

Lab 5: Algorithms

Instructions:

This worksheet serves as a guide and set of instructions to complete the lab.

- You must use the [starter file, found here](#), to get credit for the lab.
- Additionally, [here is the workbook](#) that you can read through for further context and additional (non-required) material.
- All material was sourced from the CS10 version of The Beauty and Joy of Computing course.

Submitting:

You will need to fill in the blocks under “algorithms-framework-no-solutions” and submit this to Gradescope.

- To receive full credit, you will need to complete the required blocks, and the required blocks must pass all tests from the autograder in Gradescope.
- For instructions on how to submit to labs to Gradescope, [please see this document](#).

Please note, you must use the [starter file](#), and you must NOT edit the name of any of the required blocks. Failing to do either for these will result in the autograder failing.

Objectives:

We have learned that in computer science, **algorithms** are the conceptual solutions to a problem whereas the **implementation** of an algorithm is the actual code. In this lab, you will apply what you have learned so far to write your own implementation of an algorithm. By the end of lab you will:

- Differentiate and identify parts of an algorithm
- Gain a basic understanding of why certain algorithms perform better than others.
- Use existing algorithms as building blocks for constructing new algorithms
- Practice implementing algorithms that were discussed verbally.
- Understand the components of a binary search algorithm

Required Blocks:

- Block 1: find the number (num) in unsorted list (list)
- Block 2: find the number (num) in sorted list (list)

Important Topics mentioned in the Workbook:

For better understanding of the lab please go through these workbook pages! Topics that are important but not required for this lab will be indicated with an asterisk**. These topics are best reviewed in order and as you complete the lab.

- [Everyone's Got Algorithms](#)
- [Algorithms in Snap!](#)
- [Improving Our Number Finder](#) **

Block 1: find the number (num) in unsorted list (list)

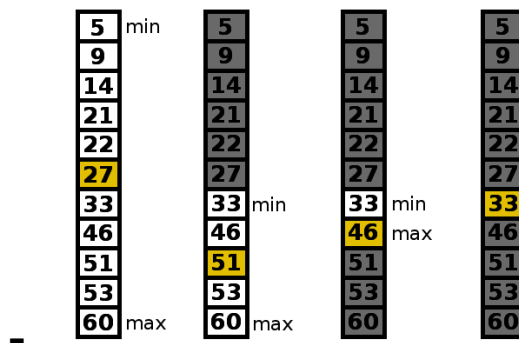
- Objective:
 - Create a reporter that reports the index of a number if found within the list. If the number doesn't exist in the list, reports -1
 - **Can use iteration**
- Inputs:
 - Num = any number
 - List = a list of any length, with any number
 - We can assume that input list has unique numbers
- Output:
 - a number
 - Reports the position of the num if found or -1
- Examples:

- find the number 6 in unsorted list [10, 11, 3, 7, 6]
- find the number -1 in unsorted list [1, 2, 3, 7, 6, -2, -1]
- find the number -1 in unsorted list [-1, 0, 1]


Block 2: Find the number (num) in sorted list (list)

- Objective:
 - Create a reporter that finds the index of the number within a sorted list. If the number doesn't exist, report -1
 - ** Please implement the binary search algorithm found in the [workbook](#)
 - Continuously "halving" the list, until index is found

Finding 33.



- Notes:
 - Not allowed: for loops, “index of”
 - Allowed: “repeat until”, variables!, “item of”
- Inputs:
 - Num = any number
 - List = a list of any length, with any number
 - We can assume that input list has unique numbers
- Output:
 - a number
 - Reports the position of the num if found or -1
- Examples:

- 
- 
- 

Remember to submit on Gradescope!