

Activity No. <6.1>	
<Linear and Binary Search>	
Course Code: CPE010	Program: Computer Engineering
Course Title: Data Structures and Algorithms	Date Performed: 9/11/2025
Section: CPE21S4	Date Submitted: 9/11/2025
Name(s): Lester Arvid P. Anastacio	Instructor: Engr. Jimlord Quejado
6. Output	
7. Supplementary Activity	
<p>Answer the following questions:</p> <ol style="list-style-type: none"> What is a search tree in data structures? <ul style="list-style-type: none"> A search tree in data structures is a hierarchical data structure where each node or element stores a value. In which is arranged in an organized format to enable efficient searching, these are basically used for database indexing, symbol tables, range queries and are foundational for advanced structures like AVL trees, which means are used in problems where we need to maintain sorted stream of data. What are the Different types of search algorithm in data structures? Differentiate each type of search. <ul style="list-style-type: none"> First is Linear Search which is used when the data is unsorted or small, this scans each element one by one until it reaches or finds the needed target, second is the Binary Search this is usually used when the data is sorted, this divides the sorted data in half repeatedly until it finds the target. What operations / implementations can be performed using binary and linear search operations? <ul style="list-style-type: none"> In Linear search, the operations that can be performed when using this is quite simple as it is for searching for an element within arrays, lists and linked lists, it is also used to find duplicates, checking for existence of a value whether it exist or not in the array, or used to brute force algorithms while in binary search, the operations used is for efficient element lookup in sorted arrays or lists, it is also used to find the positions of an element and usually used in advanced algorithms such as search trees and heaps. What are the advantages in using binary search tree as data structure? <ul style="list-style-type: none"> The advantages in using binary search tree are that it is efficient in searching within sorted arrays and algorithms, you can also used this to insert or delete elements dynamically without having to re-sorting the entire data, you can use this too for sorted data maintenance as you can easily insert and delete elements, making it easy to maintain the sorted data. Give an example program using binary search and Linear search. <p>BINARY SEARCH EXAMPLE PROGRAM:</p> <p>CODE:</p> <pre>#include <iostream> int binarySearch(int arr[], int size, int target) { int low = 0; int high = size - 1; while (low <= high) { int mid = low + (high - low) / 2; if (arr[mid] == target) return mid; // Found } }</pre>	

```

    else if (arr[mid] < target)
        low = mid + 1; // Search right half
    else
        high = mid - 1; // Search left half
}

return -1; // Not found
}

int main() {
    int arr[] = {3, 6, 9, 12, 15, 18, 21};
    int size = sizeof(arr) / sizeof(arr[0]);
    int target = 15;

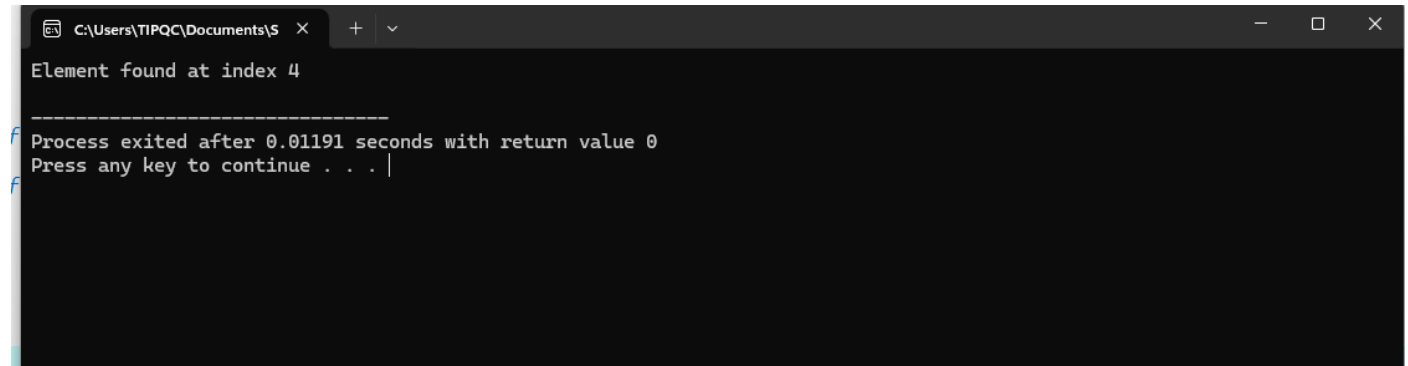
    int result = binarySearch(arr, size, target);

    if (result != -1)
        std::cout << "Element found at index " << result << std::endl;
    else
        std::cout << "Element not found in the array" << std::endl;

    return 0;
}

```

OUTPUT:



```

C:\Users\TIPQC\Documents\S >
Element found at index 4
-----
Process exited after 0.01191 seconds with return value 0
Press any key to continue . . .

```

LINEAR SEARCH EXAMPLE PROGRAM:

CODE:

```

#include <iostream>

void linearSearchArr(int arr[],int dataFind,int arrSize);

int main(){
    int arr[5]= {1,2,3,4,5};
    linearSearchArr(arr, 4, 5);

    return 0;
}

void linearSearchArr(int arr[],int dataFind,int arrSize){
    for (int i = 0; i < arrSize; i++){
        if (arr[i] == dataFind){

```

```

        std::cout << "Data found at index: " << i << std::endl;
        return;
    }
}
std::cout << "Data: (" << dataFind << ") Wasn't found in the array." << std::endl;
}

```

OUTPUT:

```

C:\Users\TIPQC\Documents\S
Data: (4) Wasn't found in the array.

-----
Process exited after 0.02031 seconds with return value 0
Press any key to continue . . . |

```

8. Conclusion

During this Activity, I've learned how the Linear and Binary Search works and how we can implement it, I've learned to understand how useful it is to find certain data, within an array or list or in a sorted one as data within it can extend in such long ways.

9. Assessment Rubric