

# CE Linear Binary

10 Possible Points

9/5/2022

Attempt 1



**IN PROGRESS**

Next Up: Submit Assignment



Add Comment

Unlimited Attempts Allowed

▼ Details

Fundamentals

CE: Linear Binary



## Learning Objectives

- Identify similarities and differences between linear search and binary search.
- Implement a search algorithm
- Review JUnit testing



## Overview

In this CE, you will create a class called Search with two static methods: linear and binary. You will write JUnit tests for both methods and implement linear search.

< [Previous](#)

(<https://slcc.instructure.com/courses/817632/modules/items/18752919>)

Submit Assignment

[Next](#) >

(<https://slcc.instructure.com/courses/817632/modules/items/18752923>)

## Linear Search vs Binary Search with arrays

- Create a package called **ceLinearBinary** and add the class **Search**. No need to include a main method.
- Add two public static methods: **linear** and **binary**.  
Both methods have a return type `int` and two parameters:
  - `numbers` of type `int[]`
  - `key` of type `int`Add doc comments to both methods.

Here is an [example](#)

We start writing the JUnit tests before implementing the methods. However, to do so, the class `Search` needs to compile.

Return 0 (the default value for `int`) and add a comment `//TODO` to resolve syntax issues and to indicate that the implementation doesn't provide the expected functionality yet.

- Add a corresponding JUnit test file called **SearchTest** to the project.  
It should be in a separate source folder called **test** and it should include test method stubs for the two methods `linear` and `binary`.  
Run the JUnit tests. At this point, both test methods are expected to fail. Here is something to [compare](#)

- Identify at least five situations we should test for (e.g. search for the first element in the array, search for an element that is not included in the array, etc.)  
The goal is to cover as many situations as we can and to maximize the likelihood of uncovering bugs in the program.

Write descriptive test method names that start with the following pattern:

`{methodName}_{what we test}`

If you test the method `linear` by searching for the first element, the name could be `linear_searchFirstElement` or `linear_firstElement`.

Once you identified five situations and the corresponding method names, [compare](#)

[< Previous](#)

(<https://slcc.instructure.com/courses/817632/modules/items/18752919>)

Submit Assignment

[Next >](#)

(<https://slcc.instructure.com/courses/817632/modules/items/18752923>)

been properly implemented yet and returns a constant 0.

- Go back to class Search and Implement the method linear.  
Run the JUnit tests again. At this point, the tests for the method linear should pass.
- Let's turn to the method binary.  
Start by identifying and implementing at least five test methods to test the method binary.  
Keep in mind that binary search requires a sorted array when choosing your test data.
- Go back to class Search and Implement the method binary.  
At this point, all tests should pass.
- Refactor  
If you used private final fields to declare the test arrays, that's great. If you haven't done so, refactor to the code to simplify the test methods.



## Submission

Create a screen recording following the [guidelines for lab recordings](https://slcc.instructure.com/courses/817632/pages/guidelines-for-lab-recordings) (<https://slcc.instructure.com/courses/817632/pages/guidelines-for-ce-recordings>).

Show both the code from class Search and SearchApp. When you run the tests, show the test results including how many tests were executed.

The video should be **30-60 seconds** long.

Post the video.

[< Previous](https://slcc.instructure.com/courses/817632/modules/items/18752919)

<https://slcc.instructure.com/courses/817632/modules/items/18752919>

Submit Assignment

[Next >](https://slcc.instructure.com/courses/817632/modules/items/18752923)

<https://slcc.instructure.com/courses/817632/modules/items/18752923>

  (<https://slcc.instructure.com/courses/817632/modules/items/18752915>)

 (<https://slcc.instructure.com/courses/817632/modules/items/18752916>)


 (<https://slcc.instructure.com/courses/817632/modules/items/18752917>)


 ([https://slcc.instructure.com/api/v1/courses/817632/module\\_item\\_redirect/18752919](https://slcc.instructure.com/api/v1/courses/817632/module_item_redirect/18752919))


  (<https://slcc.instructure.com/courses/817632/modules/items/18752921>)


 (<https://slcc.instructure.com/courses/817632/modules/items/18752923>)

### Choose a submission type

  
Text

  
Upload

  
Office 365

  
More

[< Previous](#)

<https://slcc.instructure.com/courses/817632/modules/items/18752919>

Submit Assignment

[Next >](#)

<https://slcc.instructure.com/courses/817632/modules/items/18752923>