

We are recovering from significant hosting issues. Much of the site is functional, but currently email delivery is not. Please bear with us as we validate site functionality.



Bookmarks

- ▶ [Welcome to the edX Platform](#)
- ▶ [Entrance Survey](#)
- ▶ [Download Python and Get Motivated!](#)
- ▶ [Week 1: Python Basics](#)
- ▶ [Week 2: Simple Programs](#)
- ▶ [Week 3: Structured Types](#)
- ▶ [Week 4: Good Programming Practices](#)
- ▶ [Midterm Exam](#)
- ▶ [Week 5: Object Oriented Programming](#)

Week 6: Algorithmic Complexity > 12. Searching and Sorting Algorithms > Exercise 7

Exercise 7

Bookmark this page

Exercise 7

7 points possible (graded)

ESTIMATED TIME TO COMPLETE: 14 minutes

This problem will walk through some applications of complexity analysis. Suppose you're asked to implement an application. One of the things it has to do is to report whether or not an item, `x`, is in a list `L`. `L`'s contents do not change over time. Below are two possible ways to implement this functionality. Assume that `mergeSort` is implemented as per the lecture.

`L` is a list with `n` items.

- **Application A:**

Every time it's asked to, it performs a linear search through list `L` to find whether it contains `x`.

- **Application B:**

Sort list `L` once before doing anything else (using `mergeSort`). Whenever it's asked to find `x` in `L`, it performs a binary search on `L`.

1. If the application is asked to find `x` in `L` exactly one time, what is the worst case time complexity for Application A?


☐ $O(1)$ ☐ $O(\log n)$ 

▼ **Week 6:**
Algorithmic
Complexity


11. Computational
Complexity

[Finger Exercises](#) 

12. Searching and
Sorting
Algorithms

[Finger Exercises](#) 

Problem Set 6

[Problem Set due Mar](#)
[9, 2017 15:30 PST](#) 

► **Week 7:**
Plotting

► **Exit Survey**

► **Sandbox**

☐ $O(n)$

☐ $O(n \log n)$

☐ $O(n^2)$

2. If the application is asked to find x in L exactly one time, what is the worst case time complexity for Application B?

☐ $O(1)$

☐ $O(\log n)$

☐ $O(n)$

☐ $O(n \log n)$

☐ $O(n^2)$

3. If the application is asked to find x in L k times, what is the worst case time complexity for Application A?

☐ $O(1)$

☐ $O(k + \log n)$

☐ $O(k + n)$

☐ $O(kn)$

☐ $O(n + k \log n)$



4. If the application is asked to find x in L k times, what is the worst case time complexity for Application B?

☐ $O(kn)$

☐ $O(n \log n)$

☐ $O(n + k \log n)$

☐ $O(n \log n + k \log n)$

☐ $O(kn \log n + \log n)$

5. What value(s) of k would make Application A be faster (i.e., asymptotically grow slower than) Application B?

☐ $k = 1$

☐ $k = n$

☐ $k = \log n$

☐ $k = n^2$

☒ $k = 2^n$

6. What value(s) of k would make Application A grow at the same rate as Application B?

☐ $k = 1$

☐ $k = n$ ☐ $k = \log n$ ☐ $k = n^2$ ☐ $k = 2^n$

7. Which application should you choose if you know that there are going to be n^3 requests to find x in L ?

☐ Application A☐ Application B

Exercise 7

Topic: Lecture 12 / Exercise 7

[Show Discussion](#)

© All Rights Reserved



© 2012-2017 edX Inc. All rights reserved except where noted. EdX, Open edX and the edX and Open EdX logos are registered trademarks or trademarks of edX Inc.



