The Change-Making Problem

Belu Florinel Gabriel
CR 1.1A (Hobbits)

Year 2017 - 2018
Semester II

Chapter 1

Problem Statement:

The Change-Making Problem : when given an amount of money calculate the least number of coins of given denominations needed to make that amount.

Chapter 2

Algorithms:

I solved the change-making problem by implementing two diffrent algorithms.

The Backtracking Method

The algorithm uses a number of functions that have been placed into a library that represent the process they are used for:

- Functions used for reading the denominations.
- Function used to calculate the number of coins to make an amount.
- Function to display the results.

Input

The input represents the number of coin types (example : 4),the denominations (example : 1,2,5,10) and the amount (example : 25).After that you only have to enter the amout if you want to continue the process.When you are done you can enter -1 and the process will stop.

Output

In the output you are show the number of coins of eache denomination neede to make the amount.
Example:
The number of coin types is:4
The 1 denomination is:1
The 2 denomination is:2
The 3 denomination is:5
The 4 denomination is:10
Please enter an integer amount between 0-10000 or -1 to exit:25
0 number of 1 denomination coins used
0 number of 2 denomination coins used
1 number of 5 denomination coins used
2 number of 10 denomination coins used

Determinating the number of coins.

This function is used to calculate the number of coins after the user has given the number of coin types,the denominations and the amount of money.

Data: The amount of money,an iterator,the denominations and the number of coins.
Result: The number of coins needed.
if *the amount of money is less than the samllest denomination* **then**
  return the number of coins;
else
  if *the amount of money == denomination[iterator]* **then**
    return number of coins increased by 1 ;
  else
    if *the amount of money is greater than denomination[iterator]* **then**
      increase the number of coins count by 1;
      return this function with amount - denomination[iterator];
    else
      return this fuction with iterator - 1;
    end
  end
end

Algorithm 1: Determinating the number of coins.

Display the results.

This function asks the user to enter a number that represents the amount of money.If the number is compatible with some conditions then the fucntion will call the recursive function and display the results,otherwise it will ask the user to indtroduce another amount of money or to introduce -1 to exit.

Data: The denominations,an iterator,the number of coin types and the number of coins.
Result: The display of the number of coins needed.
Get the amount of money from the user.;
while *The user doesn't introduce -1.* **do**
  if *The user introduces an accepted amount of money.* **then**
    call the recursive function and display the results;
  else
    Ask the user again for an amount of money.;
  end
end

Algorithm 2: Display the results.

Reading the denominations.

This function asks the user to introduce a number of denominations equal with the number of coin types and saves them in a pointer to a vector.

Data: The denominations and the number of coin types.
Result: Gets all the denominations from the user.
for *Each coin type.* **do**
  Read the denomination from the user.;
end

Algorithm 3: Reading the denominations.

Reading the number of coin types.

This function asks the user to introduce a number that represents the number of coin types.It save it in an auxiliary that is then returned.

Data: No parametres.
Result: Returns the number of coins types given by the user.
Ask the user to introduce the number of coin types;
Save it in an auxiliary;
return the auxiliary;

Algorithm 4: Reading the number of coin types.

Main.c

This function is used to initialise and declare the important parameters and to call the functions.

Data: The declaration of the parameters used for the functions calls.
Initializing the parameters;
Making the functions calls;
Free the pointers used;

Algorithm 5: Main.c.

The Greedy Algorithm Method

The algorithm uses a number of functions that have been placed into a library that represent the process they are used for:

- Functions used for reading the denominations.
- Function used to calculate the number of coins to make an amount and to display the results.

Input

The input represents the number of coin types (example : 4),the denominations (example : 3,4,6,9) and the amount (example : 88).After that you only have to enter the amout if you want to continue the process.When you are done you can enter -1 and the process will stop.

Output

In the output you are show the number of coins of eache denomination neede to make the amount.
Example:
The number of coin types is:4
The 1 denomination is:3
The 2 denomination is:4
The 3 denomination is:6
The 4 denomination is:9
Please enter an integer amount between 0-10000 or -1 to exit:88
9 number of 9 denomination coins used
1 number of 6 denomination coins used
0 number of 4 denomination coins used
1 number of 3 denomination coins used

Reading the denominations.

This function asks the user to introduce a number of denominations equal with the number of coin types and saves them in a pointer to a vector.

Data: The denominations and the number of coin types.
Result: Gets all the denominations from the user.
for *Each coin type.* **do**
  Read the denomination from the user.;
end

Algorithm 6: Reading the denominations.

Reading the number of coin types.

This function asks the user to introduce a number that represents the number of coin types.It save it in an auxiliary that is then returned.

Data: No parametres.
Result: Returns the number of coins types given by the user.
Ask the user to introduce the number of coin types;
Save it in an auxiliary;
return the auxiliary;

Algorithm 7: Reading the number of coin types.

Display and calculate the results.

This function asks the user to enter a number that represents the amount of money.If the number is compatible with some conditions then the fucntion will determine the number of coins using a simple while that reduces the amount given by the user by the biggest denomitaion possible until it reaches 0 and displays the results,otherwise it will ask the user to indtroduce another amount of money or to introduce -1 to exit.

```
Data: The denominations,the counter for the number of coin types and
      the number of coins.
Result: The display and the calculation of the number of coins needed.
Get the amount of money from the user.;
while The user doesn't introduce -1. do
    if The user introduces an accepted amount of money. then
        for Each denomination in descending order do
            while The amount is greater than denominations[iterator] do
                Reduce the amount by denominations[iterator];
            end
            Display the results;
        end
    else
        Ask the user again for an amount of money.;
    end
end
```

Algorithm 8: Display and calculate the results.

Main.c

This function is used to initialise and delcare the important parameters and to call the functions.

```
Data: The declaration of the parameters used for the functions calls.
Initializing the parameters;
Making the functions calls;
Free the pointers used;
```

Algorithm 9: Main.c.

Bibliography

[1] GeeksForGeeks
https://www.geeksforgeeks.org/

[2] Stack Overflow
https://stackoverflow.com/

[3] Git Hub
https://github.com/