

frankSJSU DataStructure

Frank's Home Page

CMPE126 home

Greensheet

Frank's Notes

[operator overloading](#)
[storing objects](#)
[pointer & deep copy](#)
[array of objects](#)
[linked list](#)
[variable size objects](#)
[create a linked node](#)
[create a linked list](#)
[linked list insertion](#)
[find middle](#)
[hybrid list](#)
[linked list quiz](#)
[recursion](#)
[stack](#)
[stack with array](#)
[math expression](#)
[queue](#)
[simulation](#)
[frankSimulation s16](#)
[priority queue & heap](#)
[search by hashing](#)

Frank's Slides

Frank's Code

Programming Exam

[PE #3 F16](#)
[PE #1 guide F15](#)

Midterm Exams

[midterm 2 F19](#)
[midterm 1 S18](#)
[midterm 2 F17](#)
[midterm 1 F17](#)
[midterm 2 S17](#)
[midterm1 S17](#)
[midterm2 F16](#)
[midterm1 F16](#)
[Labs and Homeworks >](#)

Lab 8 search

Objective

1. practice various searching algorithms
2. strengthen array and linked list processing

Overview

Write a menu driven search program implementing the following:

```
do {
    cout<<"\nChoose your search type:";
    cout<<"\n1. Arrays: Sequential Search without recursion";
    cout<<"\n2. Arrays: Sequential Search with recursion";
    cout<<"\n3. Ordered Arrays: Binary Search without recursion";
    cout<<"\n4. Ordered Arrays: Binary Search with recursion";
    cout<<"\n5. Linked List: Search without recursion";
    cout<<"\n6. Linked List: Search with recursion";
    cout<<"\nEnter 0 to exit.\nYour choice: ";
    ...
    cout<<"\nSpecify the element to be searched for: ";
    ...
} while(i!=0);
```

Print the list before selecting the element to search for.

The search should return the position of the element (starting at zero). If the element is not in the list, the function should return a negative one (-1).

You may find it useful to create helper functions to create filled linked lists, filled arrays, and sorted arrays.

Sample output:

```
Choose your search type:
1. Arrays: Sequential Search without recursion
2. Arrays: Sequential Search with recursion
3. Ordered Arrays: Binary Search without recursion
4. Ordered Arrays: Binary Search with recursion
5. Linked List: Search without recursion
6. Linked List: Search with recursion
Enter 0 to exit.
Your choice: 1

Specify the number of elements to be searched: 10

Elements added to the array: 41 67 34 0 69 24 78 58 62 64

Specify the element to be searched for: 62
Element index: 8
```

Discussions

[midterm S16](#)**Final Exams**[Final S17](#)[Final S16](#)[Final F15](#)[Final S15](#)**Labs and Homeworks**[Misc Lab FYI](#)[Lab 0 C++](#)[Lab 1 classes](#)[Lab 2 object array](#)[Lab 3 Linked List](#)[Lab 4 Doubly Linked List](#)[Lab 5 Recursion](#)[Lab 6 Stack](#)[Lab 6+ math expression](#)[Lab 7 Simulation](#)[Lab 7a Palindrome](#)**Lab 8 search**[Lab 9 hashing](#)[Lab 10 sort](#)

- if you're not comfortable with the exercise, you can use integer as your list elements.
- I would prefer that you use stock objects as your list elements if you're up to it.

Comments

You do not have permission to add comments.