IE201 INTERMEDIATE PROGRAMMING

Homework 1 Report

03.12.2022

Tevfik Buğra Türker

2019402120

a)

```
# O(n^2)
def max_profit (price):
    lowest = price[0]
    profit = 0
    for i in range(len(price)):
        if price[i] < lowest:
            lowest = price[i]
        for k in range(i,len(price)):
            if price[k]-lowest > profit:
                profit = price[k]-lowest
    return profit

#print(max_profit(price))
```

b)

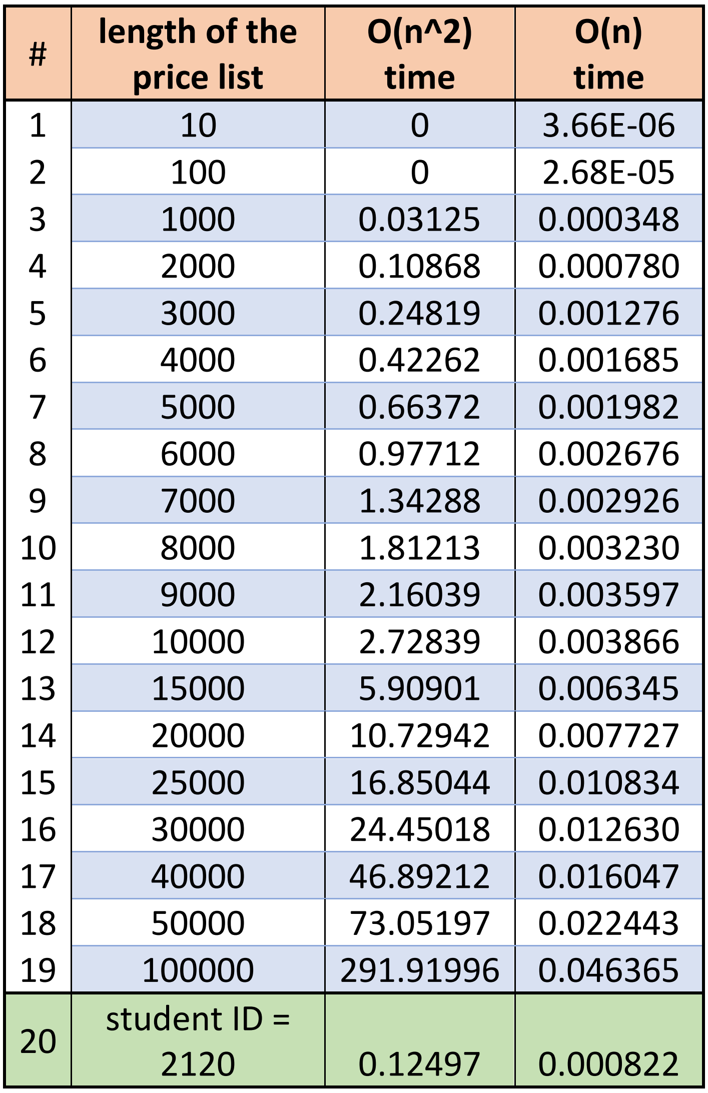
```
 # b)

# O(n)
def max_profit_quick (price):
    min_until = price[0]
    min_until_dict = dict()
    max_from = price[len(price)-1]
    max_from_dict = dict()
    for i in range(len(price)):
        if price[i] < min_until:
            min_until = price[i]
        min_until_dict[i] = min_until
    reversed_price = price[::-1]
    for k in range(len(price)):
        if reversed_price[k] > max_from:
            max_from = reversed_price[k]
        max_from_dict[len(price)-k-1] = max_from
    profit = 0
    for i in range(len(price)):
        if max_from_dict[i]-min_until_dict[i] > profit:
            profit = max_from_dict[i]-min_until_dict[i]
    return profit

#print(max_profit_quick(price))
```

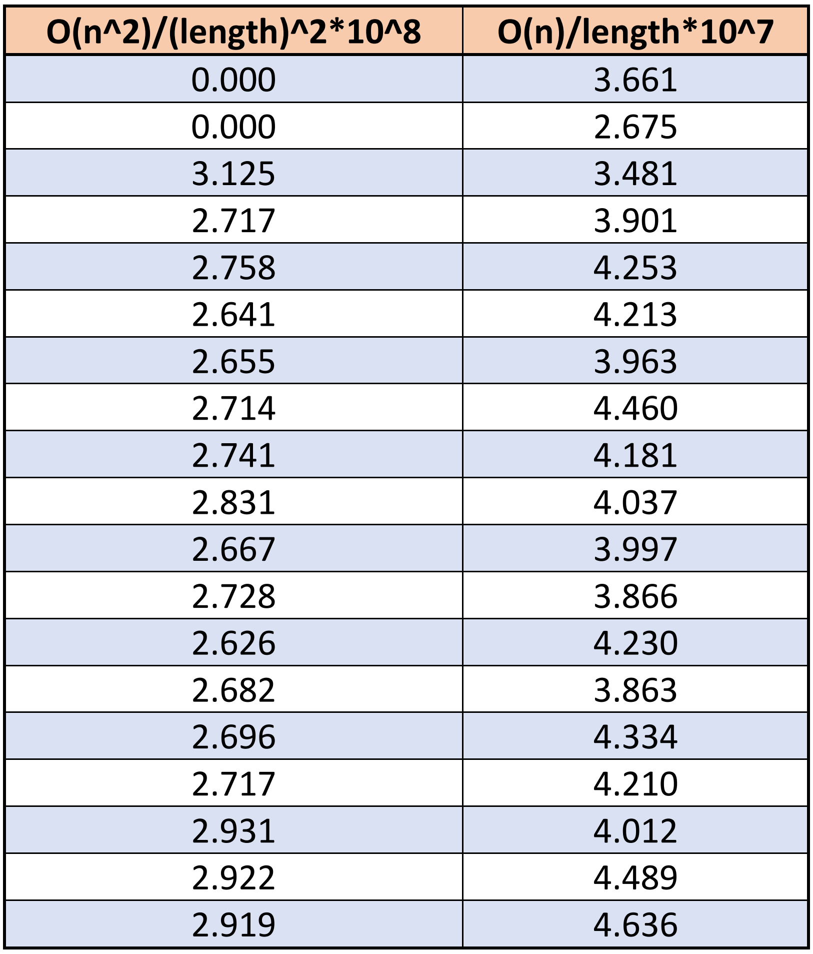
Note: Codes are also submitted to GitHub.

c) For both functions, the time it takes to call them is measured for 20 different lengths. For each length at least 10 measurements are made, and averages are calculated. With these measurements, the table given below was formed.



#	length of the price list	O(n^2) time	O(n) time
1	10	0	3.66E-06
2	100	0	2.68E-05
3	1000	0.03125	0.000348
4	2000	0.10868	0.000780
5	3000	0.24819	0.001276
6	4000	0.42262	0.001685
7	5000	0.66372	0.001982
8	6000	0.97712	0.002676
9	7000	1.34288	0.002926
10	8000	1.81213	0.003230
11	9000	2.16039	0.003597
12	10000	2.72839	0.003866
13	15000	5.90901	0.006345
14	20000	10.72942	0.007727
15	25000	16.85044	0.010834
16	30000	24.45018	0.012630
17	40000	46.89212	0.016047
18	50000	73.05197	0.022443
19	100000	291.91996	0.046365
20	student ID = 2120	0.12497	0.000822

Another table is created to check if these values match the given time complexity. In this table, O(n^2) values are divided by (length)^2, and similarly O(n) values are divided by length. If the specified time complexities are correct, the results inside of each column should be similar. The results are given below:



O(n^2)/(length)^2*10^8	O(n)/length*10^7
0.000	3.661
0.000	2.675
3.125	3.481
2.717	3.901
2.758	4.253
2.641	4.213
2.655	3.963
2.714	4.460
2.741	4.181
2.831	4.037
2.667	3.997
2.728	3.866
2.626	4.230
2.682	3.863
2.696	4.334
2.717	4.210
2.931	4.012
2.922	4.489
2.919	4.636

As seen from the table above, values inside each column are similar to each other. This provides strong evidence in favor of *“max_profit”* having time complexity of O(n’2) and *“max_profit_quick”* having time complexity of O(n).