ASSIGNMENT-2

NAME:- T. Pavan Kumar

REG NO:- 192210290

COURSE CODE:- CSA0914

COURSE NAME:- Programming in Java for Raspberry Pi

Submitted To:-

Dr. Hemavathi R.

Aim: To write down the program to Reverse a number

Procedure:

- Take an integer input from the user
- Initialize a variable reversed to 0
- While the input is greater than zero
  - → Take the last digit of the input number by finding the remainder of the number when divided by 10.
  - → Add the last digit to reversed after shifting it's current value to the left by one digit.
- Print the reversed numbers.

Program:

```
Public class reverse number{
    Public static void main (String[]args) {
        int num = 1234;
        int reversed = 0;
        While (num>0) {
            int last digit = num%10;
            reversed = reversed * 10 + last digit;
            num%. = 10;
            3
            System.out.Println("Reversed numbers :" + reversed);
        3
    3
```

Input:

Enter a number: 1234

Output:

Reversed number: 4321

2 **Aim:** To write a Java program to check whether a number is Armstrong number or not using while loop.

Procedure:

- take an integer input from the user.
- Calculate the number of digits in the input number.
- Initialize a variable 'sum' to 0.
- for each digit in the input number.
  - → Raise the digit to the power of the number of digits
  - → Add the result to the sum variable.
- Check if the sum is equal to original number.
- Print the result.

Program:

```
Public Class Armstrong number {
    Public Static void main (String [] args) {
        int num = 153;
        int num digits = count Digits (num);
        int sum = 0;
        int temp = num;
        while (temp > 0){
            int digit = temp % 10;
            sum += (int) math.Pow (digit, num digits);
            temp /= 10;
        }
        if (sum == num)
        {
            system.out.Println ("Armstrong number");
        }
```

```
else
  {
    System.out.Println("Not an Armstrong Number");
  }
}
Public static int Count Digits(int num)
{
  int count = 0;
  while (num > 0){
      num /= 10;
      count++;
    }
    return count;
  }
}
```

Input:-

Enter a number: 153

Out Put:-

It is a Armstrong Number.

3.

Aim:- To write a Java Program to calculate the GCD of two numbers.

Pseudocode:-

- Take two integers from user n_1 and n_2
- If n_2 is 0 return n_1 as the GCD.
- Otherwise calculate the remainder of n_1 divided by n_2 and store it in a temporary variable temp.
- Replace n_1 with n_2 and n_2 with temp.
- Final value of n_1 is the GCD.

Program :-

```
Public class GCD {
    Public Static void main (String [] args) {
     int n1 = 12;
     int n2 = 15;
     int gcd = calculate GCD (n1, n2)
     System.out.Println ("The GCD of "+ n1 + "and " + n2 + "is" +
      gcd);
    }
    Public static int Calculate GCD (int n1, int n2) {
      while (n2 ≠ 0) {
        int temp = n1 % n2;
            n1 = n2;
          n2 = temp;
      }
      return n1;
    }
}
```

InPut :-

Enter first number: 12

Enter second number: 15

OutPut :-

The GCD of 12 and 15 is 3.

Aim:- To write a Java Program to merge two sorted arrays

Pseudocode:-

- Initilite the variables.
- Create a new array result that is equal to the size of both the arrays.
- Initialize three endices i too, j to 0 and k to.
- While i is less than arr1, j is less than arr 2.
- Print the result.

Program:-

```
Public Class mergeSorted arrays{
    Public Static Void main (String[] args) {
        int arr1 [] = {1,3,5,7}
        int arr2 [] = {2,4,6,8}
        int result = mergeSorted Arrays (arr1, arr2)
        System.out.Println ("Merged array: " + java.util.toString (result));
        }
        Public Static int[] mergeSorted Array (int arr1[], int arr2[])
        {
          int result [] = new int [arr1.length + arr2.length];
          while (i<arr1.length && j<arr2.length) {
              if (arr1 [i] ≤ arr2[j]) {
                  result [k++] = arr2 [i++];
              }
              else {
```

```
            result[k++] = arr2[j++];
        }
    }
    while (i < arr1.length){
        result[k++] = arr1[i++];
    }
    while (j < arr2.length) {
        result[k++] = arr2[j++];
    }
    return result;
  }
}
```

Output :- [1, 2, 3, 4, 5, 6, 7, 8]

5) Aim :- To write a Java Program to count the frequency of characters in the string.

Pseudocode :-

- Take a String 'input' as input.
- Create a Hashmap 'char frequency' to store the frequency of each character.
- Initializes an empty Hashmap 'char frequency'.
- Iterate through each character 'c' in the input string.
  - → If c is already a key 'char frequency' increment value by 1.
  - → otherwise add 'c' as a new key in 'char frequency' with the value of 1.

- Return the 'char frequency' Hashmap.

Program :-

```
import java.util.HashMap;
import java.util.Map;
Public Class Character Frequency {
    Public Static void Main(String[]args) {
        String input = "hello world";
        map<Character, Integer> charFrequency = count Character
        Frequency (input);
        System.out.println ("Character Frequency:" + Char frequency);
    }
    Public static Map< character, Integer> count Character Frequency
     (String input) {
    Map <Character, Integer> Char frequency = new HashMap <> ();
    for (Char c: input.to CharArray()){
        if (Charfrequency.containkey(c)) {
            Char Frequency.Put (c, Char frequency.get(c)+1);
        }
        else {
            Char frequency.Put (c,1);
        }
    }
    return Char frequency;
}
}
```

Output :-

Character frequency = {h=1, e=1, l=3, O=2, w=1, r=1, d=1}