**Danyyal Ahmed Khalid**

**BSCS-3B**

**The Lab**:

Implementing the Coin Change problem using Greedy as well as a Dynamic Algorithm, with combinations of 1, 5, 10, and 25.

**Methodology**:

The Greedy function works iteratively, and evaluates the best fit. This is done by take the highest coin value and subtracting it from the remaining.

The Dynamic methodology I implemented using the Pseudo Code in the slides.

**Code**:

package coinchange;

import java.util.\*;

public class coinchange

{

public static int Grdy(int total)

{

int one = 0;

int five = 0;

int ten = 0;

int twentyfive = 0;

while(total > 0)

{

if(total >= 25)

{

//Checking 25 coin

twentyfive ++;

total = total-25;

}

else

if(total >= 10)

{

//Checking 10 coin

ten ++;

total = total- 10;

}

else

if(total >= 5)

{

//Checking 5 coin

five ++;

total = total- 5;

}

else

{

//Checking 1 coin

one++;

total = total -1;

}

}

//System.out.println("Ones: "+one);

//System.out.println("Fives: "+five);

//System.out.println("Tens: "+ ten);

//System.out.println("25: "+ twentyfive);

return (one+five+ten+twentyfive);

}

public static int[] DP1(int[] coins, int total)

{

int[] temp = new int[total + 1];

temp[0] = 0;

for (int j = 1; j < total+1; j++)

{

temp[j] = 1000000000; //Inifinte

for (int i = 0; i < coins.length; i++)

{

//All coins checking

if(j >= coins[i] && (1+temp[j-coins[i]]) < temp[j])

{

temp[j] = 1 + temp[j-coins[i]]; //Memoization

}

}

}

return temp;

}

public static void main(String[] args)

{

int temp = 0;

int coins[] = {1,5,10,25}; // All coins

int total; //total to make

int gcoins;

int ansDP[];

while(temp < 10) // For 10 random totals

{

Random farig = new Random(); //Random Object

total = farig.nextInt() % 1000000; //Random total

//System.out.println(total);

if(total > 0)

{

gcoins = Grdy(total); //Calling Grdy

ansDP = DP1(coins, total); //Calling DP

System.out.println("total " + total); //total Was

System.out.println("Num of Coins Grdy: " + gcoins);//Grdy Result

System.out.println("Num of Coins DP: " + ansDP[total]);//DP Result

temp++;

}

}

}

}

Analysis:

Let N be the amount to be paid. Let the optimal solution be P=A\*10 + B\*5 + C. Clearly B≤1 (otherwise we can decrease B by 2 and increase A by 1, improving the solution). Similarly, C≤4. Let the solution given by the greedy algorithm be P=a\*10 + b\*5 + c. Clearly b≤1 (otherwise the algorithm would output 10 instead of 5). Similarly c≤4.

From 0≤ C≤4 and P=(2A+B)\*5+C we have C=P mod 5.

Similarly c=P mod 5, and hence c=C. Let Q=(P-C)/5.

From 0≤ B≤ 1 and Q = 2A + B we have B=Q mod 2.

Similarly b=Q mod 2, and hence b=B.

Thus a=A, b=B, c=C, i.e., the solution given by the greedy algorithm is always optimal.

**So there is no value for SET {1,10,5,25} so that optimal is better.**

Link to Github: