ClASSMAte

Date ____

Page ____

Homework 7

1] (a) Let OPT(i) be ~~i^{th} item~~ ~~having~~ the largest sum of the contiguous subarray ~~to~~ having i^{th} items.

(b) $OPT(i) = \max(OPT(i-1) + a[i], a(i))$

(c) $s = []$ & Initialize OPT(0) as below.

for i = 1 to n-1

$OPT(i) = \max(OPT(i-1) + a[i], a(i))$

s.append(i).

end for.

(d) $OPT(0) = a[0]$

final answer is OPT(i) which will be having the largest sum and s would be having the corresponding contiguous subarray.

(e) Time complexity is $O(n)$

2] (a) Let OPT(i) be the maximum profit achieved by ~~buying~~ selling a stock on i^{th} day ~~item i~~.

& hold = max. profit achieved if we hold ~~item~~ bought it.

(b) $OPT(i) = \max(OPT(i), \cancel{OPT(i-1)} \text{hold} + price(i) - fee)$

if we sell the stock.

hold = max (hold, OPT(i) - prices[i]) if we buy.

(c) Initialize OPT(i) = 0 and hold = - prices[0].

for i in range (1, len(prices)):
    OPT(i) = max (OPT(i), hold + prices[i] - fee)
    hold = max (hold, OPT(i) - prices[i])
end for.

(d) (i) OPT(i) = 0 and hold = - prices[0]
(ii) OPT(i) is the maximum profit and our final answer.

(e) ~~O(1)~~, O(N)

3] (a) OPT(i, l) is the happiest sequence that ends at the i items and using the i^{th} element for a subsequence of length l.

(b) $OPT(i, l) = \max_{j<i,\, a(j)<a(i)} \left(OPT(j, (l-1)) + l * a[i] \right)$

(c) for i range (len(a)): for j in range (len(a)):
$OPT(i, l) = \max_{j<i \,\&\, a(j)<a(i)} \left(OPT(j, (l-1)) + l * a[i] \right)$

end for.

(d) OPT(0,1) = a[0]

final answer would be OPT(0,l).

~~total~~

(e) Total complexity = $O(n^3)$

as we have $O(n^2)$ entries which takes $O(n)$ time.

4] (a) OPT(m, n) is the max length of m and n where ~~th~~ m = n and contain only 1's.

(b) $$OPT(m,n) = \min\begin{pmatrix} OPT(i-1, j), \\ OPT(i-1, j-1), \\ OPT(i, j-1) \end{pmatrix} + 1$$

(c) OPT[m, n] = all zeros
l = ~~max~~ 0

```
for ~~i~~ i in range(len(m)):
    for j in range(len(n)):
        if a[i-1][j-1] == 1:
            OPT(i, j) = min(OPT(i-1, j), OPT(i-1, j-1),
                            OPT(i, j-1)) + 1
            l = max(l, OPT(i, j))
```

end for.

(d) We initialize OPT(m,n) = 0 for all values and update it as and when we encounter 1. the final value would be present at the botom left corner.

(e) O(mn) is the time complexity as it takes constant time for the evaluation inside the 'for loops'.