Name: Sai Gulve

PRN No: 202201040145

Batch: A4

Roll No: 63

Problem Statement:

Design and implement a coin change algorithm using the Greedy Method to determine the minimum number of coins required for a given amount. Measure the time required to compute the change.

Brief About the Problem:

The Coin Change Problem is a fundamental problem in computer science, commonly used in financial applications, vending machines, and currency exchange systems. Given a set of denominations, the goal is to make the required amount using the fewest number of coins.

The Greedy Method is an efficient approach to solve this problem by always selecting the largest available denomination that does not exceed the remaining amount. This method works optimally when the given denominations follow a specific structure, such as standard currency systems.

However, the Greedy approach is not always optimal for arbitrary coin sets, making it essential to analyze its efficiency for different inputs.

Proposed Solution:

The algorithm follows these steps:

1. **Sort the denominations in descending order (implemented manually using Selection Sort).**
2. **Iteratively pick the largest possible coin that does not exceed the remaining amount.**
3. **Subtract the coin value from the total amount and repeat until the amount reaches zero.**
4. **Measure the execution time to analyze the efficiency of the solution.**

Time Complexity Analysis:

- **Selection Sort: O(n2)O(n^2)O(n2) (for sorting the coin denominations)**
- **Greedy Selection: O(n)O(n)O(n) (for iterating through the coins and making change)**
- **Overall Complexity: O(n2)O(n^2)O(n2), dominated by sorting**

Since the number of denominations (n) is usually small in real-world scenarios, this complexity is acceptable.

Code :-

```
#include <bits/stdc++.h>
using namespace std;

// Function to implement Selection Sort (Descending Order)
void selectionSort(int arr[], int n) {
   for (int i = 0; i < n - 1; i++) {
      int max_idx = i;
      for (int j = i + 1; j < n; j++) {
         if (arr[j] > arr[max_idx])
            max_idx = j;
      }
      // Swap the found maximum element with the first element
      swap(arr[i], arr[max_idx]);
   }
}

// Function to find the minimum number of coins
void minCoins(int coins[], int n, int amount) {
   selectionSort(coins, n); // Sort coins in descending order

   int result[100]; // Array to store selected coins
   int count = 0;   // Total number of coins used

   cout << "Coins used to make change for " << amount << ": ";
   for (int i = 0; i < n; i++) {
      while (amount >= coins[i]) {
         amount -= coins[i];
         result[count++] = coins[i]; // Store the coin
      }
   }
```

```
    if (amount != 0) {
        cout << "\nCannot make exact change with given denominations.\n";
        return;
    }

    // Print the result
    for (int i = 0; i < count; i++)
        cout << result[i] << " ";

    cout << "\nMinimum coins required: " << count << endl;
}

// Driver code
int main() {
    int n, amount;

    cout << "Enter the number of coin denominations: ";
    cin >> n;

    int coins[n];
    cout << "Enter the coin denominations: ";
    for (int i = 0; i < n; i++)
        cin >> coins[i];

    cout << "Enter the amount: ";
    cin >> amount;

    minCoins(coins, n, amount);

    return 0;
}
```

Output :-

```
Enter the number of denominations: 10
Enter the denominations: 1
2
10
5
1
2
10
20
5
10
Enter the amount to make change for: 34
Coins used to make change for 34: 20 10 2 2
Minimum number of coins required: 4
Execution time: 11995 microseconds

--------------------------------
Process exited after 29.45 seconds with return value 0
Press any key to continue . . .
```