1. Number game between user and computer.

Solution:-

Source code

```
import random

def number_game():
    max_num = 20
    current = 0

    while current < max_num:
        try:
            user_input = int(input("Enter 1,2, or 3 to add
                                   to the sequence: "))
            if user_input not in [1, 2, 3]:
                print("Invalid input. please enter 1, 2 or 3")
                continue

            current += user_input
            print(f"player: {current}")

            if current >= max_num:
                print("player wins!!!")
                return

            computer_play = random.randint(1, min(3,
                                   max_num - current))
            current += computer_play
            printf"computer played: {list(range(current
                - computer_play + 1, current + 1))}")
```

```
if current >= max_num:
    print("computer wins!!!")
    return
except ValueError:
    print("Please enter a valid integer.")

number_game()
```

Output

Enter 1, 2, or 3 to add to the sequence: 3
Player: 3
computer played: [4, 5]
Sequence: 3
Player: 8
Computer played: [9, 10, 11]
Sequence: 3
Player: 14
~~Sequence: 3~~
computer played: [15, 16, 17]
Sequence: 3
Player: 20
Player wins!!!

2. Developing a function to print required Pascal triangle.

Solution:-

Source code

```
def ncr(n,r):
    from math import factorial
    return factorial(n)

def print_pascal_triangle(rows):
  for i in range(rows):
    for j in range(i+1):
      print(ncr(i,j), end=" ")
    print()

rows=int(input("Enter the no.of rows for Pascal's
                                Triangle : "))
print_pascal_triangle(rows)
```

Output

```
Enter the no.of rows for pascal's Triangle
: 3

1
1 1
1 2 1
```

3. Write a python program to print the repeated elements with frequency count in a list.

Solution

Source Code

```
def frequency_count(lst):
    freq_dict = {}
    for item in lst:
        freq_dict[item] = freq_dict.get(item,0)
                          +1
    for item, count in freq_dict.items():
        print(f"Element {item} has come
                {count} times")

# Sample list
lst = [2,2,1,2]
frequency_count(lst)
```

Output

Element 2 has come 3 times
Element 1 has come 1 times.

4. Develop a Python code to read matrices A & B and perform the addition of Matrices A & B and print the results.

Solution:

Source code

```
def read_matrices_and_add(file_path):
    with open(file_path, 'r') as file:
        rows, cols = map(int, file.readline().
                         split())

        matrix_a = [[int(num) for num in file.
                     readline().split()] for _ in range(rows)]
        matrix_b = [[int(num) for num in file.
                     readline().split()] for _ in range(rows)]

        result = [[matrix_a[i][j] + matrix_b[i][j] for
                   j in range(cols)] for i in range
                   (rows)]
        print("Result of Matrix Addition:")
        for row in result:
            print(row)

file_path = "matrices.txt"
read_matrices_and_add(file_path)
```

Output

Create the matrices . text file

for ex:-

```
2 3
1 2 3
4 5 6
7 8 9
10 11 12
```

Output of Matrix Addition:

```
[8, 10, 12]
[14, 16, 18]
```

5. Write a program that can add two objects of the class Fraction.

Solution:-

Source code

```
Class Fraction:

    def __init__(self, numerator, denominator):
        self.numerator = numerator
        self.denominator = denominator

def __add__(self, other):

    new_numerator = (self.numerator * other.
        denominator + other.numerator *
            self.denominator)
```

```
        new_denominator = self.denominator *
                          other.denominator
        return Fraction(new_numerator,
                        new_denominator)

    def __str__(self):
        return f"{self.numerator}/{self.
                                    denominator}"

frac1 = Fraction(1,6)
frac2 = Fraction(1,4)
result = frac1 + frac2
print("Result of Fraction Addition:", result)
```

Output

Result of Fraction Addition: 10/24.