****

**NAME: -** Darji Akshatkumar Hiteshbhai

**RollNo: -** 23MCD001

**Branch: -** M.tech-CSE**(Data Science)**

**Subject: -** Complexity Theory & Algorithms

**Practical-4**

**Aim:** Solve Make-Change problem using Greedy approach.

**Code for Make-Change Problem –**

#include <bits/stdc++.h>

using namespace std;

void MakingChangeProblem(vector<int> &coins, int change, int n)

{

    vector<int> coinCount(n, 0);

    for (int i = 0; i < n; i++)

    {

        if (coins[i] > change)

        {

            continue;

        }

        else

        {

            change -= coins[i];

            coinCount[i]++;

            i--;

        }

    }

    int TotalCoins = accumulate(coinCount.begin(), coinCount.end(), 0);

    if (change > 0)

    {

        cout << "IMPOSSIBLE";

    }

    else

    {

        cout << "Optimal Coins used are: " << TotalCoins;

        cout << endl;

        for (int i = 0; i < n; i++)

        {

            if (coinCount[i] > 0)

            {

                cout << "{" << coins[i] << "}"

                     << " x " << coinCount[i] << " time"

                     << ", ";

            }

        }

    }

}

int main()

{

    int n;

    cout << "Enter the size of Demons: ";

    cin >> n;

    // vector<int> coins = {10, 20, 50, 100};

    vector<int> coins(n);

    for (int i = 0; i < n; i++)

    {

        cin >> coins[i];

    }

sort(coins.begin(), coins.end());

    reverse(coins.begin(), coins.end());

    int change;

    cout << "-----------------------------------------------" << endl;

    cout << "Enter the Change: ";

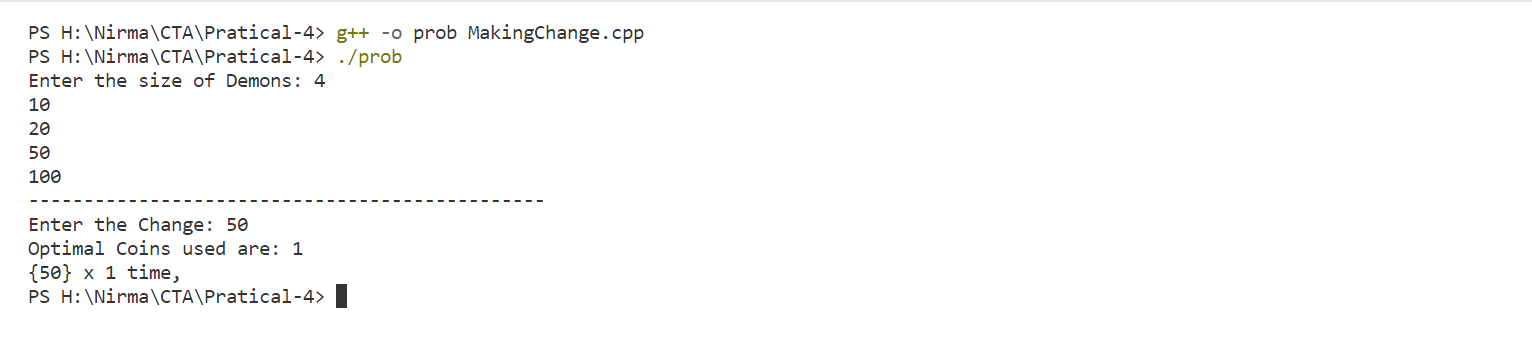
    cin >> change;

    MakingChangeProblem(coins, change, n);

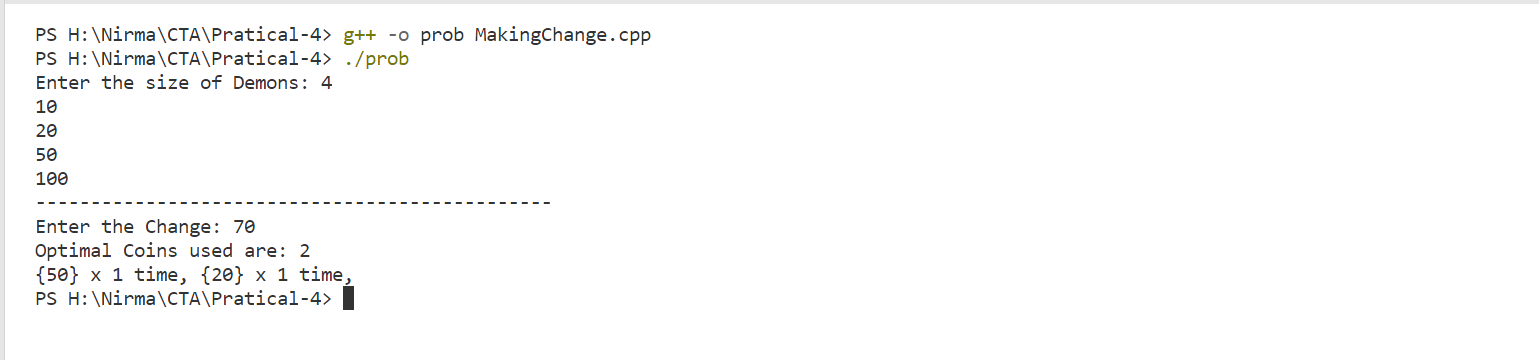
    return 0;

}

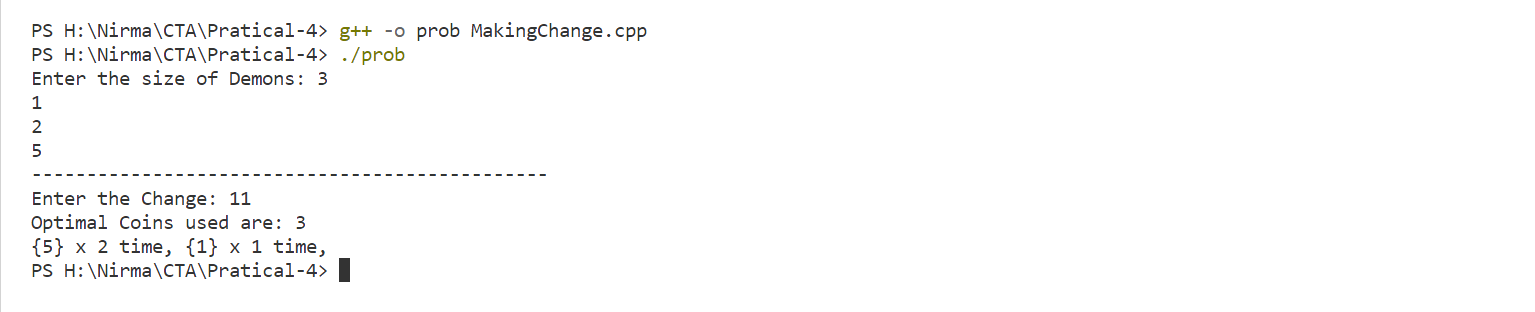
**Output –**

**Test Case - 1**

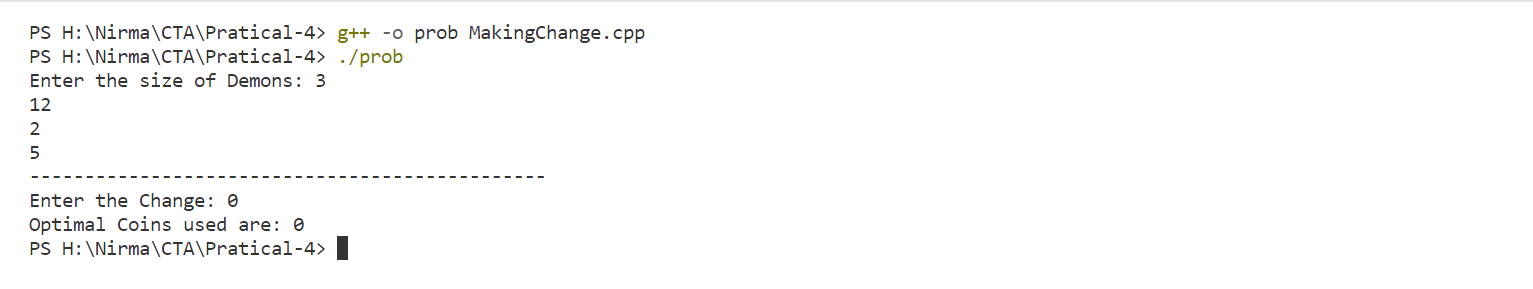
**Test Case – 2**

****

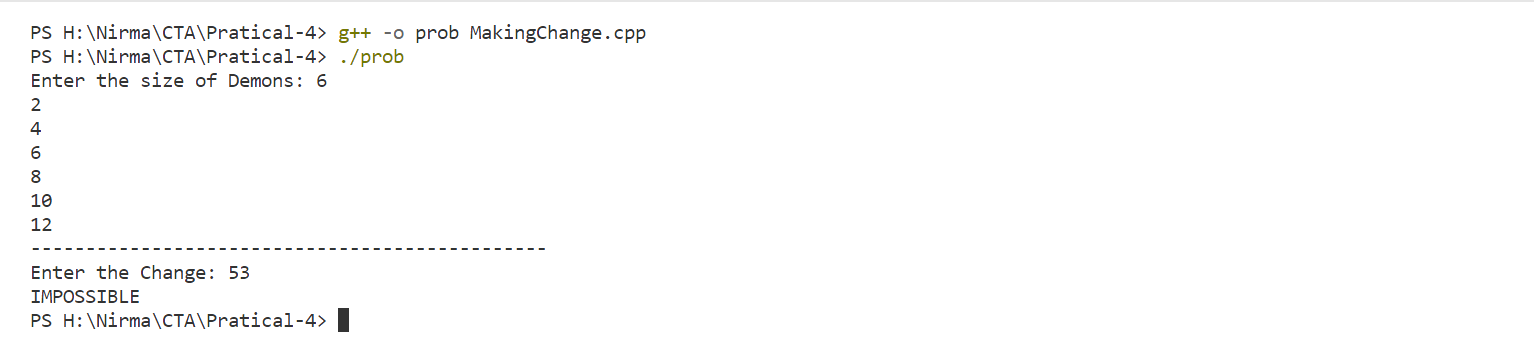
**Test Case – 3**

****

**Test Case – 4**

****

**Test Case – 5**

****