HW5 $\boxed{[c_0, c_1, \ldots, c_n]}$

1) $F(n) = \max\left(F(n), c_n + F(n-1)\right)$ for $n > 0$

$$F(0) = c_0$$

For every array element we have two choices:

1) We can start a new sequence
2) We can add current element to lastly created sequence

which one offers the most profit, we choose it.

The algorithm travels the array only 1 time, starting from index 1

$$f(n) = \sum_{i=1}^{n-1} 1 = n-1 \in \Theta(n)$$

$$f(n) \in \Theta(n)$$

space complexity = n sized array created, $\boxed{\Theta(n)}$

2) $F(n) = \max(c_n, F(n-k) + F(k))$ $n > k$

$n \geq 1$

Starting from small sized candies for every candy, best price is found by looking at the candies whose sizes are smaller. By putting together 2 of them, wanted candy is created.

Analyze: $\sum_{i=1}^{n-1} \sum_{j=1}^{\lceil \frac{i}{2} \rceil} 1 = \underset{i=2}{1} + \underset{i=3}{1} + \underset{i=4}{2} + \underset{i=5}{2} + \underset{i=6}{3} + \underset{i=7}{3} + 4 \ldots$

2

3

$\ldots \underset{i=n-1}{\lceil \frac{n}{2} \rceil}$

$= 1 + 1 + 2 + 2 + \ldots + \lceil \frac{n}{2} \rceil + \lceil \frac{n}{2} \rceil \cdot (n \% 2 == 1)$

$= 2(1 + 2 + \ldots + \lceil \frac{n}{2} \rceil) - (\lceil \frac{n}{2} \rceil \cdot (n \% 2))$

$= 2\left(\frac{\lceil \frac{n}{2} \rceil \cdot (\lceil \frac{n}{2} \rceil + 1)}{2}\right) - (\lceil \frac{n}{2} \rceil \cdot (n \% 2))$

parts that does not change the complexity classes are ignored.

$\frac{n}{2} \cdot \left(\frac{n}{2} + 1\right) = \boxed{\frac{n^2}{4} + \frac{n}{2} \in \Theta(n^2)}$

CamScanner ile tarandı

3) Algorithm calculates the $price_i / weight_i$ for every element and it finds the element which gives the most rate. Checks if weight of the element is bigger than the remaining weight. if it is, algorithm does not takes all the cheese but takes part of it so that, there is no remaining space. if remaining weight is more than the weight of cheese, algorithm takes all of it. After that, weight of the cheese is made -1 so that algorithm does not takes it again. Until capacity is filled, algorithm calls itself again and again.

Algorithm also checks that if the weight of the found element is less than zero. If it is, it means that all of the cheese is taken. Algorithm returns taken profit so far.

Best case: There is no recursive call, wanted length is filled with the first found element.

$$T(n) = \sum_{i=1}^{n-1} 1 = 1 + 2 + \dots + n-1 = \frac{(n-1).n}{2} = \frac{n^2 - n}{2} \in \Theta(n)$$

Worst case: If wanted weight is more then weight of all cheeses, algorithm is called n times.

$$T(n) = \sum_{i=0}^{n-1} \sum_{j=0}^{n} 1 = \sum_{i=0}^{n-1} n = n.n \in \Theta(n^2)$$

Average case: For every number from 1 to n, assume that possibility of calling recursively this number of times is equal.

$$T(n) = \frac{1}{n}\left(\sum_{i=1}^{1} n + \sum_{i=1}^{2} n + \dots + \sum_{i=1}^{n-1} n\right) = \frac{1}{n}\left(n + 2n + 3n + \dots + n^2\right) =$$

$$\frac{1}{\cancel{n}} \cdot \cancel{n}(1 + 2 + \dots + n) = \frac{n.(n+1)}{2} \in \Theta(n^2)$$

4) This greedy algorithm takes the course which starts earlier than other remaining courses, and starts after the last course ends. If there is no other course which starts after the last course ends, program ends and returns number of courses.

This algorithm has downside which is, the algorithm takes the course which starts most early (of course if the course is proper to be taken). This situation may cause that maximum number of courses is not reached.

Best case: if the first course taken ends after starts of all other courses, there is only 1 course can be taken. Best case happens.

$$T(n) = \sum_{i=1}^{1} \sum_{j=0}^{n} 1 = n \in \Theta(n)$$

Worst case: If any course does not prevent the student from taking other courses, Worst case happens.

$$T(n) = \sum_{i=1}^{n} \sum_{j=1}^{n} 1 = \sum_{i=1}^{n} n = \underbrace{n+n+\dots+n}_{n \text{ times}} = n.n = n^2 \in \Theta(n^2)$$

Average case: Assume that, for every number between 1 and n, it is equally likely that this number of courses can be taken

$$T(n) = \frac{1}{n} \left(n + 2n + \dots + n^2 \right) = \frac{1}{n} \cdot n \left(1 + 2 + \dots + n \right) = \frac{n.(n+1)}{2} = \frac{n^2+n}{2}$$

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

CamScanner ile tarandı