1.write a program to find whether a number is prime number

```
n=int(input("enter a number:"))
if n<=1:
    print(n," is not a prime number")
else:
    for i in range(2,n):
        if n%i==0:
            print(n,"is not a prime number")
            break
    else:
        print(n,"is a prime number")
```

2.write a program to print m raise to power n,where m and n are read from the user.

```
m=int(input("enter the base(m):"))
n=int(input("enter the exponent(n):"))
result=m**n
print(m,"raised to the power ",n,"is:",result)
```

3.write a program having a parameterized function that returns True or False depending on whether the parameter passed is even or odd

```
def is_even(number):
    if number%2==0:
        return True
    else:
        return False
num=int(input("enter  a number:"))
if is_even(num):
    print(num,"is even")
else:
    print(num,"is odd")
```

4.write a program to print the summation of the following series upto n terms ,1-2+3-4+5-6+7--------n

```
n=int(input("enter  a number of terms(n) :"))
total=0
for i in range(1,n+1):
    if i%2==0:
        total-=i
    else:
        total+=i
print("sum of the seris upto ",n,"terms is: ",total)
```

5.write a menu driven program to perform the following operations on strings using string built in functions

a) Find the frequency of a character in a string
b) Replace a character by another character in a string
c) Remove the first occurrence of a character from a string
d) Remove all occurrence of a character from a string

```
def find_frequency(string, char):
    return string.count(char)


def replace_character(string, old, new):
    return string.replace(old, new)


def remove_first_occurrence(string, char):
    return string.replace(char, '', 1)


def remove_all_occurrences(string, char):
    return string.replace(char, '')


while True:
    print("\nMenu:")
    print("1. Find frequency of a character")
    print("2. Replace character")
    print("3. Remove first occurrence of a character")
    print("4. Remove all occurrences of a character")
    print("5. Exit")

    choice = input("Enter your choice (1-5): ")
```

```
if choice == '1':
    s = input("Enter the string: ")
    ch = input("Enter the character to find frequency: ")
    print("Frequency:", find_frequency(s, ch))

elif choice == '2':
    s = input("Enter the string: ")
    old = input("Enter the character to replace: ")
    new = input("Enter the new character: ")
    print("Modified string:", replace_character(s, old, new))

elif choice == '3':
    s = input("Enter the string: ")
    ch = input("Enter the character to remove first occurrence: ")
    print("Modified string:", remove_first_occurrence(s, ch))

elif choice == '4':
    s = input("Enter the string: ")
    ch = input("Enter the character to remove all occurrences: ")
    print("Modified string:", remove_all_occurrences(s, ch))

elif choice == '5':
    print("Exiting the program.")
    break

else:
        print("Invalid choice. Please enter a number between 1 and 5.")
```

6. write a program that accepts two strings and returns the indices of all the occurrences of the second string in the first string as a list. if the second string is not present in the first string ,then is should return -1.

```
def find_positions(text, word):
    result = []
    pos = text.find(word)

    while pos != -1:
        result.append(pos)
        pos = text.find(word, pos + 1)

    if result:
        return result
    else:
        return -1

# Example usage
main_text = input("Enter the main string: ")
sub_text = input("Enter the substring to find: ")

positions = find_positions(main_text, sub_text)
print("Positions:", positions)
```

7.using Numpy module write menu driven program to do following

a) Create an array filled with 1’s
b) Find maximum and minimum values from an array
c) Dot product of 2 arrays
d) Reshape a 1-D array to 2-D array.

```
import numpy as np


while True:
    print("\nMenu:")
    print("1. Create an array filled with 1's")
    print("2. Find maximum and minimum values from an array")
    print("3. Dot product of 2 arrays")
    print("4. Reshape 1-D array to 2-D")
    print("5. Exit")

    choice = input("Enter your choice (1-5): ")

    if choice == '1':
        size = int(input("Enter number of elements: "))
        arr = np.ones(size)
        print("Array of 1's:", arr)

    elif choice == '2':
        arr = np.array(eval(input("Enter array elements as list (e.g. [1, 2,
3]): ")))
        print("Max:", np.max(arr))
        print("Min:", np.min(arr))
    elif choice == '3':
        a = np.array(eval(input("Enter first array: ")))
        b = np.array(eval(input("Enter second array: ")))
        if a.shape == b.shape:
            print("Dot product:", np.dot(a, b))
        else:
            print("Arrays must be of same shape for dot product.")

    elif choice == '4':
        arr = np.array(eval(input("Enter 1-D array (e.g. [1,2,3,4]): ")))
        rows = int(input("Enter number of rows: "))
        cols = int(input("Enter number of columns: "))
        if rows * cols == arr.size:
            reshaped = arr.reshape((rows, cols))
            print("Reshaped array:\n", reshaped)
        else:
            print("Invalid shape. Total elements mismatch.")

    elif choice == '5':
        print("Exiting...")
        break

    else:
        print("Invalid choice.")
```

8. write a function that takes a sentence as input from the user and calculates the frequency of each letter .use a variable of dictionary type to maintain the count

```
def letter_frequency(sentence):
    freq = {} # dictionary to store frequency
    for char in sentence:
        if char.isalpha(): # consider only letters
            char = char.lower() # ignore case
            freq[char] = freq.get(char, 0) + 1
    return freq


# Example usage
sentence = input("Enter a sentence: ")
result = letter_frequency(sentence)
print("Letter frequencies:")
for key, value in result.items():
    print(f"{key}: {value}")
```

9.consider a tuplet1=(1,2,5,7,9,2,4,6,8,10).write a program to perform following operations

a) Print contents of t1 in 2 separate lines such that half values comes on one line and other half in the next line
b) Print all even values of t1 as another tuple t2
c) Concatenate a tuple t2=(11,13,15)with t1
d) Return maximum and minimum value from t1

```
t1 = (1,2,5,7,9,2,4,6,8,10)


 line and half in the other line
print("First half:", t1[:len(t1)//2])
print("Second half:", t1[len(t1)//2:])
```

```
t2 = tuple([x for x in t1 if x % 2 == 0])
print("Even values tuple t2:", t2)

t3 = t1 + (11,13,15)
print("Concatenated tuple:", t3)

print("Maximum value in t1:", max(t1))
print("Minimum value in t1:", min(t1))
```

10.write a function that reads a file file1 and copies only alternative lines to another file file2.alternative lines copied should be the odd numbered lines.

```
def copy_alternate_lines(file1, file2):
    try:
        with open(file1, "r") as f1, open(file2, "w") as f2:
            for i, line in enumerate(f1, start=1):
                if i % 2 != 0:
                    f2.write(line)
    except FileNotFoundError:
            print("file not found")

copy_alternate_lines("File1.txt", "File2.txt")
```

11.write a python program to handle a ZeroDivisionError exception when dividing a number by zero

```
def divide_numbers():
    try:
        numerator = float(input("Enter the numerator: "))
        dominator = float(input("Enter the denominator: "))
        result = numerator / dominator
    except ZeroDivisionError:
        print("Error: Cannot divide by zero.")
    else:
        print(f"The result is: {result}")
divide_numbers()
```

12. write a program that reads a list of integers from the user and throws an exception if any numbers are duplicate

```
def read_unique_integers():
    try:
        numbers = list(map(int, input("Enter a list of integers separated by
spaces: ").split()))
        unique_numbers = []
        duplicates = []

        for num in numbers:
            if num in unique_numbers:
                duplicates.append(num)
            else:
                unique_numbers.append(num)

        if duplicates:
            print("Duplicate values found:", duplicates)
        else:
            print("All numbers are unique.")
    except ValueError as e:
        print("Error:", e)
read_unique_integers()
```

13.write a program that makes use of a function to display sine, cosine, polynomial and exponential curves

```
import numpy as np
import matplotlib.pyplot as plt

def plot_curves():
    x = np.linspace(-10, 10, 400)
    y_sin = np.sin(x)
    y_cos = np.cos(x)
    y_exp = np.exp(x)
    y_poly = x**3 + x**2 + x + 6

    plt.figure(figsize=(12, 8))

    # Sine
    plt.subplot(2, 2, 1)
    plt.plot(x, y_sin, color='blue')
    plt.title("Sine Curve")
    plt.grid(True)

    # Cosine
    plt.subplot(2, 2, 2)
    plt.plot(x, y_cos, color='green')
    plt.title("Cosine Curve")
    plt.grid(True)

    # Polynomial
    plt.subplot(2, 2, 3)
    plt.plot(x, y_poly, color='red')
    plt.title("Polynomial Curve")
    plt.grid(True)

    # Exponential
    plt.subplot(2, 2, 4)
    plt.plot(x, y_exp, color='purple')
    plt.title("Exponential Curve")
    plt.ylim(0, 1000)
    plt.grid(True)

    plt.tight_layout()
    plt.show()
plot_curves()
```

14.takes as input in the months and profits made by a company ABC over a year. represent this data using a line plot .generated line plot must include X axis label name=month number and Y axis label name=total profit.

```
import matplotlib.pyplot as plt
```

```
def plot_profits():
    months = list(range(1, 13))
    profits = []
    print("Enter the profits for each of the 12 months:")
    for i in months:
        profit = float(input(f"Profit for month {i}: "))
        profits.append(profit)
    plt.figure(figsize=(10, 5))
    plt.plot(months, profits, marker='o', linestyle='-', color='blue')
    plt.xlabel("Month Number")
    plt.ylabel("Total Profit")
    plt.title("Monthly Profit of Company ABC")
    plt.grid(True)
    plt.xticks(months)
    plt.show()

plot_profits()
```