STAREX UNIVERSITY

PYTHON PRACTICAL FILE

submitted to:

Mrs. rekha khatana

submitted by :

anurag
1930801016
Computer sciense

CERTIFICATE

Name Anurag Class B.tech (CSE)

Roll No. 19308010116 Exam No. 3rd Sem

Institution starex university

We Certify this to be the meritorious work of the student in the Python Laboratory during the academic year 2020/2021

No. of practicals certified _____ out of 14

in the subject of Python

Examiner's Signature	Subject Teacher's Signature	Principal's Signature

Date _____ Stamp of Institution

(N.B. : The candidate is expected to retain his/her journal till he/she passes in the subject.)

INDEX

Sl. No.	Name of the Experiments	Date of Experiment	Page No.	Date of Submission	Remarks
1.	Program to print fibbonacci series		01		
2	Compute the GCD of two numbers.		02		
3	Write a program to swap two numbers		03		
4	Write a program to find the factorial of a number		04		
5.	Find the square root of a number (Newton's method)		05		
6	Write a program to check whether a number is Palindrome or not.		06		
7.	Exponentiation (Power of numb.)		07		

INDEX

Date : .....

Expt. No. ..... Page No. .....1.....

1. Program to print fibbonacci Series

```
def FibRecursion(n):
    if n <= 1:
        return n
    else:
        return (FibRecursion(n-1) + FibRecursion(n-2))

nterms = int(input("Enter the terms:")) # take input from
                                          the user

if nterms <= 0: # Check if the number is valid
    Print("Please enter a positive integer.")
else:
    Print("Fibonacci sequence:")
    For i in range(nterms):
        Print(FibRecursion(i))
```

Teacher's Signature : .....

Date : .....

Expt. No. .....

Page No. 2

2. Compute the GCD of two numbers.

```
def gcd(a, b):
    if (b == 0):
        return a
    else:
        return gcd(b, a % b)

a = int(input("Enter first number: "))
b = int(input("Enter second number: "))

GCD = gcd(a, b)

Print("The GCD of", a, b, "is", GCD, ".")
```

Teacher's Signature : .....

Date : .....

Expt. No. .....

Page No. .....3.....

3. Write a program to swap two numbers

```
num1 = input('Enter First Number:')
num2 = input('Enter Second Number:')

print("value of num1 before swapping:", num1)
print("value of num2 before swapping:", num2)


Temp = num1
num1 = num2
num2 = temp

print("value of num1 after swapping:", num1)
print("value of num2 after swapping:", num2)
```

Teacher's Signature : .....

Date : .....

Expt. No. ..... Page No. 4

4. Write a program to find the factorial of a number

```
def factorial(num):
    """This is a recursive function that calls
    itself to find the factorial of given number"""
    if num == 1:
        return num
    else:
        return num * factorial(num-1)

# We will find the factorial of this number
num = int(input("Enter a Number: "))

# if input number is negative then an error message
# elif the input number is 0 then display 1 as output
# else calculate the factorial by calling the user
defined function
if num < 0:
    Print("Factorial cannot be Found for negative num")
elif num == 0:
    Print("Factorial of 0 is 1")
else:
    Print("Factorial of", num, "is:", Factorial(num))
```

Teacher's Signature : .....

Date : .....

Expt. No. ..... Page No. 5

5. Find the Square root of a number (Newton's method)

```
def newton_method(num, num_iters = 100):
    a = float(num)

    for i in range(num_iters):

        num = 0.5 * (num + a/num)

    return num

a = int(input("Enter first number:"))

b = int(input("Enter second number:"))

Print("Square root of first number:", newton_method(a)

Print("Square root of second number:", newton_method(b)
```

Teacher's Signature : .....

Date : .....

Expt. No. .....

6. Write a program to check whether a number is palindrome or not.

```
n = int(input("enter number:"))

temp = n
rev = 0

while (n>0):
    dig = n % 10
    rev = rev * 10 + dig
    n = n / 10
if (temp == rev):
    Print("The number is a Palindrome!")
else:
    Print("The number isn't a palindrome!")
```

Teacher's Signature : .....

Date : .....

Expt. No. .....

Page No. .....7.....

7. Exponentiation (Power of a number)

```
number = int(input("Please enter any positive integer:"))
exponent = int(input("Please enter any exponent value:"))
Power = 1

For i in range(1, exponent+1):
    Power = Power * number

Print("The Result of {0} Power {1} = {2}".Farmat(number,
      exponent, Power))
```

Teacher's Signature : .....

Date : .....

Expt. No. .....

8. Find the maximum of a list of number

```
# Creating empty list
list = [ ]

# asking number of elements to put in list
num = int(input("Enter number of elements in list: "))

# iterating till num to append elements in list
For i in range(1, num+1):
    ele = int(input("Enter elements: "))
    list1.append(ele)

# Print maximum element
Print("Largest element is:", max(list1))
```

Teacher's Signature : .....

Date : .....

Expt. No. ..... Page No. 9

9. Linear Search and Binary search

```
# This is a linear search
def linear_search(alist, key):
    """Return index of key in list. Return -1 if key not present."""
    for i in range(len(alist)):
        if alist[i] == key:
            return i
    return -1

print("*** THIS IS A LINEAR SEARCH***")

alist = input('Enter the list of numbers:')

alist = alist.split()

alist = [int(x) for x in alist]

key = int(input('The number to search for:'))

index = linear_search(alist, key)
if index < 0:
    print('{} was not found.'.format(key))
else:
    print('{} was found at index {}.'.format(key, index))
```

Teacher's Signature : .....

Date : .....

Expt. No. .....


```
##########BINARY SEARCH
##########

def binary_search(alist, key):
    """ Search key in list [start....end-1]."""
    start = 0
    end = len(alist)
    while start < end
        mid = (start+end)//2
        if alist[mid] > key:
            end = mid
        elif alist[mid] < key:
            start = mid+1
    else:
        return mid
    return -1

print("\n*** THIS IS A BINARY SEARCH***")

alist = input('Enter the sorted list of numbers:')
alist = alist.split()
alist = [int(x) for x in alist
key = int(input('The number to search for:'))
```

Teacher's Signature : .....

Date : .....

Expt. No. .....


```
index = binary_search(alist, key)
if index < 0
    Print('{} was not found.'.format(key))
else:
    Print('{} was found at index {}.'.format(key,
        index))
```

Teacher's Signature : .....

10 Selection sort , Insertion Sort

```
def selection _ sort (alist):
  For i in range (0, len(alist) - 1):
    Smallest = i
    For j in range ( i + 1 , len(alist)):
      if alist [j) < alist [ Smallest]:
        smallest = j
    alist[i] , alist [smallest] = alist [smallest], alist[i]

Print ("*** THIS IS SELECTION SORT***")

alist = input ('Enter the list of numbers:'). split()
alist = [int (x) For x in alist]

Selection - sort (alist)

Print ('Sorted list:', end = '')
Print (alist)

#################### ###### INSERTION SORT
#####################
```

Date : .....

Expt. No. .....


```
def insertion_sort(alist):
    for i in range(1, len(alist)):
        temp = alist[i]
        j = i - 1
        while (j>=0 and temp < alist[j]):
            alist[j+1] = alist[j]
            j = j-1
        alist[j+1] = temp

print("\n*** THIS IS INSERTION SORT***")

alist = input('Enter the list of number:').split()
alist = [int(x) for x in alist]

insertion_sort(alist)

print('Sorted list:', end = '')
print(alist)
```

Teacher's Signature : .....

Date : .....

Expt. No. .....

Page No. ....14....

11 Merge sort

```
def merge_sort(alist, start, end):
    """sorts the list from indexes start to end-1
       inclusive."""
    if end - start > 1:
        mid = (start + end) // 2
        merge_sort(alist, start, mid)
        merge_sort(alist, mid, end)
        merge_list(alist, start, mid, end)

def merge_list(alist, start, mid, end):
    left = alist[start:mid]
    right = alist[mid:end]
    k = start
    i = 0
    j = 0

while(start + i < mid and mid + j < end):
    if (left[i] <= right[j]):
        alist[k] = left[i]
        i = i + 1
    else:
        alists[k] = right[j]
        j = j + 1
        k = k + 1
```

Teacher's Signature : .....

Date : .....

Expt. No. .....


```
if start + i < mid:
    while k < end
        alist[k] = left[i]
        i = i + 1
        k = k + 1
else:
    while k < end
        alist[k] = right[j]
        j = j + 1
        k = k + 1

alist = input('Enter the list of numbers:').split()
alist = [int(x) for x in alist]

merge_sort(alist, 0, len(alist))
print('Sorted list:', end = '')
print(alist)
```

Teacher's Signature : .....

Date : .....

Expt. No. .....

12 First n prime numbers

```
def print_primes_till_N(N):

    # Declare the variables
    i, J, Flag = 0, 0, 0;

    # Print display message
    Print("Prime numbers between 1 and", N, "are:");

# Traverse each number from 1 to N
# With the help of for loop
For i in range(1, N+1, 1):

    # Skip 0 and 1 as they are
    # With the help of for loop
    if (i == 1 or i == 0):
        continue;
    # Flag variable to tell
    # if i is prime or not
    Flag = 1;

    For J in range(2, ((i//2)+1), 1):
        if (i % J == 0):
            Flag = 0
            break;
```

Teacher's Signature : .....

Date : .....

Expt. No. .....


```
# Flag = 1 means i is prime
# and flag = 0 means i is not prime
if (flag == 1):
    Print(i, end = " ");

# Driver code
N = int(input('Enter range:'));
Print_Primes_till_N(N);
```

Teacher's Signature : .....

Date : .....

Expt. No. .....

13. Multiply matrices

```
MAX = 100
# Function to print Matrix
def printMatrix(M, rowsize, colsize):

    for i in range(rowsize):
        for j in range(colsize):
            print(M[i][j], end = " ")
    print()

# Function to multiply two matrices
# A[][] and B[][]
def multiplyMatrix(row1, col1, A,
                   row2, col2, B):

# matrix to store the result
C = [[0 for i in range(MAX)]
        for j in range(MAX)]

# Check if multiplication is possible
if (row2 != col1):
    print("Not Possible")
    return
```

Teacher's Signature : .....

Date : .....

Expt. No. .....

Page No. 19

```
# multiply the two
for i in range (row1):
  for j in range (col2):
    C[i][j] = 0
    for k in range (row2):
      C[i][j] += A[i][k] * B[k][j]

#Print the result
Print ("Resultant Matrix:")
PrintMatrix(C, row1, col2)

# Driver code
if __name__ == "__main__":

  A = [[0 for i in range(MAX)]
       for j in range(MAX)]
  B = [[0 for i in range [MAX]
       for j in rang [MAX]

#Read size of matrix A from user
row1 = int(input("Enter the number of row of first matrix:"))
Col1 = int(input("Enter the number of columns of first matrix:"))

#Read the elements of Matrix A From user
Print("Enter the elements of first matrix:");
```

Teacher's Signature : .....

Date : .....

Expt. No. ..... Page No. 20

```
For i in range (row1):
  For j in range (col1):
    A[i][j] = int(input("A[" + str(i) +
                "][" + str(j) + "]:"))

#Read size of matrix B from user
row2 = int(input("Enter the number of row of second matrix:"))
col2 = int(input("Enter the number of columns of secont matrix:"))

#Read the elements of matrix B from user
Print("Enter the elements of second matrix:");
For i in range (row2):
  For j in range (col2):
    B[i][j] = int(input("B[" + str(i) +
                "][" + str(j) + "]:"))
#Print the matrix A
Print("First Matrix:")
PrintMatrix(A, row1, col1)

#Print the matrix B
Print("Second matrix:")
PrintMatrix(B, row2, col2)

#Find the Product of the 2 matrices
multiplyMatrix(row1, col1, row2, col2, B)
```

Teacher's Signature : .....

14. Program that takes command line arguments (word count)

Code:

```
import sys
Print("Number of words", len(sys.argv)-1, "Words")
```

```
# ----- Process ------ #
# 1. Save this (eg. test.py)
# 2. open terminal and choose location where test.py is located
# 3. open test.py as "test.py sentence" eg: test.py hello how are you
# 4. Press enter and you will get your result
```