General category programs

1. Write a program to print all the Non-Prime numbers between A and B?

Sample Input: A = 12 B = 19

Sample Output:

14, 15, 16, 18

```
a = int(input())
b = int(input())
for x in range (a, b+1):
   if x > 1:
      for i in range (2, x):
         if (x%i)== 0:
            break
      else:
         print (x)
```

2. Find the year of the given Anniversary is leap year or not. If leap year then print the next Anniversary, if not leap year then print the previous Anniversary.

Sample Input:

Enter Date: 04/11/1947 Sample Output:

Given Anniversary Year: Non Leap Year. Anniversary Date: 04/11/1946

```
date = input("Enter the date to be checked: ")
c=date.split("/")
b = list(map(int,c))
input_year=(b[2])

if(input_year%4 == 0):
   if(input_year%100 == 0):
      if(input_year%400 == 0):
         print("%d is Leap Year" %input_year)
      else:
         print("%d is not the Leap Year" %input_year)
   else:
      print("%d is the Leap Year" %input_year)
else:
   print("%d is not the Leap Year" %input_year)

x=input_year%4
if x!=0:
   print("Previous Leap year:", input_year-x)
else:
   print("Next leap year:", input_year+4)
```

3. Write a program to print the given number is Perfect number or not?

Sample Input: Given Number: 6
Sample Output: Its a Perfect Number

```
Number = int(input(" Please Enter any Number: "))
Sum = 0
for i in range(1, Number):
   if(Number % i == 0):
      Sum = Sum + i
if (Sum == Number):
   print(" %d is a Perfect Number" %Number)
else:
   print(" %d is not a Perfect Number" %Number)
```

4. Write a program to generate Pythagorean Triplets for the given limit.
Enter upper limit: 10
3 4 5
8 6 10

```
A=input("Enter upper limit:")
c=0
m=2
if A.isnumeric():
   x=int(A)
   while(c<x):
      for n in range(1,m+1):
         a=m*m-n*n
         b=2*m*n
         c=m*m+n*n
         if(c>x):
            break
         if(a==0 or b==0 or c==0):
            break
         print(a,b,c)
      m=m+1
else:
   print("invalid input")
```

5. Write a program to find the sum of digits of N digit number (sum should be single digit)
Sample Input: Enter N value : 3 Enter 3 digit number: 143
Sample Output: Sum of 3 digit number: 8

```
num=int(input("Enter the number:"))
Sum=0
temp=num
while temp>0:
   digit=temp%10
```

```
    Sum+=digit
    temp=temp//10
print("Sum of Digits:", Sum)
```

6. Program to find whether the given number is Armstrong number or not
Sample Input: Enter number: 153
Sample Output: Given number is Armstrong number

```
num=int(input("Enter the number:"))
Sum=0
temp=num
while temp>0:
    digit=temp%10
    Sum+=digit**3
    temp=temp//10
if Sum==num:
    print("Armstrong Number")
else:
    print("Not a Armstrong Number")
```

7. Program to find whether the given number is Harshad number or not
Sample Input: Enter number: 21
Sample Output: Given number is Harshad number

```
num=int(input("Enter the number:"))
Sum=0
temp=num
while temp>0:
    digit=temp%10
    Sum+=digit
    temp=temp//10
if num%Sum==0:
    print("Harshad Number")
else:
    print("Not a Harshad Number")
```

8. Program to find whether the given number is Happy number or not
Sample Input: Enter number: 19
Sample Output: Given number is happy number

```
def happy(n):
    temp=n
    sum=0
    while temp>0:
        digit=temp%10
        sum=digit**2+sum
        temp=temp//10
```

```
    return sum

# Main Program

num=int(input("Enter the number:"))
result=num
while result!=1 and result!=4:
    result=(happy(result))
if result==1:
    print("True")
elif result==4:
    print("False")
```

9. Program to find whether the given number is Tech number or not
Sample Input: Enter number: 3025
Sample Output: Given number is Tech number

```
n = 3025
m = str(n)
a = m[:len(m)//2]
b = m[len(m)//2:]
c = int(a)+int(b)
d = c**2

if(d==n):
    print("Tech number")
else:
    print("Not a Tech number")
```

10. Write a program using function to calculate the simple interest. Suppose the customer is a senior citizen. She is being offered 15 percent rate of interest; he is being offered 12 percent rate of interest for all other customers, the ROI is 10 percent.
Sample Input:
Enter the principal amount: 200000 Enter the no of years: 3
Gender (m/f): m
Is customer senior citizen (y/n): n Sample Output:
Interest: 60000

```
p=int(input("Enter the Principle amount:"))
n=int(input("Enter the number of years:"))
SC=input("Senior Citizen Yes/No:")
G=input("Male/Female:")
if SC=='Y' and G=='M':
    print("SI=",(p*n*12)/100)
elif SC=='Y' and G=='F':
    print("SI=",(p*n*15)/100)
else:
```

```
print("SI=",(p*n*10)/100)
```

11. Find the number of factors for the given number and print the 1st N factors of the given number.
Sample Input: Given number: 100
N: 4
Sample Output: Number of factors = 9
1st 4 factors are: 1, 2, 4, 5

```
x=int(input("Enter the number:"))
y=[]
print("The factors of",x,"are:")
for i in range(1, x):
    if x % i == 0:
        y.append(i)
print(y)
print("Number of factors:", len(y))
n=int(input("Enter N value:"))
if n>len(y):
    print("Invalid")
else:
    print("First", n, "factors:")
    for k in range(0,n):
        print(y[k], end=' ')
```

12. Write a program to print number of factors and to print nth factor of the given number.
Sample Input: Given Number: 100
N = 4
Sample Output:
Number of factors = 9 4th factor of 100 = 5

```
x=int(input("Enter the number:"))
y=[]
print("The factors of",x,"are:")
for i in range(1, x + 1):
    if x % i == 0:
        y.append(i)
print(y)
print("Number of factors:", len(y))
n=int(input("Enter N value:"))
print(n, "th factor is:",y[n-1])
```

13. Write a program to print unique permutations of a given number Sample Input: Given Number: 143 Sample Output:
Permutations are:
134
143

314
341
413
431

```
import itertools
n=input("Enter the number")
P=list(itertools.permutations(n))
print(*[''.join(p) for p in P])
```

14. Write a program to find the square, cube of the given decimal number Sample Input:
Given Number: 0.6
Sample Output: Square Number: 0.36 Cube Number:0.216

```
import math
num=float(input("Enter the number:"))
print("Square number=",math.pow(num,2))
print("Cube number=",round(math.pow(num,3),3))
```

15. Write a program to convert the Binary to Decimal, Octal Sample Input:
Given Number: 1101 Sample Output:
Decimal Number: 13 Octal: 15

```
num=input("Enter the binary number:")
bin_num="01"
for i in range(len(num)):
    binary=True
    if num[i] not in bin_num:
        print("Invalid input")
        binary=False
        break
if binary:
    dec_number=int(num,2)
    oct_number=oct(dec_number)
    hexa=hex(dec_number)
    print("Decimal Equivalent=",dec_number)
    print("Octal Equivalent=",oct_number)
    print("Hexa Equivalent=",hexa)
```

16. Add Binary

Given two binary strings a and b, return their sum as a binary string.

- a and b consist only of '0' or '1' characters.
- Each string does not contain leading zeros except for the zero itself.

Test cases:

1.Input: a = "11", b = "1"

1. Output: "100"

```
num1=input("Enter the binary number 1=")
num2=input("Enter the binary number 2=")
```

```
sum=bin(int(num1,2)+int(num2,2))
print("Sum of binary numbers: ",sum[2:])
```

17.Python program to find the greatest of three binary numbers

```
a='1101'
b='1110'
c='1111'

bina=int(a,2)
binb=int(b,2)
binc=int(c,2)

if bina>binb and bina>binc:
   print("Greatest is", a)
elif binb>bina and binb>binc:
   print("Greatest is", b)
else:
   print("Greatest is", c)
```

18.Write a program for matrix multiplication?

Sample Input:

Mat1 = $\begin{pmatrix} 1 & 2 \\ 5 & 3 \end{pmatrix}$

Mat2 = $\begin{pmatrix} 2 & 3 \\ 4 & 1 \end{pmatrix}$

Sample Output:

Mat Sum = $\begin{pmatrix} 10 & 5 \\ 22 & 18 \end{pmatrix}$

```
X=[[1,2],
  [5,3]]
Y=[[2,3],
  [4,1]]
result=[[0,0],
       [0,0]]

# iterate through rows of X
for i in range(len(X)):
  # iterate through columns of Y
  for j in range(len(Y[0])):
     # iterate through rows of Y
     for k in range(len(Y)):
        result[i][j] += X[i][k] * Y[k][j]
```

```
for r in result:
  print(r)
```

17. Write a program for matrix addition?

Sample Input:

Mat1 = $\begin{pmatrix} 1 & 2 \\ 5 & 3 \end{pmatrix}$

Mat2 = $\begin{pmatrix} 2 & 3 \\ 4 & 1 \end{pmatrix}$

Sample Output:

Mat Sum = $\begin{pmatrix} 3 & 5 \\ 9 & 4 \end{pmatrix}$

```
a=[[1,2],
  [5,3]]
b=[[2,3],
  [4,1]]
c=[[0,0],
  [0,0]]
for i in range(len(a)):
   for j in range(len(b)):
      c[i][j]=a[i][j]+b[i][j]
for i in c:
   print(i)
```

18. Find the LCM and GCD of n numbers

Sample Input:
N value = 2
Number 1 = 16
Number 2 = 20

Sample Output: LCM = 80 GCD = 4

```
n1 = int(input("Enter First number :"))
n2 = int(input("Enter Second number :"))
x = n1
y = n2
while(n2!=0):
   t = n2
   n2 = n1 % n2
   n1 = t
gcd = n1
print("GCD of {0} and {1} = {2}".format(x,y,gcd))
lcm = (x*y)/gcd
print("LCM of {0} and {1} = {2}".format(x,y,lcm))
```

19. Transpose of a matrix

```
matrix = [[4, 6, 7, 8],
          [3, 7, 2, 7],
          [7, 3, 7, 5]]
```

```
a=[[1,2],
  [3,2]]
c=[[0,0],
  [0,0]]
for i in range(len(a)):
    for j in range(len(a)):
        c[i][j]=a[j][i]
for i in c:
    print(i)
```

20. Program to find row, column and diagonal sum in Matrix

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

o/p:

```
Sum of 1 row: 6
Sum of 2 row: 15
Sum of 3 row: 24
Sum of 1 column: 12
Sum of 2 column: 15
Sum of 3 column: 18
Diagonal sum 15
```

```
#Initialize matrix a
a = [[1, 2, 3],
   [4, 5, 6],
   [7, 8, 9]]

#Calculates number of rows and columns present in given matrix
rows = len(a);
cols = len(a[0]);

#Calculates sum of each row of given matrix
for i in range(0, rows):
   sumRow = 0;
   for j in range(0, cols):
      sumRow = sumRow + a[i][j];
   print("Sum of " + str(i+1) +" row: " + str(sumRow));

#Calculates sum of each column of given matrix
for i in range(0, rows):
   sumCol = 0;
   for j in range(0, cols):
      sumCol = sumCol + a[j][i];
   print("Sum of " + str(i+1) +" column: " + str(sumCol));
```

```
#Calculates sum of diagonal
diagonal=0
for k in range(0,rows):
   diagonal=diagonal+a[k][k]
print("Diagonal sum",diagonal)
```

21. Given three integers **M, N** and **K**. Consider a grid of **M * N**, where **mat[i][j] = i * j** (1 based index). The task is to return the **Kth** smallest element in the **M * N** multiplication table.

```
def findKthNumber(m, n, k):

   low = 1
   high = n*m

   while low < high:
      mid = (low + high) // 2
      count = 0
      for i in range(1, m+1):
         count += min(n, mid//i)
      if count < k:
         low = mid + 1
      else:
         high = mid
   return low

#Driver Program
m=3
n=3
k=5
print(findKthNumber(m,n,k))
```

22. Print the sum of boundary elements of a matrix

```
def printBoundary(a, m, n):
   for i in range(m):
      for j in range(n):
         if (i == 0):
            print a[i][j],
         elif (i == m-1):
            print a[i][j],
         elif (j == 0):
            print a[i][j],
         elif (j == n-1):
            print a[i][j],
         else:
            print " ",
      print


# Driver code
if _name_ == "_main_":
```

```
a = [[1, 2, 3, 4], [5, 6, 7, 8],
     [1, 2, 3, 4], [5, 6, 7, 8]]
```

23. Print the given matrix in spiral order

```
a=[[2,5,3],
  [6,4,1],
  [9,7,8]]
l=[]
for i in range(len(a[0])):
   l.append(a[0][i])
for j in range(1,len(a)-1):
   l.append(a[j][-1])
for k in range(1,len(a[-1])+1):
   l.append(a[-1][-k])
for m in range(len(a[0])-1):
   l.append(a[1][m])
print(l)
```

24. Write a python program to find the sum of N numbers

Sample input: N=10

Sample output: Sum=55

```
N=int(input("Enter the limit:"))
count=0
for i in range(1,N+1):
   count+=i
print("Sum of N natural numbers",count)
```

25.Write a python program to find the sum of $1^2+2^2+.....N^2$ numbers

Sample input: N=6

Sample output: Sum=91

```
N=int(input("Enter the limit:"))
count=0
for i in range(1,N+1):
   count+=i*i
print("Sum of square of N natural numbers",count)
```

26.Find the factorial of the number.

Sample input: N=5

Sample output: Sum=120

```
def fact(n):
   if n==0 or n==1:
      return 1
   if n>1:
      return n*fact(n-1)

# Main program
num=int(input("Enter the number: "))
```

```
print(fact(num))
```

27.Write a python program to find the sum of 1!+2!+.....N! numbers

Sample input: N=4

Sample output: Sum=33

```
def fact(n):
    if n==0 or n==1:
        return 1
    if n>1:
        return n*fact(n-1)

# Main program
num=int(input("Enter the number: "))
sum=0
for i in range(1,num+1):
    sum+=fact(i)
print(sum)
```

28.Write a python program to find the sum of 1!/1+2!/2+.....N!/N numbers

Sample input: N=5

Sample output: Sum=34

```
def fact(n):
    if n==0 or n==1:
        return 1
    if n>1:
        return n*fact(n-1)

# Main program
num=int(input("Enter the number: "))
sum=0
for i in range(1,num+1):
    sum+=fact(i)/i
print(sum)
```

29.Write a python program to find the difference between sum of square and square of sum N numbers

Sample input: N=5

Sample output: Diff=170

```
n=20
x=(n*(n+1)*(2*n+1))/6
y=((n*(n+1))/2)**2
print("Difference:",y-x)
```

30.Write a python program to find the sum of all digits in a triangle

```
def digits_sum():
        for i in reversed(range(len(triangle_num) - 1)):
                for j in range(len(triangle_num[i])):
                        triangle_num[i][j] += max(triangle_num[i + 1][j],
triangle_num[i + 1][j + 1])
        return str(triangle_num[0][0])

#Main Program
triangle_num =
        [[3],
       [4,6],
      [2,7,6],
     [8,5,9,3]]
print(digits_sum())
```

31. Fibonacci series

```
def Fibonacci(n):

  if n < 0:
    print("Incorrect input")

  elif n == 0:
    return 0

  elif n == 1 or n == 2:
    return 1

  else:
    return Fibonacci(n-1) + Fibonacci(n-2)

# Driver Program
num=int(input("Enter the number of terms="))
for i in range(num):
  print(Fibonacci(i))
```

32.You are climbing a staircase. It takes n steps to reach the top. Each time you can either climb 1 or 2 steps. In how many distinct ways can you climb to the top?

Sol:

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

# Driver program
s = int(input("Enter the value of n: "))
print ("Number of ways = ", end="")
print (fib(s+1))
```

Output:
Enter the value of n: 5
Number of ways = 8

33. Vehicles and children program

```
M=int(input("Enter the number of vehicles:"))
N=int(input("Enter number of children: "))
x=M%N
if x==0:
   print("You are so lucky")
elif x!=0 and x%2!=0:
   print("Mr.Peter gets", x, "Vehicles")
elif x!=0 and x%2==0:
   print("Mr.Peter gets", x, "Vehicles. He is lucky")
```

34. Find the difference between two dates.

```
from datetime import datetime
from dateutil import relativedelta

# get two dates
d1 = '17/7/1980'
d2 = '16/3/2007'

# convert string to date object
start_date = datetime.strptime(d1, "%d/%m/%Y")
end_date = datetime.strptime(d2, "%d/%m/%Y")

# Get the relativedelta between two dates
delta = relativedelta.relativedelta(end_date, start_date)
print('Years, Months, Days between two dates is')
print(delta.years, 'Years,', delta.months, 'months,', delta.days, 'days')

delta.years
d3=d1.split('/')
d4=d2.split('/')
BY=int(d3[2])
JY=int(d4[2])

if(delta.years>=19 and BY%4==0):
   print("I m a lucky adult")
elif delta.years<19:
   print("I m aspiring to become adult")

if BY%4==0:
   print("Birth year is leap Year")
else:
   print("Birth year is not a leap Year")
```

```
if JY%4==0:
   print("Joining year is leap Year")
else:
   print("Joining year is not a leap Year")
```

35. Calendar Programs

```
# Current time
from datetime import datetime
now=datetime.now()
print(now)

# Current date
from datetime import datetime
now=datetime.today()
print(now)

# Entire month in a year
import calendar
y = int(input("Enter the Year :"))
print(calendar.prcal(y))

# Particular month in a year
import calendar
y = int(input("Enter the Year :"))
m=int(input("Enter the month :"))
print(calendar.month(y,m))

#Program to find number of weekdays in(mm/yyyy)
import numpy as np
# Number of weekdays in March 2017
print("Number of weekdays in March 2017:",
     np.busday_count('2017-03', '2017-04'))


# Number of sundays in Nov 2020
print("Number of Sunday in november 2020:",
     np.busday_count('2020-11', '2020-12',weekmask='Sun'))
# input year and month
yearMonth = '2017-05'

# getting date of first monday
date = np.busday_offset(yearMonth, 0, roll='forward',weekmask='Mon')
# display date
print(date)
```

I. STRING OPERATIONS AND METHODS

1.Write a program to find the number of special characters in the given statement
Sample Input:
Given statement: Modi Birthday @ September 17, #&$% is the wishes code for him.
Sample Output:
Number of special Characters: 5

```
#Python code to Count Alphabets, Special character Numeric values and space
string=input("Please enter a string: ")#take input from the user
alphabets,num,special,space=0,0,0,0;#variable declaration and initilization
a=[]
d=[]
spl=[]
for i in range(len(string)):
  if(string[i].isalpha()): #check Alphabets letters
      #print(string[i],end="")
      alphabets+=1
      a.append(string[i])
  elif(string[i].isdigit()):#check numeric value
      #print(string[i],end="")
      num+=1
      d.append(string[i])
  elif(string[i].isspace()):#check space
      space+=1
  else:
      #print(string[i],end="")
      special+=1
      spl.append(string[i])
print("Alpabets letters: ",alphabets, a)
print("\nnumbers: ",num, d)
print("\nSpace: ",space)
print("\nSpecial characters: ",special, spl)
```

2. Write a program to print the number of vowels and number of consonants in the given statement and which is maximum?
Sample Input:
Saveetha School of Engineering Sample Output:
Number of vowels = 12 Number of Consonants = 15

```
str = input("Enter the string:")
vcount, ccount= 0,0
Vowels = "AaEeIiOoUu"
c=[]
v=[]
#Converting entire string to lower case to reduce the comparisons
#str = str.lower()
for i in range(0,len(str)):
```

```
    #Checks whether a character is a vowel
    if str[i] in (Vowels):
        vcount = vcount + 1
        v.append(str[i])
        #count = [each for each in str if each in Vowels]

    elif (str[i] !=" " and str[i] not in (Vowels)):
        ccount = ccount + 1
        c.append(str[i])
print("Total number of vowel and consonant are" );
print(vcount,v)
print(ccount,c)
```

3. Program to find whether two strings have same character in same index and returns the number of matches

Sample input:
S1=”what”
S2=”watch”

Sample output:
1

```
def match(s1,s2):
    count=0

    for i in range(min(len(s1),len(s2))):
        if s1[i].lower()==s2[i].lower():
            count=count+1
    return count

#Driver Program
S1="What"
S2="watch"
print("Total number of matches:")
print(match(S1,S2))
```

4. Program to print number of words in a line and number of lines in a para

Sample input:
'''This is the most straightforward way to count the number
of lines in a text file in Python. The readlines() method reads all
lines from a file and stores it in a list. Next, use the len() function
to find the length of the list which is nothing but total lines present in a file.'''

Sample output:
Number of lines: 3
Number of words in each line:
Line 1 18
Line 2 15
Line 3 22

```
#Program to print number of words in a line and number of lines in a para

string='''This is the most straightforward way to count the number
of lines in a text file in Python. The readlines() method reads all
lines from a file and stores it in a list. Next, use the len() function
to find the length of the list which is nothing but total lines present in a file.'''
str1=string.split(".")
str1.pop()
print("Number of lines: ",len(str1))
print("Number of words in each line:")
for i in range(len(str1)):
    words=str1[i].split()
    #print(words)
    print("Line",i+1,len(words))
```

5. Program to find number of sentences starts with "B"

Sample input:
'''The apple doesn't fall. ...
All that glitters are not gold. ...
A picture is worth a thousand words. ...
Beggers can't be choosers. ...
A bird in the hand. ...
Better safe than sorry. ...
An apple a day keeps doctor away. ...
Blood is thicker than water. ...'''

Sample output:
Total number of lines: 8
Number of Sentences that start with letter B : 3

```
# Program to find number of sentences starts with "B"

string=input("Enter the Para: ")
str1=string.split(" ...")
str1.pop()
print("Total number of lines:",len(str1))
count=0
for i in str1:
    str2=i.split()
    #print(str2)
    for j in str2:
        if j[0]=="B":
            count=count+1
print("Number of Sentences that start with letter B :",count)
```

6. Write a program that finds whether a given character is present in a string or not. In case it is present it prints the index at which it is present. Do not use built-in find functions to search the character.
Sample Input:

Enter the string: I am a programmer Enter the character to be searched: p
Sample Output:
P is found in string at index: 8
Note: Check for non-available Character in the given statement as Hidden Test case.

```
str = input("Enter the String:")

# Character to find
c = input("Enter the character to find:")

# Using Naive Method
res = None
j=0
while j<len(str):
    for i in range(0,len(str),1):
        if str[i] == c:
            res = True
            print(str[i], "Index:",i)
        j=j+1
if res==None:
    print("Character not found")
```

7. Write a program to arrange the letters of the word alphabetically in Normal order and reverse order
Sample Input:
Enter the word: MOSQUE Sample Output:
Alphabetical Order Normal: E M O Q S U Alphabetical Order Reverse: U S Q O M E

```
str=input("Enter the string:")
str=str.upper()
sort_str=sorted(str)
print(sort_str)
join_str="".join(sort_str)
rev_str=join_str[::-1]
print(join_str)
print(rev_str)
```

8. Write a program to find the number of letters repeatedly present in the given word and print the Repeated letters.
Sample Input:
Enter the word: TEMPLE Sample Output:
Number of repeated letters = 1 Repeated letter = E

```
string = input("Enter the string:")
string=string.lower()
repeat=[]
print("Duplicate characters in a given string: ");
#Counts each character present in the string
```

```
for i in range(0, len(string)):
    count = 1

    for j in range(i+1, len(string)):

        if(string[i] == string[j] and string[i] != ' '):
            count = count + 1;

            #Set string[j] to 0 to avoid printing visited character
            string = string[:j] + '0' + string[j+1:]

    #A character is considered as duplicate if count is greater than 1
    if(count > 1 and string[i] != '0'):
        repeat.append(string[i])
        print(string[i],count)

print("Number of repeated characters:", len(repeat),repeat)
```

9. Write functions to perform the following String operations and identify the vowels count in string S3.

Sample input: Index: 1
S1=’welcome’ S2=’Homely’

Sample output: wHeolmceolmye

```
s1 = "welcome"
s2 = "homely"
n = int(input("n="))
output = ""
i = 0
j = 0
while i < len(s1) and j < len(s2):
    output += s1[i:i+n] + s2[j:j+n]
    i += n
    j += n
output += s1[i:] + s2[j:]
print(output)
```

10. Write a program that accepts a string from user and re displays the same string after removing vowels from it.
Sample Input & Output:
Enter a string: we can play the game The string without vowels is: w cn ply th gm

Sol:

```
text = input("Enter the String: ")

vowels = ['a', 'e', 'i', 'o', 'u', 'A', 'E', 'I', 'O', 'U']
newtext = ""
for i in range(len(text)):
```

```
    if text[i] not in vowels:
        newtext = newtext + text[i]

print("\nString after removing Vowels: ")
text = newtext
print(text)
```

11. Given two strings “s” and “t”, determine if they are isomorphic.
Input: s = "egg", t = "add"
Output: true

```
def isisomorphic(str1, str2):
    if len(str1) != len(str2):
        return False
    else:
        map1, map2 = {}, {}
        for i in range(len(str1)):
            ch1, ch2 = str1[i], str2[i]
            if ch1 not in map1:
                map1[ch1] = ch2
            print(map1)
            if ch2 not in map2:
                map2[ch2] = ch1
            print(map2)
            if map1[ch1] != ch2 or map2[ch2] != ch1:
                return False
    return True

str1 = input("String 1=")
str2 = input("String 2=")
print(isisomorphic(str1, str2))
```

12. Given an integer n, return the number of strings of length n that consist only of vowels (a, e, i, o, u) and are lexicographically sorted.
Input: n = 2
Output: 15

```
def countstrings(n, start):
    if n == 0:
        return 1
    cnt = 0

    for i in range(start, 5):

        # decrease the length of string
        cnt += countstrings(n - 1, i)
    return cnt

def countVowelStrings(n):
    return countstrings(n, 0)
```

```
n = int(input("n="))
print(countVowelStrings(n))
```

13. Given a string S consisting of N lowercase alphabets, the task is to modify the string S by replacing each character with the alphabet whose circular distance from the character is equal to the frequency of the character in S.
Input: S="ghee"
Output: higg

```
def modify_string(S):
    frequency = {}

    # Count the frequency of each character
    for char in S:
        frequency[char] = frequency.get(char, 0) + 1

    result = ""

    # Replace characters with the corresponding circular distance
    for char in S:
        circular_distance = ord(char) + frequency[char]

        if circular_distance > 122:
            circular_distance -= 26

        result += chr(circular_distance)

    return result

# Example usage:
S = "ghee"
modified_string = modify_string(S)
print(modified_string)  # Output: higg
```

14. Given two strings S1 and S2, representing sentences, the task is to print both sentences after removing all words which are present in both sentences
Input: S1 = "sky is blue in color", S2 ="Raj likes sky blue color "
Output: is in
Raj likes

```
def removeCommonWords(s1,s2):
  com=[]
  sent1=list(s1.split())
  sent2=list(s2.split())
  for i in sent1:
    if i in sent2:
      sent1.remove(i)
      sent2.remove(i)
      com.append(i)
```

```
      continue
  print(*sent1)
  print(*sent2)
  print("common words",*com)
sentence1 = input("Enter string1: ")
sentence2 = input("Enter string2: ")
removeCommonWords(sentence1,sentence2)
```

15. Given a string s consisting of words and spaces, return *the length of the **last** word in the string.* A **word** is a maximal substring consisting of non-space characters only.
Test Case:
Input: s = "Hello World"
Output: 5

```
s=input("Enter the string:")
s1=s.split()
n=len(s1)
print("Number of words: ",n)
print("Last word: ",s1[n-1], len(s1[n-1]))
```

16. Given a string s and an integer k, return the length of the longest substring of s such that the frequency of each character in this substring is greater than or equal to k.
s consists of only lowercase English letters.
Test cases:
1.Input: s = "aaabb", k = 3
Output: 3

```
def Substring(s):
    ans, temp = 1, 1
    for i in range(1, len(s)):
        if (s[i] == s[i - 1]):
            temp += 1
        else:
            ans = max(ans, temp)
            temp = 1

    ans = max(ans, temp)
    return ans

s = input("Enter the string: ")
print(Substring(s))
```

17. Reverse Words in a String
Given an input string s, reverse the order of the words.
Input: s = "the sky is blue"
Output: "blue is sky the"

```
str1=input("Enter the string: ")
str2=str1.split()[::-1]
print(*str2)
```

18. Raju, has again started troubling people in your city. The people have turned on to you for getting rid of Raju. Raju presents to you a number consisting of numbers from 0 to 9 characters. He wants you to reverse it from the final answer such that the number becomes Mirror number. A Mirror is a number which equals its reverse. The hope of people are on you so you have to solve the riddle. You have to tell if some number exists which you would reverse to convert the number into Mirror
Sample input:
Enter the number: 123456
Sample output:
Mirror image: 654321

```
num= int(input("Enter the integer: "))
num1=str(num)
num2=num1[::-1]
print(num2)
```

19. Given an array of strings strs, group **the anagrams** together. You can return the answer in **any order**.
Input: strs = ["eat","tea","tan","ate","nat","bat"]

Output: [["bat"],["nat","tan"],["ate","eat","tea"]]

```
def Anagrams(li ):
    dictionary = {}
    for word in li:
        sortedWord = ''.join(sorted(word))
        print(sortedWord)
        if sortedWord not in dictionary:
            dictionary[sortedWord] = [word]
        else:
            dictionary[sortedWord] += [word]
    return [dictionary[i] for i in dictionary]

li = ['pop','bat','tab','opp','cat']
print(Anagrams(li))
```

20. Program to print first letters of the word in a sentence separated by dot.

Sample input: "The cat on the wall"
Sample output: T.C.O.T.W.

```
string="The cat on the wall"
l1=list(string.split())
print(l1)

for i in range(len(l1)):
    print(l1[i][0].upper(),end=".")
    continue
```

21. Valid Palindrome
A phrase is a palindrome if, after converting all uppercase letters into lowercase letters and removing all non-alphanumeric characters, it reads the same forward and backward. Alphanumeric characters include letters and numbers.
Given a string s, return true if it is a palindrome, or false otherwise.
Test Cases:
1.Input: s = "A man, a plan, a canal: Panama"
Output: true

```
n="A man, a plan, a canal: Panam"
s=n.lower()
text="
for i in s:
    if i.isalpha():
        text+=i
x=text[::-1]
if(x==text):
    print("Valid Palindrome")
else:
    print("Invalid Palindrome")
```

22. Write a function delchar(s,c) that takes as input strings s and c, where c has length 1 (i.e., a single character), and returns the string obtained by deleting all occurrences of c in s. If c has a length other than 1, the function should return s.
Sample Input:
Enter the string: Hello world
Enter a character to be deleted: l
Sample output:
String after the character is removed: Heo Word

```
#Display String after removing the given character

text = input("Enter the String: ")
char= input("Enter the char: ")
newtext = ""
for i in range(len(text)):
    if text[i]!=char:
        newtext = newtext + text[i]

print("\nString after removing the char: ")
text = newtext
print(text)
```

23. Given two strings haystack and needle, return the index of needle in haystack, if not return -1.
Sample input:
Haystack=’sadbutsad’
Needle=’sad’

Sample output:

[0,6]

```
def strStr(haystack,needle):
    l=[]
    if needle==" ":
        return 0
    else:
        for i in range(len(haystack)):
            if haystack[i]==needle[0]:
                if haystack[i:i+len(needle)]==needle:
                    l.append(i)
                    continue
                else:
                    return -1
        return l

# Driver Program

haystack="sadsad"
needle="sad"
print(strStr(haystack,needle))
```

24. Write a python program to evaluate math expression w/o eval().

```
def evaluate(string):
    string = string.replace(" ", "")

    def splitby(string, separators):

        lis = []
        current = ""
        for ch in string:
            if ch in separators:
                lis.append(current)
                lis.append(ch)
                current = ""
            else:
                current += ch
        lis.append(current)
        return lis

    lis = splitby(string, "+-")

    def evaluate_mul_div(string):
        lis = splitby(string, "x/")
        if len(lis) == 1:
            return lis[0]

        output = float(lis[0])
```

```
        lis = lis[1:]

        while len(lis) > 0:
            operator = lis[0]
            number = float(lis[1])
            lis = lis[2:]

            if operator == "x":
                output *= number

            elif operator == "/":
                output /= number

        return output


    for i in range(len(lis)):
        lis[i] = evaluate_mul_div(lis[i])

    output = float(lis[0])
    lis = lis[1:]

    while len(lis) > 0:
        operator = lis[0]
        number = float(lis[1])
        lis = lis[2:]

        if operator == "+":
            output += number
        elif operator == "-":
            output -= number

    return output

# Main Program
testcases = "1+2x3-4"
print(evaluate(testcases))
```

26. Largest 3 digit Palindrome

```
# Largest Palindrome
n = 0
for a in range(999, 100, -1):
    for b in range(a, 100, -1):
        x = a * b
        if x > n:
            s = str(a * b)
            if s == s[::-1]:
                n = a * b
print(n)
```

27. Given string num representing a non-negative integer num, and an integer k, return the smallest possible integer after removing k digits from num.

Input: num = "1432219", k = 3
Output: "1219"

```
def removeKdigits(num,k):
    stack = []
    for digit in num:
        while k > 0 and len(stack) > 0 and stack[-1] > digit:
            k -= 1
            stack.pop()
        stack.append(digit)
    if k > 0:
        stack = stack[:-k]
    return "".join(stack).lstrip("0") or "0"

num="143219"
k=2
print(removeKdigits(num,k))
```

28. Return the Unicode of Uppercase letters

```
import string
import re
alphabets = list(string.ascii_uppercase)
for i in alphabets:
    print(i,"=",ord(i))
print(chr(65))
```

29. Given two strings s1 and s2, write a function that will convert s1 to s2(if possible) by using min conversion.

```
def editDistance(str1, str2, m, n):

    if m == 0:
        return n

    if n == 0:
        return m

    if str1[m-1] == str2[n-1]:
        return editDistance(str1, str2, m-1, n-1)

    return 1 + min(editDistance(str1, str2, m, n-1),    # Insert
                   editDistance(str1, str2, m-1, n),    # Remove
                   editDistance(str1, str2, m-1, n-1)    # Replace
                   )
```

```
# Driver code
str1 = "sunday"
str2 = "saturday"
print (editDistance(str1, str2, len(str1), len(str2)))
```

30.

LIST PROGRAMS

1. Program to remove duplicates numbers entirely from the sorted array

Sample Input:
Array = {15, 14, 25, 14, 32, 14, 31}

Sample Output:
Sorted Array = {15, 25, 31, 32}

Test cases:
1. {16, 16, 16 16, 16}
2. {0, 0, 0, 0}
3. {-12, -78, -35, -42}
4. {1,2,3,7,8,9,4,5,6}
5. {1-2,2-3,3-4,4-5,5-6}

Program:

```
l=[1,1,2,3]
u=[]
for i in l:
    if i not in u and l.count(i)==1:
        u.append(i)
print(list(u))
```

2. Find the Mean, Median and Mode of the array of numbers? Sample Input:
Array of elements = {16, 18, 27, 16, 23, 21, 19} Sample Output:
Mean = 20
Median = 19
Mode = 16
Test cases:
1. Array of elements = {26, 28, 37, 26, 33, 31, 29}
2. Array of elements = {1.6, 1.8, 2.7, 1.6, 2.3, 2.1, .19}
3. Array of elements = {0, 160, 180, 270, 160, 230, 210, 190, 0}
4. Array of elements = {20, 18, 18, 27, 16, 27, 27, 19, 20}
5. Array of elements = {1000, 100, 1000, 100, 1000, 100, 1000, 100, 1000}

Program:

```
import statistics
l=[1,2,3,4,5,5,1,1]
print("mean:",statistics.mean(l))
print("median:",statistics.median(l))
print("mode:",statistics.mode(l))
```

3. Python Program to create a list of all numbers in a range which are perfect squares and the sum of the digits of the number is less than 10.

Sample Input & Output:
Enter lower range: 1

Enter upper range: 40 [1, 4, 9, 16, 25, 36]

Test case:

1. Enter lower range: 50 Enter upper range: 100
2. Enter lower range: 5 Enter upper range: 8
3. Enter lower range: 10 Enter upper range: 5
4. Enter lower range: 500 Enter upper range: 500
5. Enter lower range: 0 Enter upper range: -100

Program:

```
lower_range = int(input("Enter lower range: "))
upper_range = int(input("Enter upper range: "))
result = []
for num in range(lower_range, upper_range + 1):
    sqrt = int(num ** 0.5)
    if sqrt * sqrt == num:
        digit_sum = sum(map(int, str(num)))
        if digit_sum < 10:
            result.append(num)
print(result)
```

4. Python Program to Find the Nth Largest Number in a List

Sample Input:

List : {14, 67, 48, 23, 5, 62}

N = 4

Sample Output:

4th Largest number: 23

Test cases: 1. N = 0

2. N = -5

3. N = 1

4. N = M 5. N = %

Program:

```
lst = [14, 67, 48, 23, 5, 62]
N = int(input("which largest number:"))
if N <= 0 or N > len(lst):
    print("Invalid input for N.")
else:
    sorted_list = sorted(lst, reverse=True)
    nth_largest = sorted_list[N-1]
    print(f"{N}th Largest number: {nth_largest}")
```

5. Python Program to Create a List of Tuples with the First Element as the Number and Second Element as the Square of the Number.

Sample Input:

Enter the lower range:45 Enter the upper range:49

Sample Output:

[(45, 2025), (46, 2116), (47, 2209), (48, 2304), (49, 2401)]

Test case:

1. Enter lower range: 50 Enter upper range: 100
2. Enter lower range: 5 Enter upper range: 8
3. Enter lower range: 10 Enter upper range: 5
4. Enter lower range: 500 Enter upper range: 500
5. Enter lower range: 0 Enter upper range: -100

Program:

```
lower_range = int(input("Enter the lower range: "))
upper_range = int(input("Enter the upper range: "))
if lower_range > upper_range:
    print("Invalid input: Lower range is greater than upper range.")
else:
    result = [(num, num**2) for num in range(lower_range, upper_range + 1)]
    print(result)
```

6. Write a program to find the number of composite numbers in an array of elements
Sample Input;:
Array of elements = {16, 18, 27, 16, 23, 21, 19} Sample Output:
Number of Composite Numbers = 5 Test cases:
1.Array of elements = {26, 28, 37, 26, 33, 31, 29}
2. Array of elements = {1.6, 1.8, 2.7, 1.6, 2.3, 2.1, .19}
3. Array of elements = {0, 160, 180, 270, 160, 230, 210, 190, 0}
4. Array of elements = {200, 180, 180, 270, 270, 270, 190, 200}
5. Array of elements = {100, 100, 100, 100, 100, 100, 100, 100}

Program:

```
# Sample input
array = [16, 18, 27, 16, 23, 21, 19]
# Count the number of composite numbers
count = 0
for num in array:
    if num < 4:
        continue
    for i in range(2, num):
        if num % i == 0:
            count += 1
            break
# Output the result
print("Number of Composite Numbers =", count)
```

7. Write a program to reverse an array Sample Input;:
Array of elements = {16, 18, 27, 16, 23, 21, 19} Sample Output:
Reverse Array elements = {19, 21 23, 16, 27, 18, 16} Test cases:
1.Array of elements = {26, 28, 37, 26, 33, 31, 29}
2. Array of elements = {1.6, 1.8, 2.7, 1.6, 2.3, 2.1, .19}
3. Array of elements = {0, 160, 180, 270, 160, 230, 210, 190, 0}
4. Array of elements = {200, 180, 180, 270, 270, 270, 190, 200}
5. Array of elements = {100, 100, 100, 100, 100, 100, 100, 100}

Program:

```
# Sample input
array = [16, 18, 27, 16, 23, 21, 19]
# Reverse the array using reverse indexing
reversed_array = array[::-1]
# Output the result
print("Reverse Array elements =", reversed_array)
```

8. Write a program to find the Non composite number in the array of numbers Sample Input:
Array of elements = {26, 28, 47, 26, 43, 51, 29} Sample Output:
Prime numbers in Array elements = {47, 43, 29} Test cases:
1.Array of elements = {26, 28, 37, 26, 33, 31, 29}
2. Array of elements = {1.6, 1.8, 2.7, 1.6, 2.3, 2.1, .19}
3. Array of elements = {0, 160, 180, 270, 160, 230, 210, 190, 0}
4.Array of elements = {20, 18, 18, 27, 27, 27, 190, 20}
5. Array of elements = {100, 100, 100, 100, 100, 100, 100, 100}

Program:

```
# Sample input
array = [26, 28, 47, 26, 43, 51, 29]
# Find the non-composite numbers
non_composite_numbers = []
for num in array:
    if num < 2:
        continue
    count = 0
    for i in range(2, num):
        if num % i == 0:
            count += 1
            break
    if count == 0:
        non_composite_numbers.append(num)
# Output the result
print("Non-composite numbers in Array elements =", non_composite_numbers)
```

9. Write a program to print the number of negative numbers in the list of numbers Sample Input:
List of elements = {16, -18, 27, -16, 23, -21, 19} Sample Output:
Number of negative numbers in List of elements = 3 Test cases:
1. List of elements = {-26, 28, 37, -26, 33, -31, -29}
2. List of elements = {1.6, 1.8, 2.7, -1.6, 2.3, -2.1, .19}
3. List of elements = {0, 160, 180, 270, 160, 230, 210, 190, 0}
4. List of elements = {-16, 2.8, -7, -1.5, 2.8, -2.8, -.19}
5. List of elements = {-160, -160, -180, -270, -160, -230, -210, 1-90, 0}

Program:

```
numbers = [16, -18, 27, -16, 23, -21, 19]
count = 0
for num in numbers:
    if num < 0:
        count += 1
print("Number of negative numbers in the list:", count)
```

10. Find the Mth maximum number and Nth minimum number in an array and then find the sum of it, difference of it and product of it.
Sample Input:
Array of elements = {14, 16, 87, 36, 25, 89, 34}
M = 1
N = 3
Sample Output:
1st Maximum Number = 89 3rd Minimum Number = 25 Sum = 114
Difference = 64
Product = 2225 Test cases:
1. {16, 16, 16 16, 16}, M = 0, N = 1
2. {0, 0, 0, 0}, M = 1, N = 2
3. {-12, -78, -35, -42, -85}, M = 3 , N = 3
4. {15, 19, 34, 56, 12}, M = 6 , N = 3
5. {85, 45, 65, 75, 95}, M = 5 , N = 7

Program:

```
l=[14,67,48,23,5,62]
asc=sorted(l)
dsc=asc[::-1]
n=int(input("enter which largest number:"))
m=int(input("enter which smallestnumber:"))
print(n,"largest:",dsc[n-1])
print(m,"smallest:",asc[m-1])
print("sum:",dsc[n-1]+asc[m-1])
print("difference:",dsc[n-1]-asc[m-1])
print("product:",dsc[n-1]*asc[m-1])
```

11. Write a program to merge two sorted lists to the third list.
Input: list1 = [1,2,4], list2 = [1,3,4]
Output: [1,1,2,3,4,4]

Program:

```
list1=[1,2,4]
list2=[1,2,4]
output=list1+list2
print(output)
```

12. A peak element is an element that is strictly greater than its neighbours. Given a **0-indexed** integer array nums, find a peak element, and return its index. If the array contains multiple peaks, return the index to **any of the peaks**

Input: nums = [1,2,3,1]
Output: 2

Program:

```
arr = [1, 3, 2, 4, 6, 5]
peaks = []
# Check if the first or last element is a peak
if len(arr) == 1 or arr[0] >= arr[1]:
    peaks.append(arr[0])
if arr[-1] >= arr[-2]:
    peaks.append(arr[-1])
# Iterate through the array and check for peaks
for i in range(1, len(arr) - 1):
    if arr[i] >= arr[i - 1] and arr[i] >= arr[i + 1]:
        peaks.append(arr[i])
if peaks:
    print("Peak elements:", peaks)
else:
    print("No peak elements found")
```

13. Write a program to read the numbers until -1 is encountered. Find the average of positive numbers and negative numbers entered by user. Restrict the decimal up to 2 digits.

Sample Input:
Enter -1 to exit… Enter the number: 7 Enter the number: -2 Enter the number: 9 Enter the number: -8 Enter the number: -6 Enter the number: -4 Enter the number: 10 Enter the number: -1

Sample Output:
The average of negative numbers is: -5.00 The average of positive numbers is: 8.67
Test cases:
1. -1,43, -87, -29, 1, -9
2. 73, 7-6,2,10,28,-1
3. -5, -9, -46,2,5,0
4. 9, 11, -5, 6, 0,-1
5. -1,-1,-1,-1,-1

Program:

```
l=[]
while True:
    num=int(input("enter list elements:"))
    if num==-1:
        break
    else:
        l.append(num)
pos_nums,neg_nums=[],[]
```

```
pos_avg,neg_avg=0.0,0.0
for i in range(len(l)):
    if l[i]>0:
        pos_nums.append(l[i])
        pos_avg+=l[i]
    else:
        neg_nums.append(l[i])
        neg_avg+=l[i]
print("positive avg:",pos_avg/len(pos_nums))
print("negative avg:",neg_avg/len(neg_nums))
```

14.Write a Python function sumsquare(l) that takes a nonempty list of integers and returns a list [odd, even], where odd is the sum of squares of all the odd numbers in l and even is the sum of squares of all the even numbers in l.

Sample Input:

Enter the number of elements:7
Enter the element: 18
Enter the element:9
Enter the element:1
Enter the element:12
Enter the element:13
Enter the element:4
Enter the element:30

Output:

[251,1384]

Program:

```
def sumsquare(l):
    odd_sum = 0
    even_sum = 0
    for num in l:
        if num % 2 == 0:
            even_sum += num ** 2
        else:
            odd_sum += num ** 2
    return [odd_sum, even_sum]
n = int(input("Enter the number of elements: "))
l = []
for i in range(n):
    l.append(int(input("Enter the element: ")))
output = sumsquare(l)
print(output)
```

15. Given an array of integers nums, return the number of good pairs.
A pair (i, j) is called good if nums[i] == nums[j] and i < j.
Input: nums = [1,2,3,1,1,3]
Output: 4
Explanation: There are 4 good pairs (0,3), (0,4), (3,4), (2,5) 0-indexed.

Program:

```
l=[1,2,3,1,1,3]
c=0
for i in range(0,len(l)):
    for j in range(i+1,len(l)):
        if l[i]==l[j]:
            print("(",i,j,")")
            c+=1
print("number of good pairs:",c)
```

16. How Many Numbers Are Smaller Than the Current Number
Given the array nums, for each nums[i] find out how many numbers in the array are smaller than it. That is, for each nums[i] you have to count the number of valid j's such that j != i and nums[j] < nums[i].
Input: nums = [8,1,2,2,3]
Output: [4,0,1,1,3]

Program:

```
l=[8,1,2,2,3]
l1=[]
for i in range(len(l)):
    c=0
    for j in range(len(l)):
        if l[i]>l[j]:
            c+=1
    l1.append(c)
print(l1)
```

17. A party has been organised on a cruise. The party is organised for a limited time (T). The number of guests entering (E[i]) and leaving (L[i]) the party at every hour is represented as elements of the array. The task is to find the maximum number of guests present on the cruise at any given instance within T hours.
Sample Input:
5 ---> Value of T
[7,0,5,1,3] ---> E[], element of E[0] to E[N-1], where input each element is separated by new line
[1,2,1,3,4] -----> L[],element of L[0] to L[N-1], where input each element is separated by new line
Sample Output:
8 -----> Maximum number of guests on cruise at an instance.

Program:

```
t=int(input("enter instance time:"))
e=[7,0,5,1,3]
l=[1,2,1,3,4]
x=[0,0,0,0,0]
for i in range(t):
    if t>len(e) or t>len(l):
        print("out of index")
    else:
        x[i]=(x[i-1]+e[i])-l[i]
        print(x[i],end=" ")
print("\nmax:",max(x))
```

18. Permutations
Given a collection of numbers, nums, that might contain duplicates, return all possible unique permutations in any order.
Test cases:
1.Input: nums = [1,1,2]
Output:
[[1,1,2],
[1,2,1],
[2,1,1]]
Program:

```
import itertools
p = itertools.permutations([1, 1, 2])
unique = list(dict.fromkeys(list(p)))
output = [list(perm) for perm in unique]
print(output)
```

19. Given an integer n, return *a string array* answer ***(1-indexed)*** *where*:
- answer[i] == "FizzBuzz" if i is divisible by 3 and 5.
- answer[i] == "Fizz" if i is divisible by 3.
- answer[i] == "Buzz" if i is divisible by 5.
- answer[i] == i (as a string) if none of the above conditions are true.

Input: n = 5
Output: ["1","2","Fizz","4","Buzz"]

Program:

```
n = 5
result = []
for i in range(1, n+1):
    if i % 3 == 0 and i % 5 == 0:
        result.append("FizzBuzz")
    elif i % 3 == 0:
        result.append("Fizz")
    elif i % 5 == 0:
        result.append("Buzz")
    else:
        result.append(str(i))
print(result)
```

20. Python Program to Remove the Duplicate Items from a List
Sample Input:
Enter the number of elements in list:7
Enter element1:10
Enter element2:20
Enter element3:20
Enter element4:30
Enter element5:40
Enter element6:40
Enter element7:50
Sample Output:
Non-duplicate items: [10, 20, 30, 40, 50]

Program:

```
n = int(input("Enter the number of elements in the list: "))
lst = []
for i in range(n):
    element = int(input(f"Enter element{i+1}: "))
    lst.append(element)
non_duplicate = list(set(lst))
print("Non-duplicate items:", non_duplicate)
```

21. Suppose an array of length n sorted in ascending order is rotated between 1 and n times. For example, the array nums = [0,1,2,4,5,6,7] might become:

[4,5,6,7,0,1,2] if it was rotated 4 times.
[0,1,2,4,5,6,7] if it was rotated 7 times.
Notice that rotating an array [a[0], a[1], a[2], ..., a[n-1]] 1 time results in the array [a[n-1], a[0], a[1], a[2], ..., a[n-2]].

Given the sorted rotated array nums of unique elements, return the minimum element of this array.

Input: nums = [3,4,5,1,2]
Output: 1
Explanation: The original array was [1,2,3,4,5] rotated 3 times.

Program:

```
nums = [3, 4, 5, 1, 2]
left = 0
right = len(nums) - 1
while left < right:
    mid = left + (right - left) // 2
    if nums[mid] > nums[right]:
        left = mid + 1
    else:
        right = mid
min_element = nums[left]
print(min_element)
```

22. Given an array of integers nums sorted in non-decreasing order, find the starting and ending position of a given target value. If target is not found in the array, return [-1, -1].

Input: nums = [5,7,7,8,8,10], target = 8
Output: [3,4]

Program:

```
nums = [5, 7, 7, 8, 8, 10]
target = 7
start = -1
end = -1
for i in range(len(nums)):
    if nums[i] == target:
        if start == -1:
            start = i
        end = i
result = [start, end]
print(result)
```

23. Write a python program to insert an element in a specific index.
Sample input:
Enter the number of elements=5
Enter the elements: 47,34,21,89,12

Enter the element to be Inserted: 100
Position: 4

Sample output: [12,21,34,100,47,89]

Program:

```
elements = [47, 34, 21, 89, 12]
element_to_insert = 100
position = 4
elements.insert(position, element_to_insert)
print("Modified list:", elements)
```

24. Given a date, return the corresponding day of the week for that date.
The input is given as three integers representing the day, month and year respectively.

Return the answer as one of the following values {"Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"}.

Input: day = 31, month = 8, year = 2019
Output: "Saturday"

Program:

```
import datetime

def findDay(day, month, year):
    # Create a datetime object for the given date
    date = datetime.datetime(year, month, day)

    # Get the weekday as an integer (0 = Monday, 1 = Tuesday, ..., 6 = Sunday)
    weekday = date.weekday()

    # Define a list of weekday names
    weekdays = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday",
"Sunday"]

    # Return the corresponding weekday name based on the weekday index
    return weekdays[weekday]
# Example usage
day = 31
month = 8
year = 2019
result = findDay(day, month, year)
print(result)
```

25. Write a Python program to display the MSB and LSB of a number.

Sample input: 1267899
Sample output: LSB: 1 MSB: 9

Program:

```
s=input("enter number:")
print("lsb:",s[0])
print("msb:",s[-1])
```

26. Given a 1-indexed array of integers numbers that is already sorted in non-decreasing order, find two numbers such that they add up to a specific target number. Let these two numbers be numbers[index1] and numbers[index2] where 1 <= index1 < index2 <= numbers.length.

Return the indices of the two numbers, index1 and index2, added by one as an integer array [index1, index2] of length 2.

Input: numbers = [2,3,4], target = 6
Output: [1,3]

Program:

```
l = [2, 3, 4]
t = 6
left = 0
right = len(l) - 1
while left < right:
    c_sum = l[left] + l[right]
     if c_sum == t:
        print([left + 1, right + 1])
        break
    elif c_sum > t:
        right -= 1
    else:
        left += 1
else:
    result = []
    print(result)
```

27. You are given with an array which contains integer elements. Sort these elements in ascending order. If any negative number is encountered it has to be replaced with the average of the array.

Input: [9,0,4,5,6] output: [0,4,5,6,9]

```
l = [-1, 2, 3, -4, 6]
avg = sum(l) / len(l)
for i in range(len(l)):
    if l[i] < 0:
        l[i] = avg
print(l)
```

28. Write a python program to compute the sum of all the multiples of 3 and 5 below 200.

```
c=0
l=[]
for i in range(1,200):
    if i%3==0 and i%5==0:
        l.append(i)
print(sum(l))
```

29. You have n jobs and m workers. You are given three arrays: difficulty, profit, and worker where:

For example, if three workers attempt the same job that pays $1, then the total profit will be $3. If a worker cannot complete any job, their profit is $0.
Return the maximum profit we can achieve after assigning the workers to the jobs.

Input: difficulty = [2,4,6,8,10], profit = [10,20,30,40,50], worker = [4,5,6,7]
Output: 100
Explanation: Workers are assigned jobs of difficulty [4,4,6,6] and they get a profit of [20,20,30,30] separately.

```
def maxProfitAssignment(difficulty, profit, worker):
    jobs = sorted(zip(difficulty, profit))
    res = i = best = 0
    for ability in sorted(worker):
        while i < len(jobs) and ability >= jobs[i][0]:
            best = max(jobs[i][1], best)
            i += 1
            print(jobs,best)
        res += best
    return res

#Main Program
diff = [2,4,6,8,10]
pro = [10,20,30,40,50]
w = [4,5,6,7]
print(maxProfitAssignment(diff, pro, w))
```

30. Write a python program to find the frequency of all each element present in an array.

```
import pandas as pd
 # declaring the list
l = [1, 1, 2, 2, 2, 3, 3, 4, 4, 5, 5]
 count = pd.Series(l).value_counts()
print("Element Count")
print(count)
```

31. Given an array of integers containing n+1 integers where each integer is in the range of[1,n]inclusive. There is only one repeated number. Return the repeated number.
Input: [1,3,4,2,2] output: 2

```
l = [1, 3, 4, 2, 2]
l1 = []
for i in l:
    if l.count(i) > 1:
        l1.append(i)
s=set(l1)
```

```
print(list(s))
```

32. Write an efficient algorithm that searches for a value target in an m x n integer matrix. This matrix has the following properties:
Integers in each row are sorted in ascending from left to right.
Integers in each column are sorted in ascending from top to bottom.
Input: matrix = [[1,4,7,11,15],[2,5,8,12,19],[3,6,9,16,22],[10,13,14,17,24],[18,21,23,26,30]], target = 5
Output: true

```
def searchMatrix(matrix, target):
    for row in matrix:
        if target in row:
            return True
    return False

matrix = [[1,4,7,11,15],[2,5,8,12,19],[3,6,9,16,22],[10,13,14,17,24],[18,21,23,26,30]]
target=5
print(searchMatrix(matrix,target))
```

33. Write a python program to create and display all the combinations of letters, selecting each letter from a different key in a dictionary.

Input: {1: [a,b], 2: [c,d]}
Output: ac,ad,bc,bd

```
import itertools
d ={'1':['a','b'], '2':['c','d']}
for combo in itertools.product(*[d[k] for k in sorted(d.keys())]):
    print(''.join(combo))
```

34. Patterns

1 1 2 1 2 3 1 2 3 4 1 2 3 4 5	1 2 2 3 3 3 4 4 4 4 5 5 5 5 5	10 10 20 10 20 30 10 20 30 40 10 20 30 40 50	0.1 0.1 0.2 0.1 0.2 0.3 0.1 0.2 0.3 0.4 0.1 0.2 0.3 0.4 0.5

35. Patterns

10
5 5
20 20 20
10 10 10 10