Seatwork 6.1	
Linear and Binary Search	
Course Code: CPE010	**Program:** Computer Engineering
Course Title: Data Structures and Algorithms	**Date Performed:** 9/11/25
Section: CPE21S4	**Date Submitted:** 9/11/25
Name(s): Avila, Vince Gabriel V.	**Instructor:** Engr. Jimlord Quejado

1. Output

1. **What is a search tree in data structures?**
   A search tree is like a family tree for data, where each piece is placed so you can find it quickly. The most common type, a binary search tree, keeps smaller values on the left and bigger ones on the right.

2. **What are the different types of search algorithms in data structures?**
   Linear search checks each item one by one, while binary search cuts the list in half each time to find things faster. Other types, like DFS and BFS, are used for more complex structures like graphs.

3. **What operations can you do with binary and linear search?**
   Linear search can find, add, or remove items anywhere in a list, but it's slow. Binary search is super fast for finding things in a sorted list, but it can't add or remove items directly.

4. **What are the advantages of using a binary search tree?**
   Binary search trees make searching, adding, and removing data much faster than a normal list. They also keep your data sorted automatically.

5. **Give an example program using binary search and Linear search.**

Linear Search:

```cpp
#include <iostream>
#include <vector>
using namespace std;

int search(vector<int> &arr, int target){
    int n = arr.size();

    // Iterate linearly through the array
    for (int i = 0; i < n; i++)
        if (arr[i] == target)
            return i;
    return -1;
}

int main() {
    vector<int> arr = {2, 3, 4, 7, 1, 5};
    int target = 7;
    int index = search(arr, target);
    cout<<index<<endl;
    return 0;
}
```

```
C:\Users\TIPQC\Desktop\Unti
3

--------------------------------
Process exited after 0.01183 seconds with return value 0
Press any key to continue . . .
```

Binary Search:

```cpp
#include <iostream>
#include <vector>
using namespace std;

int binarySearch(vector<int> &arr, int target, int low, int high) {

    // Repeat until the pointers low and
    // high meet each other
    while (low <= high) {
        int mid = low + (high - low) / 2;

        if (arr[mid] == target)
            return mid;

        if (arr[mid] < target)
            low = mid + 1;

        else
            high = mid - 1;
    }

    return -1;
}

int main(){
    vector<int> arr = {2, 3, 4, 7, 9, 10};
    int n = arr.size();
    int target = 7;
    int low = 0;
    int high = n;
    int index = binarySearch(arr, target, low, high);
    cout<<index<<endl;
    return  0;
}
```

```
C:\Users\TIPQC\Desktop\Unti
3

--------------------------------
Process exited after 0.01183 seconds with return value 0
Press any key to continue . . .
```

APA Citation:

GeeksforGeeks. (2025, September 4). Binary search tree. GeeksforGeeks. https://www.geeksforgeeks.org/dsa/binary-search-tree-data-structure/

3. Conclusion

After exploring this topic. Search trees and algorithms are just ways of finding stuff and putting stuff in order. Some can be really simple but are slow. Others, like binary search trees, are smarter and find stuff really fast. The best one to use will depend on what you want. If you want things to be really fast then binary search trees will help you.

4. Assessment Rubric