Zaid Dandia
2021719
CS-478
Assignment #2.

1 A) $dp[j] = \min(dp[j-coins[i]]+1)$.

Parameters :- $dp[j]$ → minimum numbers of coins
coins is the list of available coin denominations.

For each amount j, the recurrence will try using each coin(coin[i]
Then find the minimum value of $dp[j-coins[i]]+1$. Adding
1 will represent one coin of coins[i]
Base case will be $dp[0]=0$ meaning 0 coins are needed

Parameters settings :-
Use dp[n], which gives the min number of coins
needed.

B) Algorithm CoinChange (coins, n) :

```
dp[0] = 0
~~dp[i] = INF~~
for (i=1 : n)
    dp[i] = INF
for (i = 0 : n)
    coinUsed[i] = -1

for j=1 to n;
    for each coin in coins:
        if ( j >= coin).
            If (dp[j - coin] + 1 < dp[j]).
                dp[j] = dp[j-coin]+1
                coinUsed [j] = coin.
change[length(coins)] = 0.
j = n
while j > 0
```

```
coin = coinUsed[j].
  index = coins.index(coin)
  change[index] += 1
   j -= coin
return change.
```

2A)
```
MinBillsRecursive(bills, n):
  if(n = 0)
    return 0
  if(n<0)
    return INF

  minBills = INF
  for each bill in bills:
     result = MinBillsRecursive(bills, n-bill)
     if(result != INF)
        minBills = min(minBills, result+1)

  return minBills
```

B)
```
MinBillsDP(bills, n)
  dp[0] = 0
  for(i = 1; i<n)
    dp[i] = INF

  For j = 1 to n
    For each bill in bills:
      if j >= bill
        dp[j] = min(dp[j], dp[j-bill]+1).

  return dp[n]
```

bills = [1, 4, 7, 13, 28, 52, 91, 365]

dp[90] = 6 (1 => 52, 1 => 28, 1 => 7, 3 => 1)

c) bills = [1, 4, 7, 13, 28, 52, 91, 365]   n = ~~2~~ 416

~~Greedy => One 13 (24-13=11)~~
~~one 7 (11-7=4)~~
~~one 4 (4-4=0)~~

~~Optimal solution => Two 12's (24-24=0)~~
~~one 416~~

Greedy = [365, 28, 13, 7, 1, 1, 1]

Optimal = [91, 91, 91, 91, 52]