ShanghaiTech University

CS240 Algorithm Design and Analysis

Fall 2023

Problem Set 1

Due: 23:59, Oct. 29, 2023

1. Submit your solutions to Gradescope (www.gradescope.com).
2. In "Account Settings" of Gradescope, set your FULL NAME to your Chinese name and enter your STUDENT ID correctly.
3. If you want to submit a handwritten version, scan it clearly. Camscanner is recommended.
4. When submitting your homework, match each of your solution to the corresponding problem number.

Problem 1:

Given a sequence $S = a_1, a_2, \cdots, a_n$. For some t between 1 and n, the sequence satisfies $a_1 < a_2 < \cdots < a_t$ and $a_t > a_{t+1} > a_n$. You would like to find the maximum value of the S by reading as few elements of S as possible. Show your algorithm and analyze the time complexity of it. For example, suppose the sequence S is $1, 7, 8, 9, 4, 3, 2$, then the max value is 9.

Solution:

Problem 2:

Suppose there are n trees in the campus of ShanghaiTech and their heights are denoted as $h_1, h_2, \cdots, h_n$. Now we want to find the m $(m \leq n)$ trees that are closest to this height for a given arbitrary height h. Please give an efficient algorithm to achieve this. For example, suppose the input of height of trees, a target height and a target number of trees are $h_1 = 8, h_2 = 5, h_3 = 3, h_4 = 1$, $h = 4$ and $m = 2$ respectively, then the output should be h_2, h_3. Note that the height of trees in the output must be in the original order.

Solution:

Problem 3:

Asymptotic Order of Growth. Sort all the functions below in increasing order of asymptotic (Big-Oh) growth.

1. $8n$
2. lgn
3. $10lg(lgn)$
4. $n^{3.14}$
5. n^{n^2}
6. lgn^{10lgn}
7. n^{lgn}

Solution:

Problem 4:

Asymptotic Order of Growth. Analyze the running time of following algorithm, and express it using "Big-Oh" notation.

Algorithm 1

```
Input: integers $n > 0$
  $i = 0, j = 0$
  while $i < n$ do
    while $j < i^2$ do
      $j+ = 1$
    end while
    $i+ = 1$
    $j = 0$
  end while
  return $j$
```

Solution:

Problem 5:

Greedy Algorithm. You are given n sorted arrays. Your task is to merge them into a single sorted array. You can only merge two arrays at a time, and the cost of merging is the sum of the lengths of the two arrays. Your goal is to find the merge strategy with the minimum total cost.
Detailed Problem Description:
Suppose we have 3 sorted arrays:

$$[1, 3, 9], [2, 4, 6], [0, 5, 7, 8]$$

We first merge the first two arrays, the cost of merging is 6 (the sum of the lengths of the two arrays). Then, we merge the resulting array with the third array, the cost of merging is 9 (the sum of the lengths of the two arrays). So, the total cost of merging is 15.

Solution:

Problem 6:

Greedy Algorithm. Suppose you're a manager of a candy store. The store has different candies for sale every day. Each day, you can choose to buy a type of candy and then sell it on a future day. Your goal is to maximize your profit by buying low and selling high.
Given an array where the i-th element is the price of a candy on day i, design an algorithm to find the maximum profit.
Detailed Description:
For instance, consider the following input:
Candy prices: $[7, 1, 5, 3, 6, 4]$
You should buy candy on day 2 (price $= 1$) and sell it on day 3 (price $= 5$), buy again on day 4 (price $= 3$) and sell it on day 5 (price $= 6$). The total profit is $5 - 1 + 6 - 3 = 7$.

Solution: