list and a tuple which contains every number. Suppose the following input is supplied to the program: 34, 67, 55, 33, 12, 98. Then, the output should be: ['34', '67', '55', '33', '12', '98'] ('34',67', '55', '33', '12', '98').

```
values = input("Input some comma seprated numbers : ")
list = values.split(",")
tuple = tuple(list)
print('List : ',list)
print('Tuple : ',tuple)
```

Output:

```
Input some comma seprated numbers : 34,67,55,33,12,98
List : ['34', '67', '55', '33', '12', '98']
Tuple : ('34', '67', '55', '33', '12', '98')
```

b) 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.

```
tpl = (1,2,3,4,5,6,7,8,9,10)
for i in range(0,5):
    print(tpl[i],end = ' ')
print()

for i in range(5,10):
    print(tpl[i],end = ' ')
```

Output:

```
1 2 3 4 5
6 7 8 9 10
```

IX. Exercise programs on files.

a) Write Python script to display file contents.

```
f=open("d:/python.txt","r+")
print("displaying file contents using read method")
print(f.read(16))
print(f.readline())
print(f.readlines())
f.close()
```

Output:

```
displaying file contents using read method
hii good morning
```

b)Write Python script to copy file contents from one file to another.

```
a=open("d:/mca.txt","r+")
b=open("d:/mca1.txt","r+")
a.seek(0)
for i in a:
    b.write(i)
a.seek(0)
b.seek(0)
print("printingmca file")
print("          *****************           ")
print(a.readlines())
print("printing mca1 file")
print("          *****************           ")
print(b.readlines())
```

Output:

```
printingmca file
          *****************
['welcome to python']
printing mca1 file
          *****************
['welcome to python']
```

X. Exercise programs on searching & sorting Techniques.
a) Implement a python script to check the element is in the list or not by using Linear search & Binary search.

Using linear search:

```
def linearsearch(list, n, x):
    for i in range(0, n):
        if (list[i] == x):
            return i
    return -1
list = [2, 4, 0, 1, 9]
x = 1
n = len(list)
result = linearsearch(list, n, x)
if(result == -1):
    print("Element not found")
else:
    print("Element found at index: ", result)
```

Output:

```
Element found at index: 3
```

Using binary search:

```
def binarySearch(list, x, low, high):

    if high >= low:

        mid = low + (high - low)//2
        if list[mid] == x:
            return mid
        elif list[mid] > x:
            return binarySearch(list, x, low, mid-1)
        else:
            return binarySearch(list, x, mid + 1, high)

    else:
        return -1
list = [3, 4, 5, 6, 7, 8, 9]
x = 8
result = binarySearch(list, x, 0, len(list)-1)
if result != -1:
    print("Element is present at index " + str(result))
else:
    print("Not found")
```

Output:

```
Element is present at index 5
```

b) Implement a python script to arrange the elements in sorted order using Bubble, Selection, Insertion and Merge sorting techniques.

Selection sort:

```
def selection_sort(list):
    for i in range(len(list)):
        min_index = i
        for j in range(i + 1, len(list)):
            if list[min_index] > list[j]:
                min_index = j
        list[i], list[min_index] = list[min_index], list[i]
if__name__== '_main_':
    list = [3, 2, 1, 5, 4]
    selection_sort(list)
    print("Sorted lst: ", list)
```

Output:

```
Sorted lst: [1, 2, 3, 4, 5]
```

Bubble sort:

```
def bubble_sort(list):
    for i in range(len(list)):
        for j in range(0, len(list) - i - 1):
            if list[j] > list[j + 1]:
                list[j], list[j + 1] = list[j + 1], list[j]
if__name__== '_main_':
    list = [3, 2, 1, 5, 4]
    bubble_sort(list)
    print("Sorted list is: ", list)
```

Output:

```
Sorted list is: [1, 2, 3, 4, 5]
```

Insertion sort:

```
def insertion_sort(list):
    for i in range(1, len(list)):
        key = list[i]
        j = i - 1
        while j >= 0 and key < list[j]:
            list[j + 1] = list[j]
            j -= 1
        list[j + 1] = key
if__name__== '_main_':

    list = [3, 2, 1, 5, 4]
    insertion_sort(list)
    print("Sorted list is: ", list)
```

Output:

```
Sorted list is: [1, 2, 3, 4, 5]
```

Merge sort:

```
def mergeSort(myList):
    if len(myList) > 1:
        mid = len(myList) // 2
        left = myList[:mid]
        right = myList[mid:]
        mergeSort(left)
        mergeSort(right)
        i = 0
        j = 0
        k = 0

        while i < len(left) and j < len(right):
            if left[i] <= right[j]:
              myList[k] = left[i]
              i += 1
            else:
                myList[k] = right[j]
                j += 1
            k += 1
        while i < len(left):
            myList[k] = left[i]
            i += 1
            k += 1

        while j < len(right):
            myList[k]=right[j]
            j += 1
            k += 1
myList = [10,9,8,3,2,1,7,6,5,4]
mergeSort(myList)
print(myList)
```

Output:

```
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```