Basic Programs

1.

```
#Q1->WAP Program to add two entered integer values.
a=int(input("Enter: "))
b=int(input("Enter: "))
print("Sum:",a+b)
```

2.

```
#Q2-> Program to subtract two entered integer values.
a=int(input("Enter: "))
b=int(input("Enter: "))
print("Difference:",a-b)
```

3.

```
#Q3->Program to multiply two entered integer values.
a=int(input("Enter: "))
b=int(input("Enter: "))
print("Multiply:",a*b)
```

4.

```
#Q4->Program to input two integer values and calculate first number raised to the power second number
a=int(input("Enter: "))
b=int(input("Enter: "))
print("Number raised to power:",a**b)
```

5.

```
#Q5-> Program to find the area and perimeter of a rectangle.
l=int(input("Enter The length: "))
b=int(input("Enter the breadth: "))
print("Area:",l*b)
print("Perimeter:",2*(l+b))
```

If-Else Programs

1.

```
#Q1->Program to find whether an input number is even or odd.
a=int(input('Enter number: '))
if a%2==0:
    print("Even")
else:
    print("Odd")
```

2.

```
#Q2->WAP to find maximum between two numbers.
a=int(input('Enter 1st Number: '))
b=int(input('Enter 2nd Number: '))
print("Max: ",end='')
if a>b:
    print(a)
else:
    print(b)
```

3.

```
#Q3->WAP to find maximum between three numbers.
a=int(input('Enter 1st number: '))
b=int(input('Enter 2nd number: '))
c=int(input('Enter 3rd number: '))
if b<a>c:
    print(a)
elif c<b>a:
    print(b)
elif b<c>a:
    print(c)
```

4.

```
#Q4->WAP to check whether a year is leap year or not.
a=int(input('Enter year: '))
if a%400==0 or (a%4==0 and a%100==0):
    print("Leap Year")
else:
    print("Not a Leap Year")
```

5.

```
#Q5->WAP to check whether a character is alphabet or not.
a=input('Enter: ')
if a.isalpha():
    print("Alphabet")
else:
    print("Not a alphabet")
```

Loop Programs

1.

```
#Q1->WAP to find a number is prime or not.
s=int(input("entre the number"))
c=0
for i in range(1,s+1):
    if s%i==0:
        c=c+1
if c==2:
    print("Number is prime")
else:
    print("Number is not prime")
```

2.

```
#Q2->Find whether given number is armstrong or not.
num = input('Enter the positive integer ')
s = 0
x = len(num)
for ch in num:
    s += int(ch)**x
if s == int(num):
    print(f'Number {num} is Armstrong')
else:
    print(f'Number {num} is not Armstrong')
```

3.

```
#Q3->Find the factorial of given number.
n = int(input('enter the number '))
fact = 1
for i in range(n, 0, -1):
    fact = fact * i
print('Factorial is ', fact)
```

4.

```
#Q4->Write a python program to find the lcm of two number.
num1 = int(input('enter the positive integer'))
num2 = int(input('enter the positive integer '))
lcm = num2 if num1 < num2 else num1
while 1:
    if lcm % num1 == 0 and lcm % num2 == 0:
        break
    lcm += 1
print(f'LCM of two number is {lcm}')
```

5

```
#Q5->Find whether a given number is perfect or not.
n=int(input("enter the number"))
s = 0
for i in  range(1,n):
    if(n%i==0):
        s += i
if s == n:
    print("perfect number ")
else:
    print("not a perfect number ")
```

List and Tuple Programs

1.

```
#Q1->WAP to remove even number from a list.
l=[1,2,3,4,5,6]
for i in l:
    if i%2==0:
        l.remove(i)
print(l)
```

2.

```
#Q2->WAP to remove odd number from a list.
l=[1,2,3,4,5,6]
for i in l:
    if i%2!=0:
        l.remove(i)
print(l)
```

3.

```
#Q3->WAP to find sum of elements of tuple.
t=(1,2,3,4,5,6)
s=0
for i in t:
    s+=i
print("Sum:",s)
```

4.

```
#Q4->WAP to print square of elements of tuple.
t=(1,2,3,4,5,6)
for i in t:
    print(i**2)
```

5.

```
#Q5->WAP to print square root of elements of tuple.
t=(1,2,3,4,5,6)
for i in t:
    print(i**0.5)
```

Dictionary Programs

1.

```
#Q1->WAP to sort (ascending and descending) a dictionary by value.
import operator
d = {1: 2, 3: 4, 4: 3, 2: 1, 0: 0}
print('Original dictionary : ',d)
sorted_d = sorted(d.items(), key=operator.itemgetter(1))
print('Dictionary in ascending order by value : ',sorted_d)
sorted_d = dict( sorted(d.items(), key=operator.itemgetter(1),reverse=True))
print('Dictionary in descending order by value : ',sorted_d)
```

2.

```
#Q2->WAP to check whether a given key already exists in a dictionary.
d = {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}
def is_key_present(x):
  if x in d:
      print('Key is present in the dictionary')
  else:
      print('Key is not present in the dictionary')
is_key_present(5)
```

3.

```
#Q3->WAP to generate dictionary in the form (x, x*x).
n=int(input("Input a number "))
d = dict()
for x in range(1,n+1):
    d[x]=x*x
print(d)
```

4.

```
#Q4->WAP to print a dictionary where the keys are numbers 1 to 15 and the values are square of keys.
d=dict()
for x in range(1,16):
    d[x]=x**2
print(d)
```

5.

```
#Q5->WAP to remove a key from a dictionary.
myDict = {'a':1,'b':2,'c':3,'d':4}
print(myDict)
if 'a' in myDict:
    del myDict['a']
print(myDict)
```

Function Programs

1.

```
#Q1->Check whether the number is prim or not using function.
def test_prime(n):
    if (n==1):
        return False
    elif (n==2):
        return True
    else:
        for x in range(2,n):
            if(n % x==0):
                return False
        return True
print(test_prime(9))
```

2.

```
#Q2->Find the factorial using Function.
def factorial(n):
    if n == 0:
        return 1
    return n * factorial(n-1)
num = 5
print("Factorial of", num, "is",
factorial(num))
```

3.

```
#Q3->Check whether a number falls in given range or not.
def test_range(n):
    if n in range(3,9):
        print( " %s is in the range"%str(n))
    else :
        print("The number is outside the given range.")
test_range(5)
```

4.

```
#Q4->Copy unique elements of the first list using function.
def unique_list(l):
  x = []
  for a in l:
    if a not in x:
      x.append(a)
  return x
print(unique_list([1,2,3,3,3,3,4,5]))
```

5.

```
#Q5->Write a function to sum all the numbers in a list.
def sum(numbers):
    total = 0
    for x in numbers:
        total += x
    return total
print(sum((8, 2, 3, 0, 7)))
```

File Handling Programs

1.

```
#Q1->WAP to read first n lines of a file.
def file_read_from_head(fname, nlines):
        from itertools import islice
        with open(fname) as f:
                for line in islice(f, nlines):
                        print(line)
file_read_from_head('test.txt',2)
```

2.

```
#Q2->WAP to append text to a file and display the text.
def file_read(fname):
        from itertools import islice
        with open(fname, "w") as myfile:
                myfile.write("Python Exercises\n")
                myfile.write("Java Exercises")
        txt = open(fname)
        print(txt.read())
file_read('abc.txt')
```

3.

```
#Q3->WAP to read last n lines of a file.
import sys
import os
def file_read_from_tail(fname,lines):
        bufsize = 8192
        fsize = os.stat(fname).st_size
        iter = 0
        with open(fname) as f:
                if bufsize > fsize:
                        bufsize = fsize-1
                        data = []
                        while True:
                                iter +=1
                                f.seek(fsize-bufsize*iter)
                                data.extend(f.readlines())
                                if len(data) >= lines or f.tell() == 0:
                                        print(''.join(data[-lines:]))
                                        break
file_read_from_tail('test.txt',2)
```

4.

```
#Q4->WAP to write the even numbers into a file with user given number range.
f = open('even_numberfile.txt', 'w')
start, stop = [1, 101]
p = 1
for i in range(start, stop):
    if i % 2 == 0:
        f.write(f"{i}\t")
        p += 1
        if p == 11:
            f.write('\n')
            p = 1
f.close()
```

5.

```
#Q5->WAP to read an entire text file.
def file_read(fname):
        txt = open(fname)
        print(txt.read())
file_read('test.txt')
```

Numpy Module Programs

1.

```
#Q1->WAP to repeat the elements three times of a given string array using numpy.
import numpy as np
x1 = np.array(['Python', 'PHP', 'Java', 'C++'], dtype=np.str)
print("Original Array:")
print(x1)
new_array = np.char.multiply(x1, 3)
print("New array:")
print(new_array)
```

2.

```
#Q2->WAP to remove the leading and trailing whitespaces of all the elements of a given array.
import numpy as np
x = np.array([' python exercises ', ' PHP  ', ' java  ', '  C++'], dtype=np.str)
print("Original Array:")
print(x)
lstripped_char = np.char.lstrip(x)
print("\nRemove the leading whitespaces : ", lstripped_char)
```

3.

```
#Q3->WAP to encode all the elements of a given array in cp500 and decode it again.
import numpy as np
x = np.array(['python exercises', 'PHP', 'java', 'C++'], dtype=np.str)
print("Original Array:")
print(x)
encoded_char = np.char.encode(x, 'cp500')
decoded_char = np.char.decode(encoded_char,'cp500')
print("\nencoded =", encoded_char)
print("decoded =", decoded_char)
```

4.

```
#Q4->WAP to split the element of a given array with spaces.
import numpy as np
x = np.array(['Python PHP Java C++'], dtype=np.str)
print("Original Array:")
print(x)
r = np.char.split(x)
print("\nSplit the element of the said array with spaces: ")
print(r)
```

5.

```
#Q5->WAP to test whether none of the elements of a given array is zero.
import numpy as np
x = np.array([1, 2, 3, 4])
print("Original array:")
print(x)
print("Test if none of the elements of the said array is zero:")
print(np.all(x))
x = np.array([0, 1, 2, 3])
print("Original array:")
print(x)
print("Test if none of the elements of the said array is zero:")
print(np.all(x))
```

Random Module Programs

1.

```
#Q1->Roll dice in such a way that every time you get the same number.
import random
dice = [1, 2, 3, 4, 5, 6]
print("Randomly selecting same number of a dice")
for i in range(5):
    random.seed(25)
    print(random.choice(dice))
```

2.

```
#Q2->WAP to Generate 6 digit random secure OTP.
import secrets
secretsGenerator = secrets.SystemRandom()
print("Generating 6 digit random OTP")
otp = secretsGenerator.randrange(100000, 999999)
print("Secure random OTP is ", otp)
```

3.

```
#Q3->WAP to Calculate multiplication of two random float numbers.
import random
num1  = random.random()
print("First Random float is ", num1)
num2 = random.uniform(9.5, 99.5)
print("Second Random float is ", num1)
num3 = num1 * num2
print("Multiplication is ", num3)
```

4.

```
#Q4->WAP to Pick a random character from a given String.
import random
name = 'pynative'
char = random.choice(name)
print("random char is ", char)
```

5.

```
#Q5->WAP to Generate random String of length 5.
import random
import string
def randomString(stringLength):
    """Generate a random string of 5 charcters"""
    letters = string.ascii_letters
    return ''.join(random.choice(letters) for i in range(stringLength))
print ("Random String is ", randomString(5))
```

Exception Handling Programs

1.

```
#Q1->WAP to depict else clause with try-except.
def AbyB(a , b):
    try:
        c = ((a+b) / (a-b))
    except ZeroDivisionError:
        print ("a/b result in 0")
    else:
        print (c)
AbyB(2.0, 3.0)
AbyB(3.0, 3.0)
```

2.

```
#Q2->WAP to handle simple runtime error.
a = [1, 2, 3]
try:
    print ("Second element = %d" %(a[1]))
    print ("Fourth element = %d" %(a[3]))
except:
    print ("An error occurred")
```

3.

```
#Q3->WAP to demonstrate finally.
try:
    k = 5//0
    print(k)

except ZeroDivisionError:
    print("Can't divide by zero")

finally:
    print('This is always executed')
```

4.

```
#Q4->WAP to handle multiple errors with one.
def fun(a):
    if a < 4:
        b = a/(a-3)
    print("Value of b = ", b)
try:
    fun(3)
    fun(5)
except ZeroDivisionError:
    print("ZeroDivisionError Occurred and Handled")
except NameError:
    print("NameError Occurred and Handled")
```

5.

```
#Q5->Import module sys to get the type of exception.
import sys
randomList = ['a', 0, 2]
for entry in randomList:
    try:
        print("The entry is", entry)
        r = 1/int(entry)
        break
    except:
        print("Oops!", sys.exc_info()[0], "occurred.")
        print("Next entry.")
        print()
print("The reciprocal of", entry, "is", r)
```