Sub : PYTHON PROGRAMMING MANUAL

Class : BCA VI SEM

Name : Merline Fernandes

1. **a. Write a python program to print “Hello Python”**

```
print("Hello Python")
```

Output :

```
Hello Python
```

1. **b. Write a python program to do Arithmetic Operations.**

```
n1=input("Enter First number : ")
n2=input("Enter second Number : ")
sum=int(n1)+int(n2)
sub=int(n1)-int(n2)
mul=int(n1)*int(n2)
div=int(n1)//int(n2)
print("Aritmetic operations")
print('{0} + {1} = {2}'.format(n1,n2,sum))
print('{0} - {1} = {2}'.format(n1,n2,sub))
print('{0} x {1} = {2}'.format(n1,n2,mul))
print('{0} / {1} = {2}'.format(n1,n2,div))
```

Output :

```
Enter First number : 20
Enter second Number : 10
Arithmetic operations
20 + 10 = 30
20 - 10 = 10
20 x 10 = 200
20 / 10 = 2
```

2. **Write a python program to find the area of the Triangle.**

```
b=float(input('Enter the base of Triangle : '))
h=float(input('Enter the height of Triangle : '))
area=b*h
print('Area of Triangle = %.2f' %(area))
```

Output :

```
Enter the base of Triangle : 3
Enter the height of Triangle : 4
Area of Triangle = 12.00
```

3. **Write a python program to solve quadratic equation**

```
import cmath

a=float(input('Enter Value for a : '))
b=float(input('Enter value for b : '))
c=float(input('Enter value for c : '))
d=(b**2)-(4*a*c)
sol1=(-b-cmath.sqrt(d))/(2*a)
sol2=(-b+cmath.sqrt(d))/(2*a)
print('The Solution are {0} and {1}'.format(sol1,sol2))
```

Output :

```
Enter Value for a : 2
Enter value for b : 3
Enter value for c : 4
```

4. Write a python program to swap two variables.

```
a=int(input('Enter value for A : '))
b=int(input('Enter value for B : '))
print('Before Swapping A = %d & B = %d' %(a,b))
t=a
a=b
b=t
print('Before Swapping A = %d & B = %d' %(a,b))
```

Output :

```
Enter value for A : 11
Enter value for B : 28
Before Swapping A = 11 & B = 28
Before Swapping A = 28 & B = 11
```

5. Write a python program to convert Celsius to Fahrenheit.

```
c=float(input('Enter Temperature in Celsius : '))

f=(c*1.8)+32
print('%.2f degree celsius = %.2f Fahrenheit' %(c,f))
```

Output :

```
Enter Temperature in Celsius : 36

36.00 degree celsius = 96.80 Fahrenheit
```

6. **Write a python program to check if a number is odd or even.**

```
n=int(input('Enter an Integer : '))

if (n % 2) == 0 :
  print('%d is Even Number ' %(n))
else :
   print('%d is Odd Number '%(n))
```

Output :

```
Run 1 :

Enter an Integer : 4

4 is Even Number.
```

```
Run 1 :

Enter an Integer : 3

3 is Odd Number.
```

7. **Write a python program to print all prime number in a interval.**

```
l=int(input('Enter the lower limit : '))
u=int(input('Enter the upper limit : '))
print('Prime Number between %d to %d are ' %(l,u))
for num in range(l, u + 1):
   if num > 1:
       for i in range(2, num):
           if (num % i) == 0:
               break
       else:
           print(num)
```

Output :

```
Enter the lower limit : 2

Enter the upper limit : 10

Prime Number between 2 to 10 are

2
3
5
7
```

8. Write a python program to find the factorial of a number.

```
n=int(input('Enter the range : '))
fact=1
for i in range(1,n+1):
    fact=fact*i
print('The Factorial of %d is %d' %(n,fact))
```

Output :

```
Enter the range : 5
The Factorial of 5 is 120
```

9. Write a python program to Display multiplication table.

```
n=int(input('Enter a number : '))
for i in range(1,11):
    t=n*i
    print('%d x %d = %d' %(n,i,t))
```

Output :

```
Enter a number : 5
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50
```

10. Write a python program to multiply two matrices.

```
def matrix_multiply(a, b):
    # Check if the matrices can be multiplied
    if len(a[0]) != len(b):
        return "Matrices cannot be multiplied"

    # Create an empty matrix to store the result
    result = [[0 for j in range(len(b[0]))] for i in range(len(a))]

    # Multiply the matrices
    for i in range(len(a)):
        for j in range(len(b[0])):
            for k in range(len(b)):
                result[i][j] += a[i][k] * b[k][j]

    return result

# Get the dimensions of the matrices from the user
rows1 = int(input("Enter the number of rows of the first matrix: "))
cols1 = int(input("Enter the number of columns of the first matrix: "))
rows2 = int(input("Enter the number of rows of the second matrix: "))
cols2 = int(input("Enter the number of columns of the second matrix: "))

matrix1 = []
matrix2 = []

print("Enter the values of the first matrix: ")
for i in range(rows1):
    row = []
    for j in range(cols1):
        row.append(int(input()))
    matrix1.append(row)

print("Enter the values of the second matrix: ")
for i in range(rows2):
    row = []
    for j in range(cols2):
        row.append(int(input()))
    matrix2.append(row)
    # Multiply the matrices
result = matrix_multiply(matrix1, matrix2)

# Print the result
if result == "Matrices cannot be multiplied":
    print(result)
else:
    print("The result is:")
    for row in result:
        print(row)
```

Output :

```
Enter the number of rows of the first matrix: 2
Enter the number of columns of the first matrix: 2
Enter the number of rows of the second matrix: 2
Enter the number of columns of the second matrix: 2
Enter the values of the first matrix:
1
2
3
4
Enter the values of the second matrix:
4
3
2
1
The result is:
[8, 5]
[20, 13]
```

11. Write a python program to find LCM and GCD using function.

```
# Function to find the GCD of two numbers
def gcd(a, b):
    while b:
        a, b = b, a % b
    return a

# Function to find the LCM of two numbers
def lcm(a, b):
    return (a * b) // gcd(a, b)

# Get the numbers from the user
num1 = int(input("Enter the first number: "))
num2 = int(input("Enter the second number: "))

# Find the GCD and LCM
gcd_result = gcd(num1, num2)
lcm_result = lcm(num1, num2)

# Print the results
print("The GCD of", num1, "and", num2, "is", gcd_result)
print("The LCM of", num1, "and", num2, "is", lcm_result)
```

Output :

Run 1 :

```
Enter the first number: 2
Enter the second number: 4
The GCD of 2 and 4 is 2
The LCM of 2 and 4 is 4
```

Run 2 :

```
Enter the first number: 12
Enter the second number: 14
The GCD of 12 and 14 is 2
The LCM of 12 and 14 is 84
```

12. **Write a python program to read a word and print the number of letters, vowels in the word.**

```
# Function to count the number of letters and vowels in a word
def count_letters_vowels(word):
    letters = 0
    vowels = 0
    for letter in word:
        if letter.isalpha():
            letters += 1
            if letter.lower() in ['a', 'e', 'i', 'o', 'u']:
                vowels += 1
    return letters, vowels

# Get the word from the user
word = input("Enter a word: ")

# Count the number of letters and vowels in the word
num_letters, num_vowels = count_letters_vowels(word)

# Print the results
print("The word", word, "has", num_letters, "letters and", num_vowels,
"vowels.")
```

Output :

```
Enter a word: Python Programming

The word Python Programming has 17 letters and 4 vowels.
```

13. **Write a python to input an array of n number and find separately the sum of positive numbers and negative number.**

```
# Get the number of elements in the array from the user
n = int(input("Enter the number of elements in the array: "))

# Initialize variables to store the sum of positive and negative numbers
positive_sum = 0
negative_sum = 0

# Get the elements of the array from the user
arr = []
for i in range(n):
    arr.append(int(input("Enter element " + str(i+1) + ": ")))

# Loop through the array and add up the positive and negative numbers
for num in arr:
    if num > 0:
        positive_sum += num
    elif num < 0:
        negative_sum += num

# Print the results
print("The sum of positive numbers is:", positive_sum)
print("The sum of negative numbers is:", negative_sum)
```

Output :

```
Enter the number of elements in the array: 6
Enter element 1: 11
Enter element 2: 28
Enter element 3: -2
Enter element 4: -1
Enter element 5: 6
Enter element 6: 7
The sum of positive numbers is: 52
The sum of negative numbers is: -3
```

14. Write a python program to search an element using linear search equation.

```python
# Function to perform linear search on an array
def linear_search(arr, n, x):
    for i in range(n):
        if arr[i] == x:
            return i
    return -1

# Get the number of elements in the array from the user
n = int(input("Enter the number of elements in the array: "))

# Get the elements of the array from the user
arr = []
for i in range(n):
    arr.append(int(input("Enter element " + str(i+1) + ": ")))

# Get the element to be searched from the user
x = int(input("Enter the element to be searched: "))

# Call the linear_search function to search for the element in the array
result = linear_search(arr, n, x)

# Print the result
if result == -1:
    print("Element", x, "not found in the array.")
else:
    print("Element", x, "found at position", result+1)
```

Output :

Run 1:

```
Enter the number of elements in the array: 5
Enter element 1: 30
Enter element 2: 28
Enter element 3: 11
Enter element 4: 50
Enter element 5: 7
Enter the element to be searched: 11
Element 11 found at position 3
```

Run 1:

```
Enter the number of elements in the array: 6
Enter element 1: 70
Enter element 2: 29
Enter element 3: 54
Enter element 4: 72
Enter element 5: 95
Enter element 6: 80
Enter the element to be searched: 7
Element 7 not found in the array.
```

15. Write a python program to search an element using binary search.

```python
# Function to perform binary search on a sorted array
def binary_search(arr, left, right, x):
    if right >= left:
        mid = (left + right) // 2
        if arr[mid] == x:
            return mid
        elif arr[mid] > x:
            return binary_search(arr, left, mid-1, x)
        else:
            return binary_search(arr, mid+1, right, x)
    else:
        return -1

# Get the number of elements in the array from the user
n = int(input("Enter the number of elements in the array: "))

# Get the elements of the array from the user
arr = []
for i in range(n):
    arr.append(int(input("Enter element " + str(i+1) + ": ")))

# Sort the array
arr.sort()

# Get the element to be searched from the user
x = int(input("Enter the element to be searched: "))

# Call the binary_search function to search for the element in the array
result = binary_search(arr, 0, n-1, x)

# Print the result
if result == -1:
    print("Element", x, "not found in the array.")
else:
    print("Element", x, "found at index", result)
```

Output :

Run 1:

```
Enter the number of elements in the array: 5
Enter element 1: 2
Enter element 2: 4
Enter element 3: 5
Enter element 4: 8
Enter element 5: 9
Enter the element to be searched: 5
Element 5 found at index 2
```

Run 2:

```
Enter the number of elements in the array: 5
Enter element 1: 12
Enter element 2: 14
Enter element 3: 26
Enter element 4: 38
Enter element 5: 49
Enter the element to be searched: 19
Element 19 not found in the array.
```

16. Write a python program to insert a number in sorted array.

```
# Function to insert a number in a sorted array while maintaining the sorted
order
def insert_in_sorted_array(arr, n, x):
    i = 0
    while i < n and arr[i] < x:
        i += 1
    arr.insert(i, x)
    return arr

# Get the number of elements in the array from the user
n = int(input("Enter the number of elements in the array: "))

# Get the elements of the sorted array from the user
arr = []
for i in range(n):
    arr.append(int(input("Enter element " + str(i+1) + " (in sorted order): ")))

# Get the number to be inserted from the user
x = int(input("Enter the number to be inserted: "))

# Call the insert_in_sorted_array function to insert the number in the sorted
array
arr = insert_in_sorted_array(arr, n, x)

# Print the updated sorted array
print("Updated sorted array: ", end="")
for i in range(n+1):
    print(arr[i], end=" ")
```

Output :

Run 1 :

```
Enter the number of elements in the array: 4
Enter element 1 (in sorted order): 10
Enter element 2 (in sorted order): 20
Enter element 3 (in sorted order): 40
Enter element 4 (in sorted order): 50
Enter the number to be inserted: 30
Updated sorted array: 10 20 30 40 50
```

Run 2 :

```
Enter the number of elements in the array: 5
Enter element 1 (in sorted order): 20
Enter element 2 (in sorted order): 40
Enter element 3 (in sorted order): 50
Enter element 4 (in sorted order): 70
Enter element 5 (in sorted order): 80
Enter the number to be inserted: 10
Updated sorted array: 10 20 40 50 70 80
```

17. Write a python program to stimulate stack operation.

```
# Define the Stack class
class Stack:
    def __init__(self):
        self.stack = []

    # Function to push an element into the stack
    def push(self, element):
        self.stack.append(element)
        print("Pushed element", element, "into the stack")

    # Function to pop an element from the stack
    def pop(self):
        if not self.is_empty():
            element = self.stack.pop()
            print("Popped element", element, "from the stack")
            return element
        else:
            print("Stack is empty")

    # Function to check if the stack is empty
    def is_empty(self):
        return len(self.stack) == 0

    # Function to print the stack
    def print_stack(self):
        print("Stack:", self.stack)

# Create a new stack
stack = Stack()

# Loop for stack operations
while True:
    print("\nSTACK OPERATIONS:")
    print("1. Push an element")
    print("2. Pop an element")
    print("3. Print the stack")
    print("4. Exit")

    # Get the choice from the user
    choice = int(input("Enter your choice (1-4): "))

    # Perform the selected operation
    if choice == 1:
        element = int(input("Enter the element to be pushed: "))
        stack.push(element)
    elif choice == 2:
        stack.pop()
    elif choice == 3:
        stack.print_stack()
    elif choice == 4:
        break
    else:
       print("Invalid choice. Please enter a number between 1 and 4.")
```

Output :

```
STACK OPERATIONS:
1. Push an element
2. Pop an element
3. Print the stack
4. Exit
Enter your choice (1-4): 1
Enter the element to be pushed: 11
Pushed element 11 into the stack

STACK OPERATIONS:
1. Push an element
2. Pop an element
3. Print the stack
4. Exit
Enter your choice (1-4): 1
Enter the element to be pushed: 28
Pushed element 28 into the stack

STACK OPERATIONS:
1. Push an element
2. Pop an element
3. Print the stack
4. Exit
Enter your choice (1-4): 1
Enter the element to be pushed: 7
Pushed element 7 into the stack

STACK OPERATIONS:
1. Push an element
2. Pop an element
3. Print the stack
4. Exit
Enter your choice (1-4): 3
Stack: [11, 28, 7]

STACK OPERATIONS:
1. Push an element
2. Pop an element
3. Print the stack
4. Exit
Enter your choice (1-4): 2
Popped element 7 from the stack

STACK OPERATIONS:
1. Push an element
2. Pop an element
3. Print the stack
4. Exit
Enter your choice (1-4): 2
Popped element 28 from the stack
```

```
STACK OPERATIONS:
1. Push an element
2. Pop an element
3. Print the stack
4. Exit
Enter your choice (1-4): 2
Popped element 11 from the stack

STACK OPERATIONS:
1. Push an element
2. Pop an element
3. Print the stack
4. Exit
Enter your choice (1-4): 2
Stack is empty

STACK OPERATIONS:
1. Push an element
2. Pop an element
3. Print the stack
4. Exit
Enter your choice (1-4): 4
PS D:\ps programs>
```

18. Write a python program to draw shapes & GUI controls.

```
import tkinter as tk

# Create a window
window = tk.Tk()
window.geometry("400x400")
window.title("Drawing Shapes")

# Create a canvas to draw on
canvas = tk.Canvas(window, width=300, height=300)
canvas.pack()

# Draw a rectangle
canvas.create_rectangle(50, 50, 150, 150, fill="blue")

# Draw an oval
canvas.create_oval(180, 100, 250, 200, fill="red")

# Draw a line
canvas.create_line(50, 250, 250, 250, width=5, fill="green")

# Create a label
label = tk.Label(window, text="Enter your name:")
label.pack()

# Create a text input
input = tk.Entry(window)
input.pack()

# Create a button
button = tk.Button(window, text="Submit")
button.pack()

# Run the window
window.mainloop()
```

Output :

19. Write a python program to using the build-in methods of the string list and dictionary classes.

```
# String methods
string = input("Enter a string: ")
print("Upper case:", string.upper())
print("Lower case:", string.lower())
sub = input("Enter a substring to replace: ")
new_sub = input("Enter a new substring: ")
print("Replaced string:", string.replace(sub, new_sub))

# List methods
my_list = []
n = int(input("Enter the length of the list: "))
for i in range(n):
    item = input(f"Enter element {i+1}: ")
    my_list.append(item)
print("List length:", len(my_list))
item = input("Enter an element to append to the list: ")
my_list.append(item)
print("Appended list:", my_list)
item = my_list.pop()
print("Popped item:", item)
print("Updated list:", my_list)

# Dictionary methods
my_dict = {}
n = int(input("Enter the number of key-value pairs: "))
for i in range(n):
    key = input(f"Enter key {i+1}: ")
    value = input(f"Enter value {i+1}: ")
    my_dict[key] = value
print("Keys:", my_dict.keys())
print("Values:", my_dict.values())
key = input("Enter a key to update: ")
value = input("Enter a new value: ")
my_dict[key] = value
print("Updated dictionary:", my_dict)
```

Output :

```
Enter a string: Python
Upper case: PYTHON
Lower case: python
Enter a substring to replace: Py
Enter a new substring: Mera
Replaced string: Merathon
Enter the length of the list: 4
Enter element 1: 11
Enter element 2: 28
Enter element 3: 19
Enter element 4: 7
List length: 4
Enter an element to append to the list: 14
Appended list: ['11', '28', '19', '7', '14']
Popped item: 14
Updated list: ['11', '28', '19', '7']
Enter the number of key-value pairs: 2
Enter key 1: Name
Enter value 1: Doy
Enter key 2: Age
Enter value 2: 28
Keys: dict_keys(['Name', 'Age'])
Values: dict_values(['Doy', '28'])
Enter a key to update: Address
Enter a new value: Karwar
Updated dictionary: {'Name': 'Doy', 'Age': '28', 'Address': 'Karwar'}
```

20. Write a python program to demonstrate exception handling.

```
try:
    dividend = int(input("Enter the dividend: "))
    divisor = int(input("Enter the divisor: "))
    quotient = dividend / divisor
    print("Quotient is:", quotient)
except ValueError:
    print("Please enter only integer values.")
except ZeroDivisionError:
    print("Cannot divide by zero.")
```

Output :

```
Run 1:

Enter the dividend: 20
Enter the divisor: 2
Quotient is: 10.0

Run 2:

Enter the dividend: 20
Enter the divisor: 0
Cannot divide by zero.

Run 3 :

Enter the dividend: 100
Enter the divisor: xyz
Please enter only integer values.
```