Linear Search

The reason I'm doing linear search is that it is the first searching method you learned, as it teaches you most of the fundamentals used for other searches and sorting methods.
Such as learning to control and understand which index you're looking at and running loops until it's finished. Linear search also teaches you early exit, as it stops when you find the value. All of these skills help you learn other and more complex search and sorting algorithms.

Problem Decomposition

- Get a list of values
- Get a target value to search for
- loop to move through the list one index at a time
- Compare each value with the target
- Keep track of each step for the animation
- Deciding when to stop (value found / not in list)

Pattern Recognition

Each step is the same: compare the values at the current index, then move down the list

Abstraction

We don't care about how the list is stored in memory
We don't care about the speed of individual comparison, only focusing on the index number and the target

Algorithmic Thinking

Inputs: a list and a target value
Outputs: the animation and steps to find the value or if the value is not found

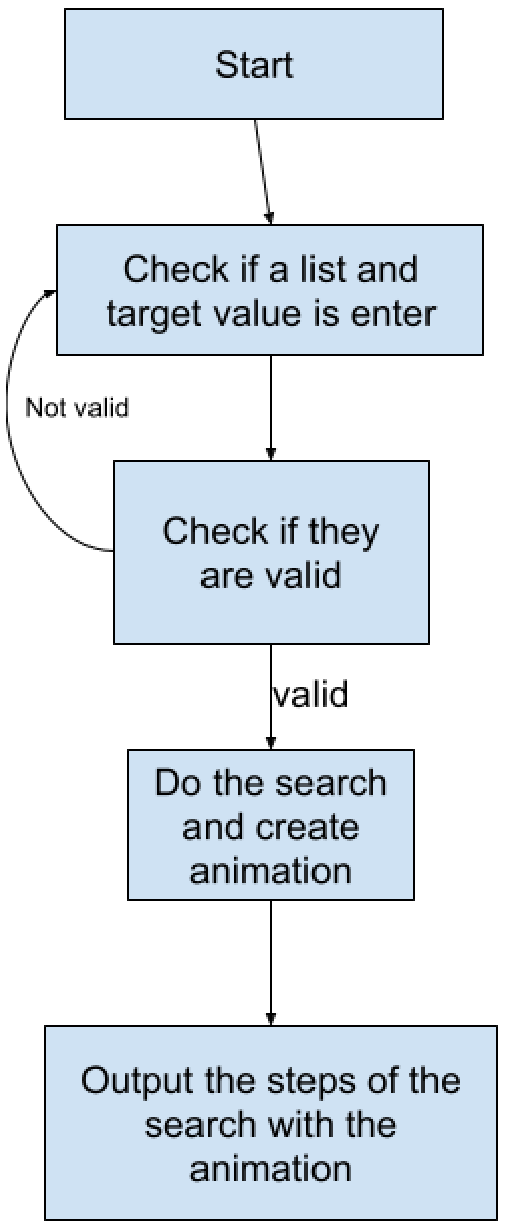
Steps to Run

1. Start at index 0
2. Compare the current value with the index
3. Stop if they match and return the index
4. If not, move the index by 1
5. Repeat until the end of the list
6. If no match, return not found

Algorithm Design

In this, the user interacts with a GUI built with Gradio where they enter a list of numbers and a target number into textboxes. The program performs a linear search by checking each element of the list one by one. During the process, every comparison step is stored and used to generate a visual animation. The animation logic highlights the current index, moving an arrow along the array and pausing for a second, then continuing, stopping on the target or end of the list There are arrows to go along the steps step by step with text to highlight what step the search is on . Finally, the program outputs both the animated visualization and a text-based explanation of each step, concluding with either the index where the target was found or a message indicating that the target does not exist in the list.

Flowchart


Start
Check if a list and target value is enter
Not valid
Check if they are valid
valid
Do the search and create animation
Output the steps of the search with the animation

Hugging Face app link

https://huggingface.co/spaces/Andrewtucker/CISC-121-Project

Ai disclaimer

Chat gpt was used to help build the gradio GUI and test and pervent edge cases , everything on the readme.md was 100% made by me with no ai besides basic grammar fixes . the design of the code was made by me and built with ai help all steps were monster and all changes were made with human approval.

Test and Verify

- regular lists of any size work as long as they hav “,” inbeteen them
- If “,” is not inbeteen or inproerly done it wont break but an error message telling you what to fix will apare
- If the target is not in the list it will work
- Empty list will work and not break it
- Invalid character wont break it (ex:letters , char)