**Python Programming**

1. Develop programs to understand the control structures of python.

# Control Statement  
for char in 'Python':  
 if (char =='h'):  
 break  
 print("character:",char)  
  
# 2  
for char in 'python':  
 if (char == 'y'):  
 continue  
 print("charater:", char)

# Conditional Statements

num = 7  
if num < 0:  
 print("Negative number")  
elif num == 0:  
 print("Zero")  
else:  
 print("Positive number")

# odd even no  
n=int(input("Enter any number :"))  
if n%2==0:  
 print(n,'is an even number')  
else:  
 print(n,'is an odd number!')

1. Develop programs to learn different types of structures (list, dictionary, tuples) in python.

# List  
my\_list = ['AAA', 'BBB', 'CCC', 'DDD', 'EEE']  
# first item  
print(my\_list[0]) # p  
# third item  
print(my\_list[2]) # o  
# fifth item  
print(my\_list[4]) # e  
print(my\_list[-1])  
# elements from index 2 to index 4  
print(my\_list[2:5])  
  
# Tuple  
my\_tuple = ("ABCD", 1, 'EFGH', 2, 'Python')  
print(my\_tuple)  
print(my\_tuple[0])  
print(my\_tuple[3])  
print(my\_tuple[-1])  
print(my\_tuple[1:4])  
print(my\_tuple[:3])  
print(my\_tuple[3:])  
print((my\_tuple) + (3, 'Programming', 4))  
print(("ABCD",) \* 3)  
print(my\_tuple.index(2))  
  
# Dictionary  
my\_dict = {'name': 'Umesh', 'age': 21}  
print(my\_dict['name'])  
print(my\_dict.get('age'))  
# update value  
my\_dict['age'] = 22  
print(my\_dict)  
# add item  
my\_dict['address'] = 'Shirpur'  
print(my\_dict)  
# remove item  
print(my\_dict.pop('address'))

1. Develop programs to learn concept of functions scoping, recursion and list mutability.

# Factorial Number.  
def factorial(n):  
 if n == 0 or n == 1:  
 return 1  
 else:  
 #Using recursion  
 return n \* factorial(n - 1)  
n = int(input("Enter any number: "))  
fact = factorial(n)  
print(fact)

1. Develop programs to understand object oriented programming using python.

class A():  
 def \_\_init\_\_(self, rno):  
 self.rno = rno  
 def f1(self,name):  
 return {self.rno, name}  
class main():  
 a1 = A(15)  
 a2 = a1.f1("abc")  
 print(a2)  
main()

1. Develop programs to learn regular expressions using python.

import re  
pattern = r"ABCD"  
if re.match(pattern,"ABCD"):  
 print("match")  
else:  
 print("no match")

1. Demonstrate the concept of exception handling using try/except/else Statement, Unified try/except/finally, try/finally Statement, raise Statement, assert Statement, catch multiple specific exceptions.

def umesh():  
 try:  
 print("Hello")  
 print(10)  
 except ZeroDivisionError:  
 print("ZeroDivisionError")  
 return 10  
 finally:  
 print("This code will run")  
 return 5  
print(umesh())

1. Demonstrate implementation of the Anonymous Function Lambda.

# Program to show the use of lambda functions  
double = lambda x: x \* 2  
print(double(5))  
  
# Program to filter out only the even items from a list  
my\_list = [1, 5, 4, 6, 8, 11, 3, 12]  
new\_list = list(filter(lambda x: (x%2 == 0) , my\_list))  
print(new\_list)  
  
# Program to double each item in a list using map()  
my\_list = [1, 5, 4, 6, 8, 11, 3, 12]  
new\_list = list(map(lambda x: x \* 2 , my\_list))  
print(new\_list)

1. Demonstrate image insertion in python.

from tkinter import \*  
class ImageDemo(Frame):  
 def \_\_init\_\_(self):  
 Frame.\_\_init\_\_(self)  
 self.master.title("Image Demo")  
 self.grid()  
 self.\_image=PhotoImage(file="1dpmw.gif")  
 self.\_imageLabel=Label(self,image=self.\_image)  
 self.\_imageLabel.grid()  
 self.\_textLabel=Label(self,text="umesh")  
 self.\_textLabel.grid()  
def main():  
 ImageDemo() .mainloop()  
main()

1. Armstrong Number.

num = int(input("Enter a number: "))  
sum = 0  
temp = num  
while temp > 0:  
 digit = temp % 10  
 sum += digit \*\* 3  
 temp //= 10  
if num == sum:  
 print(num,"is an Armstrong number")  
else:  
 print(num,"is not an Armstrong number")

1. Prime Number.

num = int(input("Enter a number: "))  
flag = 0  
if num > 1:  
 for i in range(2, num):  
 if (num % i) == 0:  
 flag = 1  
 break  
if flag:  
 print(num, "is not a prime number")  
else:  
 print(num, "is a prime number")

1. Fibonacci series.

n = int(input("Enter the number :"))  
n1, n2 = 0, 1  
count = 0  
if n <= 0:  
 print("Please enter a positive integer")  
elif n == 1:  
 print("Fibonacci sequence upto",n,":")  
 print(n1)  
else:  
 print("Fibonacci sequence:")  
 while count < n:  
 print(n1)  
 n3 = n1 + n2  
 n1 = n2  
 n2 = n3  
 count += 1

1. Palindrome Number.

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!")

1. Calculator.

def add(x, y):  
 return x + y  
def sub(x, y):  
 return x - y  
def mult(x, y):  
 return x \* y  
def div(x, y):  
 return x / y  
print("Select operation")  
print("1.Add")  
print("2.Sub")  
print("3.Mult")  
print("4.Div")  
while True:  
 choice = input("Enter ch(1/2/3/4):")  
 if choice in ('1', '2', '3', '4'):  
 num1 = float(input("Enter first number: "))  
 num2 = float(input("Enter second number: "))  
 if choice == '1':  
 print(num1, "+", num2, "=", add(num1, num2))  
 elif choice == '2':  
 print(num1, "-", num2, "=", sub(num1, num2))  
 elif choice == '3':  
 print(num1, "\*", num2, "=", mult(num1, num2))  
 elif choice == '4':  
 print(num1, "/", num2, "=", div(num1, num2))  
 next\_calculation = input("Let's do next calculation? (yes/no): ")  
 if next\_calculation == "no":  
 break  
 else:  
 print("Invalid Input")

1. Binary search.

def binary(arr, x, low, high):  
 if low > high:  
 return -1  
 mid = (low + high) // 2  
  
 if x == arr[mid]:  
 return mid  
 elif x < arr[mid]:  
 high = mid - 1  
 return binary(arr, x, low, high)  
 else:  
 low = mid + 1  
 return binary(arr, x, low, high)  
  
  
arr = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]  
  
x = int(input("Enter x :"))  
  
print()  
res = binary(arr, x, 0, len(arr) - 1)  
  
if res >= 0:  
 print(x, "FOUND at index", res)  
else:  
 print("NOT FOUND")

1. Linear Search

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