Name	Surti Keyur
Roll No.	25MCE026
Branch	M.Tech (CSE)

Experiment 4

Aim: To implement the Make-Change problem using the Greedy approach

Overview

The task is to determine the **minimum number of coins/notes** needed to make a given amount using available denominations.

In this practical, we implement the solution using the **Greedy approach**, which works by always choosing the largest possible denomination that does not exceed the remaining amount. This strategy is efficient and works optimally for most real-world currency systems (like Indian Rupees or US Dollars).

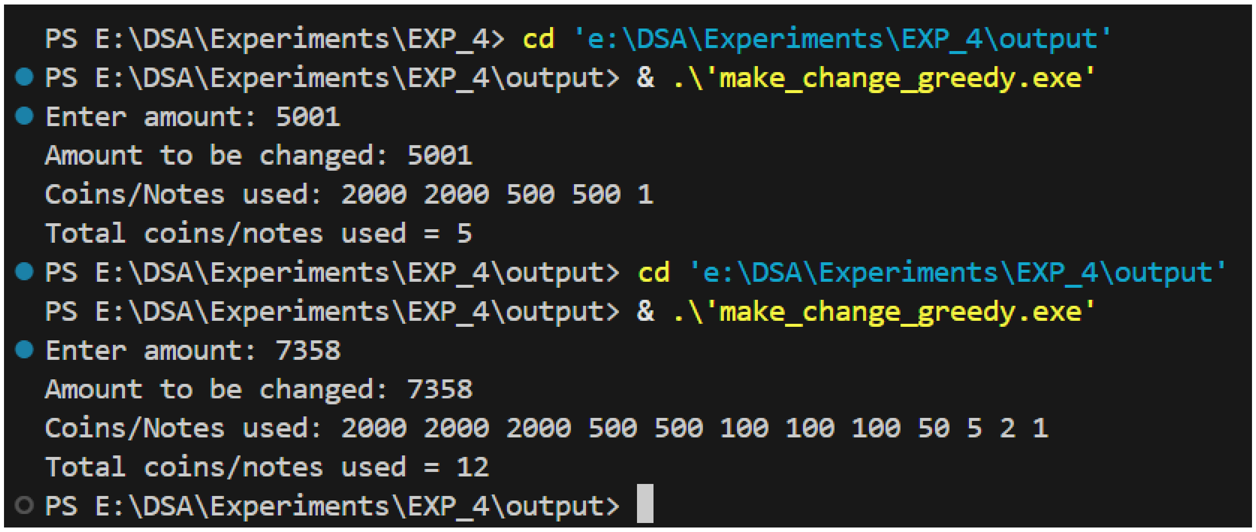
This experiment helps to:

- Understand the concept of **Greedy algorithms**.
- Learn how to apply Greedy logic to real-world problems.

Source Code

```
#include <iostream>
#include <algorithm>
#include <vector>
using namespace std;

// Function to find minimum number of coins/notes
void makeChange(int amount, vector<int> denominations) {
    // Sort denominations in descending order
    sort(denominations.rbegin(), denominations.rend());

    cout << "Amount to be changed: " << amount << endl;
    cout << "Coins/Notes used: ";
    int totalCoins = 0;
    for (int coin : denominations) {
        if (amount >= coin) {
            int count = amount / coin;        // how many coins of this type
            amount -= count * coin;           // reduce the remaining amount
            totalCoins += count;              // add to total coin count

            for (int i = 0; i < count; i++) {
                cout << coin << " ";
            }
        }
    }

    cout << "\nTotal coins/notes used = " << totalCoins << endl;
}

int main() {

    vector<int> denominations = {1, 2, 5, 10, 20, 50, 100, 500, 2000};

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

    makeChange(amount, denominations);

    return 0;
}
```

Output



```
PS E:\DSA\Experiments\EXP_4> cd 'e:\DSA\Experiments\EXP_4\output'
PS E:\DSA\Experiments\EXP_4\output> & .\'make_change_greedy.exe'
Enter amount: 5001
Amount to be changed: 5001
Coins/Notes used: 2000 2000 500 500 1
Total coins/notes used = 5
PS E:\DSA\Experiments\EXP_4\output> cd 'e:\DSA\Experiments\EXP_4\output'
PS E:\DSA\Experiments\EXP_4\output> & .\'make_change_greedy.exe'
Enter amount: 7358
Amount to be changed: 7358
Coins/Notes used: 2000 2000 2000 500 500 100 100 100 50 5 2 1
Total coins/notes used = 12
PS E:\DSA\Experiments\EXP_4\output>
```

Time and Space Complexity

- **Time Complexity:**
  - Sorting denominations: O(d log d) (but if already sorted → O(d))
  - Iterating through denominations: O(d)
  - Overall: **O(d log d)** or **O(d)** if predefined sorted
- **Space Complexity:**
  - Uses only a few variables and the denominations array.
  - **O(d)** for denominations, **O(1)** extra space.

Conclusion

The **Greedy algorithm** for the **Make-Change problem** is simple and efficient when applied to real-world currency systems. It provides an optimal solution in **O(d)** time by always selecting the largest possible denomination first. Although Greedy may fail for arbitrary coin sets, it works perfectly in practice for commonly used denominations, making it a practical and effective method.