# CSCE 5150 – Analysis of Computer Algorithms

## Programming Assignment 3 – Greedy Algorithms

**Neha Goud Baddam**
**11519516**

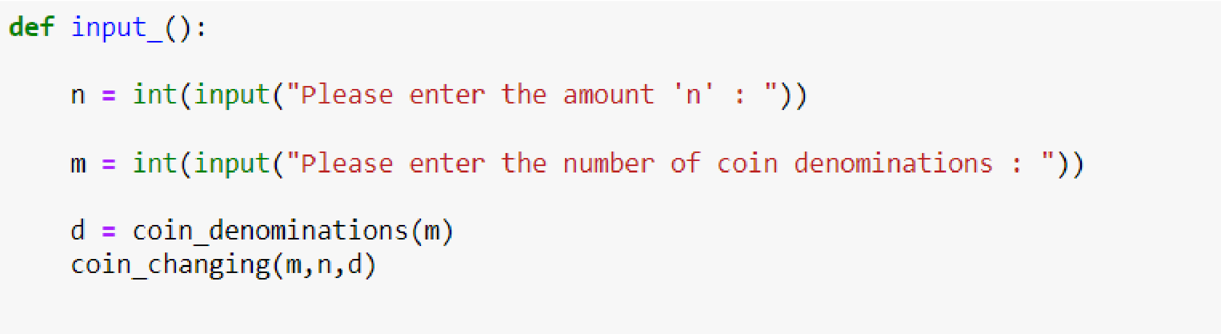
## Coin-Changing problem

**Definition:**

1. The coin-changing problem finds the minimum number of coins that must be generated based on the given denominators, which adds up to the input amount.

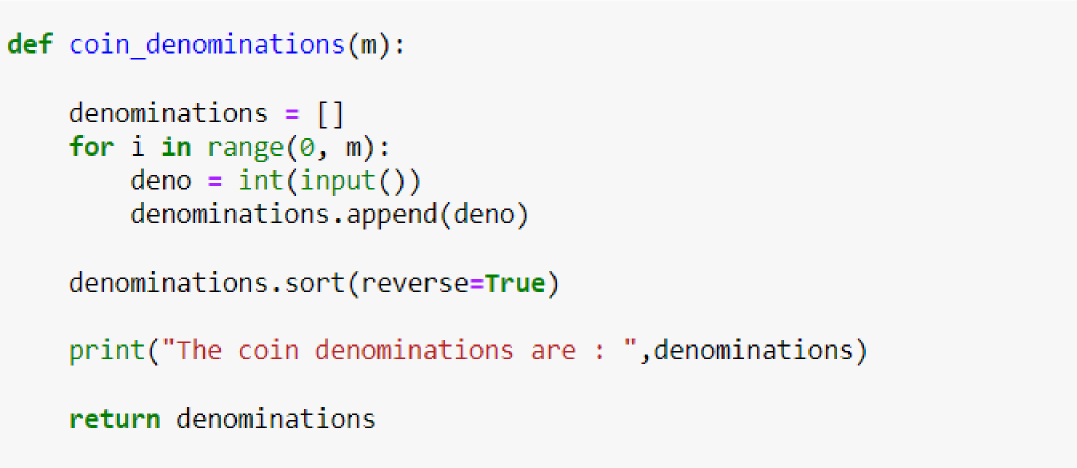
   **Coin-changing Algorithm:**

   **i.** Firstly, the function input is called to take the amount (n cents) and the number of coin denominations (m denominators) as input. Then we further call the function "coin_denominations" to get the denominators and the function "coin_changing" to calculate the solution.



```
def input_():

    n = int(input("Please enter the amount 'n' : "))

    m = int(input("Please enter the number of coin denominations : "))

    d = coin_denominations(m)
    coin_changing(m,n,d)
```

   **ii.** The function "coin_denominations" takes all the denominators and stores it in a list. It also sorts the list in ascending order.



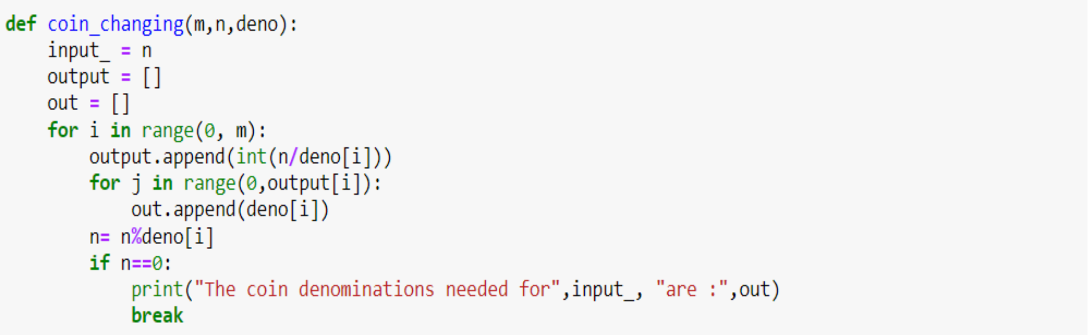
```
def coin_denominations(m):

    denominations = []
    for i in range(0, m):
        deno = int(input())
        denominations.append(deno)

    denominations.sort(reverse=True)

    print("The coin denominations are : ",denominations)

    return denominations
```

# CSCE 5150 – Analysis of Computer Algorithms

## Programming Assignment 3 – Greedy Algorithms

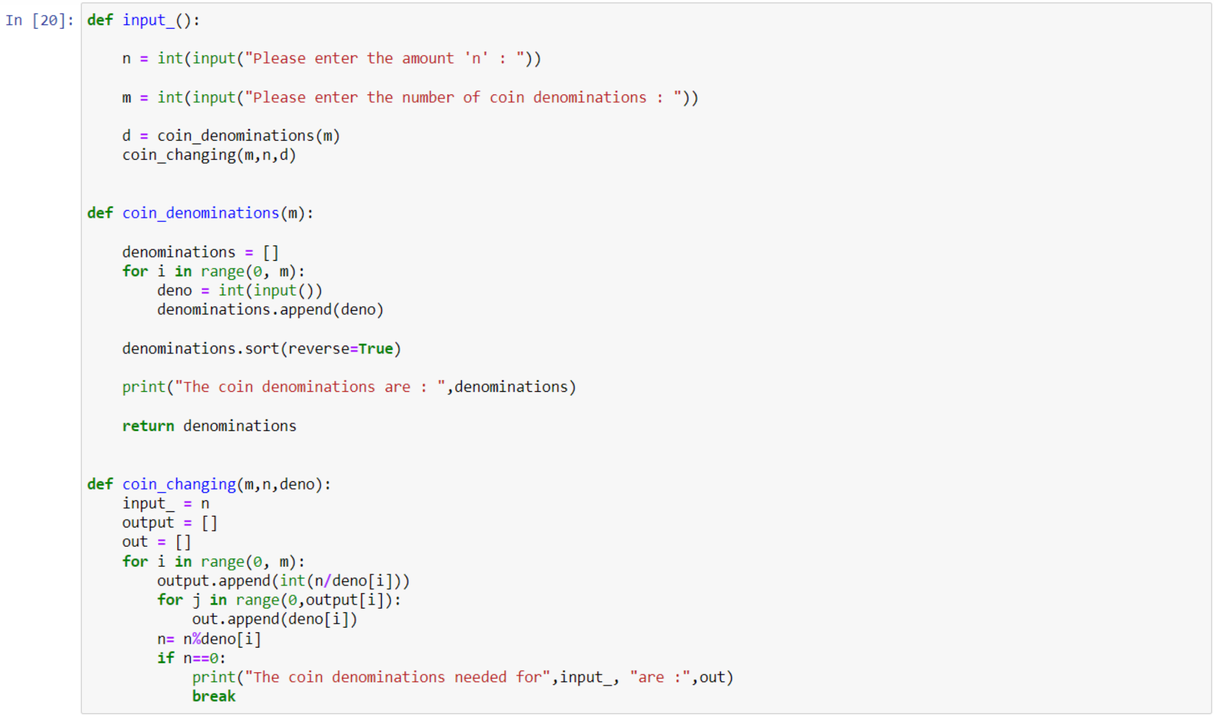
**Neha Goud Baddam**
**11519516**

**iii.** The function "coin_changing" divides the amount with the highest denominator to calculate the number of coins with the highest denominator needed. The remainder is stored and is further divided by the next highest denominator to calculate the number of coins with the next highest denominator needed. This process is repeated until the remainder is zero, and the output is printed.



```
def coin_changing(m,n,deno):
    input_ = n
    output = []
    out = []
    for i in range(0, m):
        output.append(int(n/deno[i]))
        for j in range(0,output[i]):
            out.append(deno[i])
        n= n%deno[i]
        if n==0:
            print("The coin denominations needed for",input_, "are :",out)
            break
```

**<u>Pseudocode of the greedy algorithm for the coin-changing problem, with an amount n and coin denominations d1 > d2 > d3 > . . . > dm as its input. (Hint. You may use integer divisions in your algorithm)</u>**



```
In [20]: def input_():

             n = int(input("Please enter the amount 'n' : "))

             m = int(input("Please enter the number of coin denominations : "))

             d = coin_denominations(m)
             coin_changing(m,n,d)


         def coin_denominations(m):

             denominations = []
             for i in range(0, m):
                 deno = int(input())
                 denominations.append(deno)

             denominations.sort(reverse=True)

             print("The coin denominations are : ",denominations)

             return denominations


         def coin_changing(m,n,deno):
             input_ = n
             output = []
             out = []
             for i in range(0, m):
                 output.append(int(n/deno[i]))
                 for j in range(0,output[i]):
                     out.append(deno[i])
                 n= n%deno[i]
                 if n==0:
                     print("The coin denominations needed for",input_, "are :",out)
                     break
```

# CSCE 5150 – Analysis of Computer Algorithms

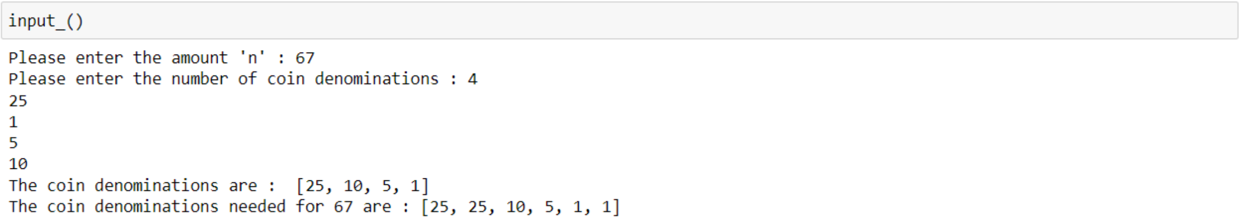
## Programming Assignment 3 – Greedy Algorithms

**Neha Goud Baddam**
**11519516**

```
In [21]: input_()
```

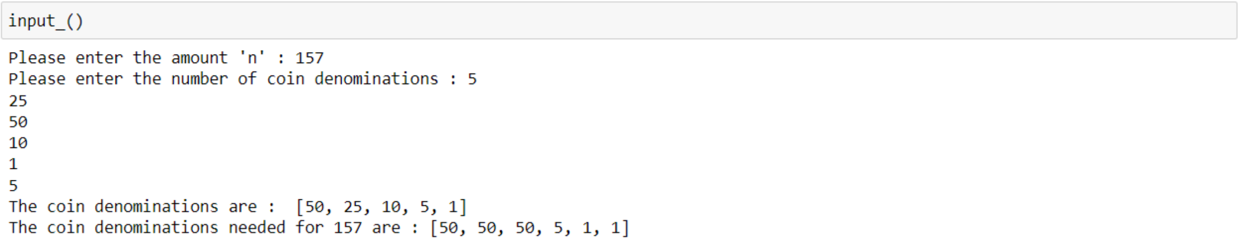
**Execution/ Output:**

1. **Verify the correctness of your program by solving the problem of a greedy algorithm to make a change of the amount of 67 cents (n=67 cents) using 4-coin denominations (m=4) consisting of quarters (d1=25 cents), dimes (d2=10 cents), nickels (d3=5 cents), and pennies (d4=1 cent).**



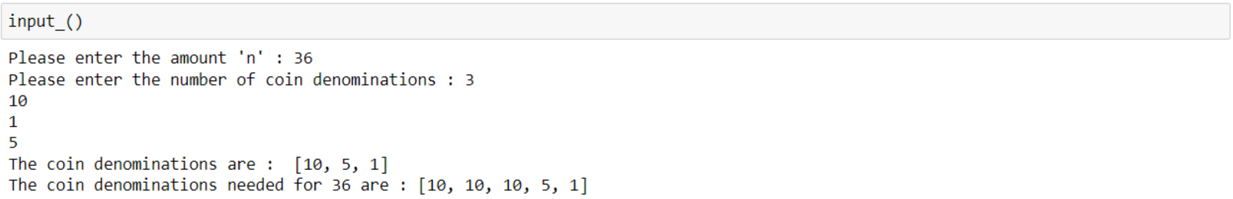
```
input_()

Please enter the amount 'n' : 67
Please enter the number of coin denominations : 4
25
1
5
10
The coin denominations are :  [25, 10, 5, 1]
The coin denominations needed for 67 are : [25, 25, 10, 5, 1, 1]
```

2. **Greedy algorithm to make a change of the amount of 157 cents (n=157 cents) using 5-coin denominations (m=5) consisting of half-dollar(d1= 50 cents), quarters (d2= 25 cents), dimes (d3=10 cents), nickels (d4=5 cents), and pennies (d5=1 cent).**



```
input_()

Please enter the amount 'n' : 157
Please enter the number of coin denominations : 5
25
50
10
1
5
The coin denominations are :  [50, 25, 10, 5, 1]
The coin denominations needed for 157 are : [50, 50, 50, 5, 1, 1]
```

3. **Greedy algorithm to make a change of the amount of 36 cents (n=36 cents) using 3-coin denominations (m=3) consisting of quarters dimes (d3=10 cents), nickels (d4=5 cents), and pennies (d5=1 cent).**



```
input_()

Please enter the amount 'n' : 36
Please enter the number of coin denominations : 3
10
1
5
The coin denominations are :  [10, 5, 1]
The coin denominations needed for 36 are : [10, 10, 10, 5, 1]
```

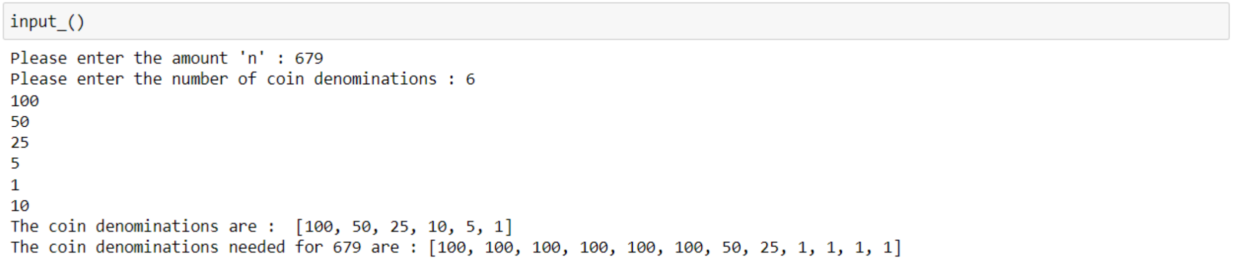
4. **Greedy algorithm to make a change of the amount of 679 cents (n=679 cents) using 6-coin denominations (m=6) dollar(d1= 100 cents), half-dollar(d2= 50 cents), quarters (d3= 25 cents), dimes (d4=10 cents), nickels (d5=5 cents), and pennies**

# CSCE 5150 – Analysis of Computer Algorithms

# Programming Assignment 3 – Greedy Algorithms

## Neha Goud Baddam
## 11519516

### (d6=1 cent).



```
input_()
```

```
Please enter the amount 'n' : 679
Please enter the number of coin denominations : 6
100
50
25
5
1
10
The coin denominations are :  [100, 50, 25, 10, 5, 1]
The coin denominations needed for 679 are : [100, 100, 100, 100, 100, 100, 50, 25, 1, 1, 1, 1]
```