1)

We are using the dinamic algorithm method so we don't need to calculate all subset over an over again, We are use the alredy calculated result.

a)

$$F(n) = max(c_i + f(n-1), F(n))$$

$$F(0) = 0$$

Decrease and conquer algorithm

$$T(n) = T(n-1) + 1 \quad \boxed{\in \Theta(n)}$$

b) In previous implementation I used 3 method to calculate most profit cluster

① find AllSubset(); → $\Theta(n^2)$. copy() => $\Theta(n^3)$

②③ calculateProfit(_subset); → $\Theta(n)$

② find Most Profitable Cluster (subsets); → $\underbrace{n.\text{ calculateProfit();}}_{n.n}$ => $\Theta(n^2)$

$$T(n) = \Theta(n^3)$$

But in this case the time complexity much more decrese

but space complexity increase

Old	New
$T(n) = \Theta(n^3)$	$T(n) = \Theta(n)$
$S(n) = 0$	$S_p(n) = \Theta(n+1)$

2)

$$f(n) = max(max\_val, prices[j] + F(i-j-1))$$

$i = 1$ to n target range

J = coins lesser than target

In this problem:

```
for i range (1, n+1):
    for j in range(i):
        F(n)=max(max, c_i + F(i-j-1)
```

$$\sum_{i=1}^{n+1} \sum_{j=0}^{i} 1 = \sum_{i=1}^{n+1} i =$$

$$\Rightarrow \frac{(n+1)(n+2)}{2}$$

$$\boxed{\in \Theta(n^2)}$$

Scanned with CamScanner

3) We are using knapsack algoritm for this problem.

we are applying greedy approach for chess and calculate the type of chees per kg for each chees

values = $\frac{\text{prices}}{\text{weights}}$ and choose the most valuable cheese

```
for value in values
      if W >0;
          if W > weights [i]
                getlist [i] = 1
          else
              getlist [i] = W/weight
      W -= weight [i]
```

$\overbrace{\quad}^{\text{getlist}}$

$\underbrace{[0,0,0\ ....,0]}$

If we choose the nth element we doing 1 if we cut cheese 2/3 like real number

This method is take n time complexity $\boxed{T\ \Theta(n)}$

and sorting use $\boxed{T(n\log n)}$ complexity $S(n) = \Theta(n)$

4) In this problem for greedy approach calculate the lecture times

lecture times [ ] = Li[finish - start]

and we are pick the lectures which it has least lecture time

then picking the least to most lectures.

```
for lect in lectures:
     if conflict ( ourList, lect) == False:
          ourList.append(lect).
```

$\xrightarrow{n}$ Iterating the over our choosed $\underbrace{\text{lectures.}}_{(m)}$

$\boxed{T(n) = \Theta(mn)}$

Muhammed Bedir ULUCAY

1801042697