//q1.Write a program to convert number of days to measure of time given in years, weeks and days(Ignore leap year)

```
def days_to_years_weeks_days(days):
    years = days // 365
    remaining_days = days % 365
    weeks = remaining_days // 7
    remaining_days = remaining_days % 7
    return years, weeks, remaining_days

days = int(input("Enter number of days: "))
years, weeks, remaining_days = days_to_years_weeks_days(days)

print(f"{days} days is equivalent to {years} years, {weeks} weeks, and {remaining_days} days.")
```

//q2.Write a Program to Prompt for a Score between 0 and 100 and display the result as below:

```
def get_grade(score):
    if score >= 90:
        return "O (Outstanding)"
    elif score >= 80:
        return "A+ (Excellent)"
    elif score >= 70:
        return "A (Very Good)"
    elif score >= 60:
        return "B+ (Good)"
    elif score >= 55:
        return "B (Above Average)"
    elif score >= 50:
        return "C (Average)"
    elif score >= 40:
        return "P (Poor)"
    else:
        return "F (Fail)"

score = float(input("Enter the score (between 0 and 100): "))

if 0 <= score <= 100:
    grade = get_grade(score)
    print(f"Grade: {grade}")
else:
    print("Invalid score! Please enter a score between 0 and 100.")
```

//q3.Program to Check If a Given Year Is a Leap Year

```
def is_leap_year(year):
    return (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0)

year = int(input("Enter a year: "))

if is_leap_year(year):
    print(f"{year} is a leap year.")
else:
    print(f"{year} is not a leap year.")
```

//q4.Enter the numbers till the user enters word “end”, Count and print the number of Odd, Even, Positive number, Negative number, Prime number and average in each case.

```
def is_prime(num):
   if num < 2:
      return False
   for i in range(2, num):
      if num % i == 0:
         return False
   return True

odd_count = even_count = positive_count = negative_count = prime_count = total = count = 0

#loop to take in all the numbers until user enters ’end’
while(1):
   num = input("Enter a number (type ’end’ to finish): ")
   if num.lower() == ’end’:
      break          #exiting the loop once ’end’ is entered and to resume calculations
   num = int(num)     #converts the num to interger value
   total += num       #adds all the numbers entered by user
   count += 1         #increments everytime user enters a num to keep count of total number

   if num % 2 == 0:
      even_count += 1
   else:
      odd_count += 1

   if num > 0:
      positive_count += 1
   elif num < 0:
      negative_count += 1

   if is_prime(num):
      prime_count+=1

if count > 0:
   average = total / count
else:
   average = 0


print(f"Odd numbers: {odd_count}")
print(f"Even numbers: {even_count}")
print(f"Positive numbers: {positive_count}")
print(f"Negative numbers: {negative_count}")
print(f"Prime numbers: {prime_count}")
print(f"Average: {average}")
```

//q5. Write a python program to display the Fibonacci sequence between the intervals specified by the user and also print the count of odd, even and prime Fibonacci numbers in the same interval.

```
def is_prime(num):
    if num < 2:
        return False
    for i in range(2, num):
        if num % i == 0:
            return False
    return True

n = int(input("Enter the number of terms"))

n1=0
n2=1
count= even = odd =prime =0

if n<0:
    print("Enter +ve number")
elif n==1:
    print(n1)
else:
    while count<n:
        print(n1)
        if n1%2==0:
            even+=1
        else:
            odd+=1
        if is_prime(n1):
            prime += 1

        nth=n1+n2
        n1=n2
        n2=nth
        count+=1

    print(f"even count = {even}")
    print(f"odd count = {odd}")
    print(f"Prime count = {prime}")
```

//q6. Write a Program to compute area of hexagon, area of a pentagon, area of octagon, area of decagon using functions

```
#area for different figures
from math import sqrt
a=2

area1=((sqrt(5*(5+2*sqrt(5))))*a*a)/4
print(f"area of pentagon = {area1}")

area2=(3*sqrt(3)*a*a)/2
print(f"area of hexagon = {area2}")

area3=(2*(1+sqrt(2))*a*a)
print(f"area of octagon = {area3}")

area4=(5*a*a*(sqrt(5+2*sqrt(5))))/2
print(f"area of decagon = {area4}")
```

//q7. WAP that accepts a sentence and calculate the number of vowels, consonants, words, digits, blanks, uppercase letters and lowercase letters.

```
def string_processing(string):
    word_count = vowels = consonants = digits = blanks = upper_count = lower_count = 0

    for char in string:
        if char.isdigit():
            digits += 1
        elif char.isspace():
            blanks += 1
        elif char.isalpha():
            if char.lower() in ’aeiou’:
                vowels += 1
            else:
                consonants += 1
            if char.islower():
                lower_count += 1
            else:
                upper_count += 1

    # Words are better counted by splitting the string based on spaces
    words = string.split()
    word_count = len(words)

    print(f"No of vowels in the string = {vowels}")
    print(f"No of consonants in the string = {consonants}")
    print(f"No of words in the string = {word_count}")
    print(f"No of digits in the string = {digits}")
    print(f"No of blanks in the string = {blanks}")
    print(f"No of upper case in the string = {upper_count}")
    print(f"No of lower case in the string = {lower_count}")

string = input("Enter a string: ")
string_processing(string)
```

//q8. patterns:

```
def print_butterfly(rows):
    # Upper part of the butterfly
    for i in range(1, rows + 1):
        for j in range(1, i + 1):
            print("*", end=" ")
        for j in range(1, 2 * (rows - i) + 2):
            print(" ", end=" ")
        for j in range(1, i + 1):
            print("*", end=" ")
        print()

    # Lower part of the butterfly
    for i in range(rows, 0, -1):
        for j in range(1, i + 1):
            print("*", end=" ")
        for j in range(1, 2 * (rows - i) + 2):
            print(" ", end=" ")
        for j in range(1, i + 1):
            print("*", end=" ")
        print()

# Example usage
rows = 3
print_butterfly(rows)
```

```
*           *
* *       * *
* * *   * * *
* * *   * * *
* *       * *
*           *
```

```
def print_staircase(n):
    for i in range(1, n + 1):
        # Print spaces before the #
        print(" " * (n - i), end="")
        # Print the # symbols
        print("*" * i)

n=5
print_staircase(n)
```

```
    *
   **
  ***
 ****
*****
```

//q9. Write Python Program to Sort Numbers in a List in Ascending Order Using Bubble Sort by Passing the List as an Argument to the Function Call.

```
def bubble_sort(arr):
    n = len(arr)
    for i in range(n):
        for j in range(0, n-i-1):
            if arr[j] > arr[j+1]:
                arr[j], arr[j+1] = arr[j+1], arr[j]
        return arr

user_input = input("Enter the list elements separated by spaces: ")
arr = [int(x) for x in user_input.split()]

print("Original list: ", arr)
bubble_sort(arr)
print("Sorted list: ", arr)
```

//q10. Prompt the user to enter list of marks, increase the odd marks by 1 and print the list of modified marks, average marks.

```
# Prompt the user to enter the list of marks
input_marks= input("Enter the list of marks separated by spaces: ")
marks = [int(x) for x in input_marks.split()]

for i in range(len(marks)):
    if marks[i] % 2 != 0:
        marks[i] += 1

print("Modified marks:", marks)

total_marks = sum(marks)
average_marks = total_marks / len(marks)
print("Average marks:", average_marks)
```

//q11.Write a program to find Mean, Variance and Standard Deviation of List Numbers

```
import math
from math import sqrt

input_str = input("Enter the list numbers to calculate with spaces: ")
n = [int(x) for x in input_str.split()]

def statistics(n):
    mean = sum(n)/len(n)
    print(f"Mean is {mean}")

    variance = 0
    for i in n:
         variance+=(i - mean)**2
    variance /= len(n)
    print(f"Var is {variance}")

    std_dev = sqrt(variance)
    print(f"Std dev is {std_dev}")

#statistics([1,2,3,4])
statistics(n)
```

//q12.Write a python program to add/subtract two matrices

```
matrix1=[[1,2,3],
        [4,5,6],
        [7,8,9]]
matrix2=[[1,2,3],
        [4,5,6],
        [7,8,9]]
matrix_result=[[0,0,0],
              [0,0,0],
              [0,0,0]]
```

```
for rows in range (len(matrix1)):
    for columns in range (len(matrix2[0])):
        matrix_result[rows][columns] = matrix1[rows][columns]+matrix2[rows][columns]

print("addition of 2 matrices is: ")
for items in matrix_result:
    print(items)
```

//q13.Write a python program to input information for n number of students as given below:
Name, Registration number, Attendance, Total marks. The user has to specify a value for n number of students. The program should output the registration number and marks of a specified student given his name.

```
#creating a dict
def student_details(no_of_students):
    student_name = {}
    for i in range(0,no_of_students):
        name = input("Enter the name")
        usn = input("Enter the usn")
        total_marks = input("Enter total marks")
        attendance=input("Enter attendance")
        student_name[name]=[usn,total_marks]

    student_search=input("Enter name of student u want to search")

    if student_search not in student_name.keys():
        print("Student u searched not exists")
    else:
        print("student u r searching is present")
        print(f"student reg num is {student_name[student_search][0]}")
        print(f"student total marks is {student_name[student_search][1]}")


no_of_students = int(input("enter the number of students: "))
student_details(no_of_students)
```

//q14. Write a program to search an element using Binary Search of a Sorted List/ Linear search

```
#Perform linear search to find the target element in the list.
def linear_search(arr, target):
    n=len(arr)
    for i in range(n):
        if arr[i] == target:
            return i
    return -1

#Perform binary search to find the target element in the sorted list.
def binary_search(arr, target):
    n=len(arr)
    left, right = 0,n- 1
    while left <= right:
        mid = (left + right) // 2

        if arr[mid] == target:
            return mid

        elif arr[mid] < target:
            left = mid + 1

        else:
            right = mid - 1
    return -1



    # Sorted list
sorted_list = [1, 3, 5, 7, 9, 11, 13, 15, 17, 19]

    # Target element to search for
target = 13

    # Perform linear search
linear_index = linear_search(sorted_list, target)
if linear_index != -1:
    print(f"Linear search: Element {target} found at index {linear_index}.")
else:
    print(f"Linear search: Element {target} not found.")

    # Perform binary search
binary_index = binary_search(sorted_list, target)
if binary_index != -1:
    print(f"Binary search: Element {target} found at index {binary_index}.")
else:
    print(f"Binary search: Element {target} not found.")
```

//q15. Write a Program to sort an array in ascending/ descending order using Bubble sort/selection sort/Insertion sort.

```
def bubble_sort(arr, ascending=True):
    n = len(arr)
    for i in range(n):
        for j in range(0, n-i-1):
            if (ascending and arr[j] > arr[j+1]) or (not ascending and arr[j] < arr[j+1]):
                arr[j], arr[j+1] = arr[j+1], arr[j]

def selection_sort(arr, ascending=True):
    n = len(arr)
    for i in range(n - 1):
        min_idx = i
        for j in range(i + 1, n):
            if (ascending and arr[j] < arr[min_idx]) or (not ascending and arr[j] > arr[min_idx]):
                min_idx = j
        arr[i], arr[min_idx] = arr[min_idx], arr[i]

def insertion_sort(arr, ascending=True):
    n = len(arr)
    for i in range(1, n):
        key = arr[i]
        j = i - 1
        while j >= 0 and ((ascending and arr[j] > key) or (not ascending and arr[j] < key)):
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = key


arr = [64, 25, 12, 22, 11]

print("Original array:", arr)

# Choose sorting algorithm and order
sorting_algorithm = input("Enter sorting algorithm (bubble/selection/insertion): ").lower()
sorting_order = input("Enter sorting order (ascending/descending): ").lower()

if sorting_algorithm == ’bubble’:
    bubble_sort(arr, sorting_order == ’ascending’)
elif sorting_algorithm == ’selection’:
    selection_sort(arr, sorting_order == ’ascending’)
elif sorting_algorithm == ’insertion’:
    insertion_sort(arr, sorting_order == ’ascending’)
else:
    print("Invalid sorting algorithm.")

print("Sorted array:", arr)
```