

# Linear and binary search

Michael Nowak

Texas A&M University

# Overview

MyArray

Searching for some value in an array

Linear Search

Binary Search

# Overview

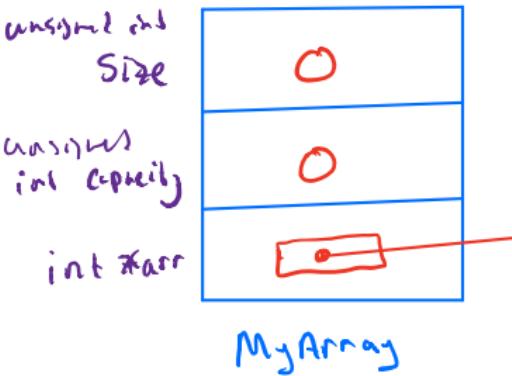
## MyArray

Searching for some value in an array

Linear Search

Binary Search

## MyArray



```
1 #ifndef MYARRAY_H
2 #define MYARRAY_H
3 struct MyArray {
4     int *arr = nullptr;
5     unsigned int capacity = 0; // no elements can store
6     unsigned int size = 0; // no elements currently held
7 };
8 #endif
```

# Overview

MyArray

Searching for some value in an array

Linear Search

Binary Search

## Setting-up

```
1 #include <iostream>
2 #include "MyArray.h"
3 #include "linearSearch.h"
4 #include "binarySearch.h"
5
6 using namespace std;
7
8 int main()
9 {
10     MyArray ma1;
11     ma1.arr = new int[7];
12     ma1.capacity = 7;
13     ma1.size = 7;
14     for (unsigned int i = 0; i < ma1.size; ++i)
15         ma1.arr[i] = i * 10;
16
17     unsigned int idxofvalue = linearSearch(ma1, 20);
18
19     unsigned int idxofvalue2 = binarySearch(ma1, 20);
20
21     delete [] ma1.arr;
22
23     return 0;
24 }
```

IDENTIFIER

STACK

FREE  
STORE

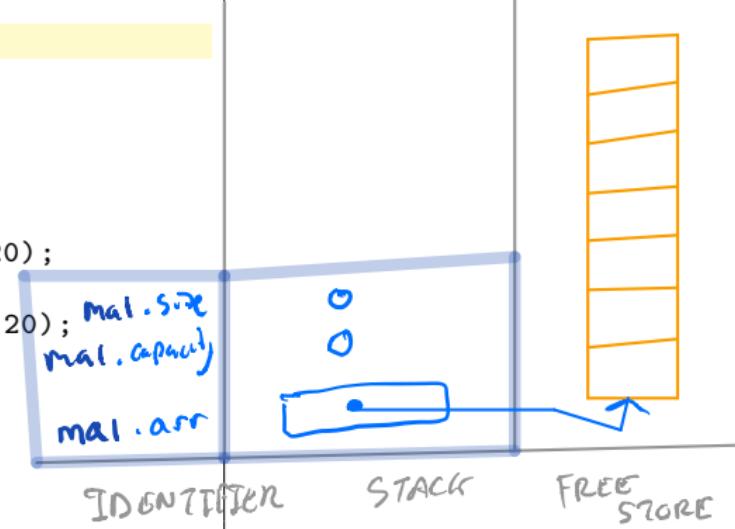
## Setting-up

```
1 #include <iostream>
2 #include "MyArray.h"
3 #include "linearSearch.h"
4 #include "binarySearch.h"
5
6 using namespace std;
7
8 int main()
9 {
10     MyArray ma1;
11     ma1.arr = new int[7];
12     ma1.capacity = 7;
13     ma1.size = 7;
14     for (unsigned int i = 0; i < ma1.size; ++i)
15         ma1.arr[i] = i * 10;
16
17     unsigned int idxofvalue = linearSearch(ma1, 20);
18
19     unsigned int idxofvalue2 = binarySearch(ma1, 20); ma1.size
20
21     delete [] ma1.arr;
22
23     return 0;
24 }
```



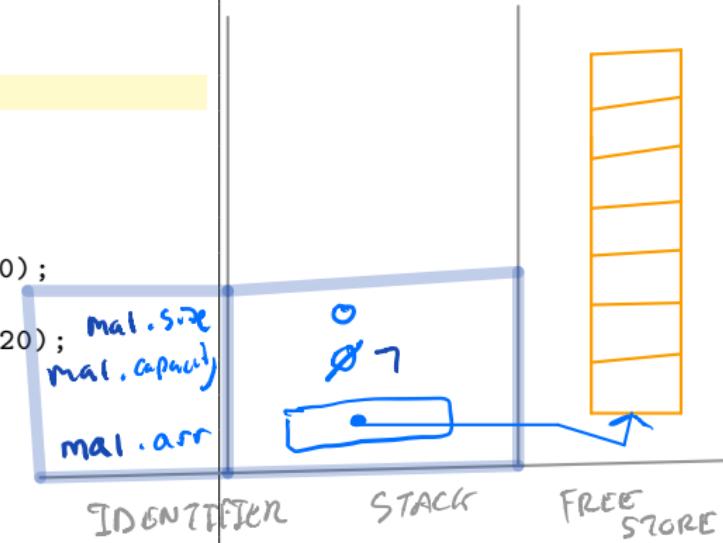
## Setting-up

```
1 #include <iostream>
2 #include "MyArray.h"
3 #include "linearSearch.h"
4 #include "binarySearch.h"
5
6 using namespace std;
7
8 int main()
9 {
10     MyArray ma1;
11     ma1.arr = new int[7];
12     ma1.capacity = 7;
13     ma1.size = 7;
14     for (unsigned int i = 0; i < ma1.size; ++i)
15         ma1.arr[i] = i * 10;
16
17     unsigned int idxofvalue = linearSearch(ma1, 20);
18
19     unsigned int idxofvalue2 = binarySearch(ma1, 20)
20
21     delete [] ma1.arr;
22
23     return 0;
24 }
```



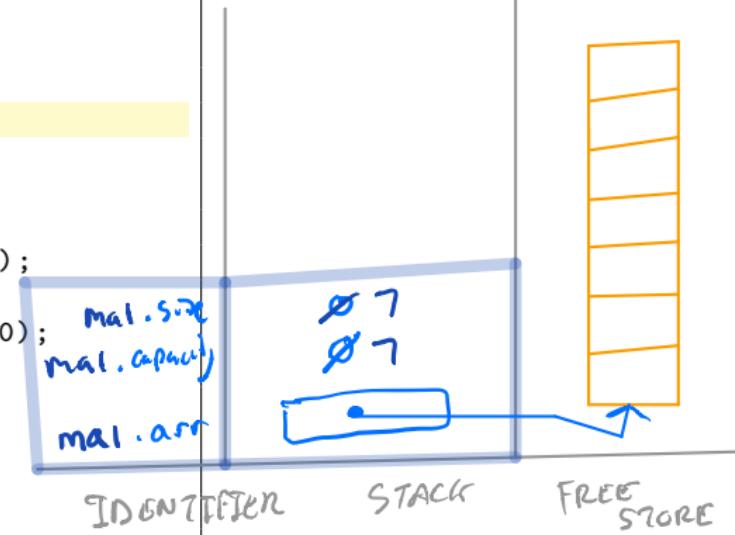
# Setting-up

```
1 #include <iostream>
2 #include "MyArray.h"
3 #include "linearSearch.h"
4 #include "binarySearch.h"
5
6 using namespace std;
7
8 int main()
9 {
10     MyArray ma1;
11     ma1.arr = new int[7];
12     ma1.capacity = 7;
13     ma1.size = 7;
14     for (unsigned int i = 0; i < ma1.size; ++i)
15         ma1.arr[i] = i * 10;
16
17     unsigned int idxofvalue = linearSearch(ma1,
18
19     unsigned int idxofvalue2 = binarySearch(ma1,
20
21     delete [] ma1.arr;
22
23     return 0;
24 }
```



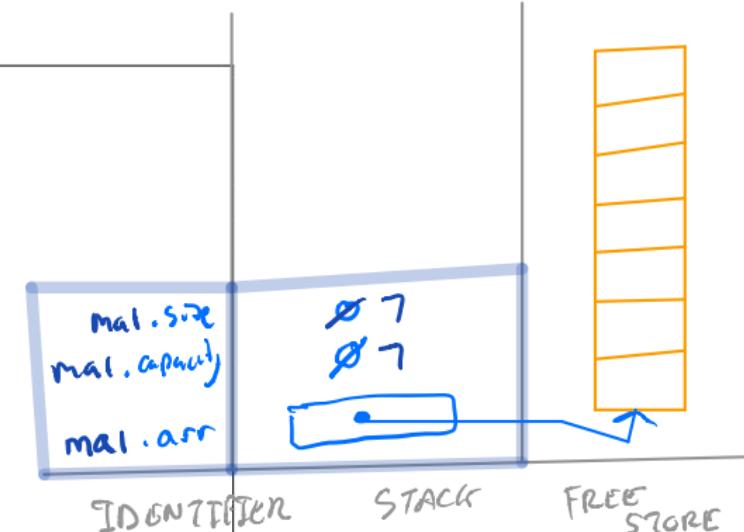
## Setting-up

```
1 #include <iostream>
2 #include "MyArray.h"
3 #include "linearSearch.h"
4 #include "binarySearch.h"
5
6 using namespace std;
7
8 int main()
9 {
10     MyArray ma1;
11     ma1.arr = new int[7];
12     ma1.capacity = 7;
13     ma1.size = 7; // Line 13
14     for (unsigned int i = 0; i < ma1.size; ++i)
15         ma1.arr[i] = i * 10;
16
17     unsigned int idxofvalue = linearSearch(ma1, 20);
18
19     unsigned int idxofvalue2 = binarySearch(ma1, 20); // Line 19
20
21     delete [] ma1.arr;
22
23     return 0;
24 }
```



## Setting-up

```
1 #include <iostream>
2 #include "MyArray.h"
3 #include "linearSearch.h"
4 #include "binarySearch.h"
5
6 using namespace std;
7
8 int main()
9 {
10     MyArray ma1;
11     ma1.arr = new int[7];
12     ma1.capacity = 7;
13     ma1.size = 7;
14     for (unsigned int i = 0; i < ma1.size; ++i)
15         ma1.arr[i] = i * 10;
16
17     unsigned int idxofvalue = linearSearch(ma1, 20);
18
19     unsigned int idxofvalue2 = binarySearch(ma1, 20);
20
21     delete [] ma1.arr;
22
23     return 0;
24 }
```



## Setting-up

```
1 #include <iostream>
2 #include "MyArray.h"
3 #include "linearSearch.h"
4 #include "binarySearch.h"
5
6 using namespace std;
7
8 int main()
9 {
10     MyArray ma1;
11     ma1.arr = new int[7];
12     ma1.capacity = 7;
13     ma1.size = 7;
14     for (unsigned int i = 0; i < ma1.size; ++i)
15         ma1.arr[i] = i * 10;
16
17     unsigned int idxofvalue = linearSearch(ma1, 20);
18
19     unsigned int idxofvalue2 = binarySearch(ma1, 20);
20
21     delete [] ma1.arr;
22
23     return 0;
24 }
```

mal.size  
mal.capacity  
ma1.arr

IDENTIFIER

∅ 7  
∅ 7

STACK

60  
50  
40  
30  
20  
10  
0

FREE STORE

# Overview

MyArray

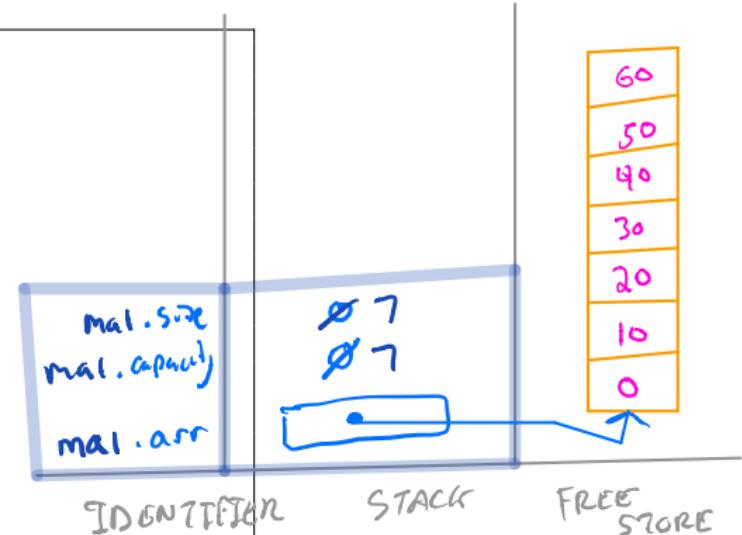
Searching for some value in an array

Linear Search

Binary Search

## Calling our linearSearch function

```
1 #include <iostream>
2 #include "MyArray.h"
3 #include "linearSearch.h"
4 #include "binarySearch.h"
5
6 using namespace std;
7
8 int main()
9 {
10     MyArray ma1;
11     ma1.arr = new int[7];
12     ma1.capacity = 7;
13     ma1.size = 7;
14     for (unsigned int i = 0; i < ma1.size; ++i)
15         ma1.arr[i] = i * 10;
16
17     unsigned int idxofvalue = linearSearch(ma1, 20);
18
19     unsigned int idxofvalue2 = binarySearch(ma1, 20);
20
21     delete [] ma1.arr;
22
23     return 0;
24 }
```

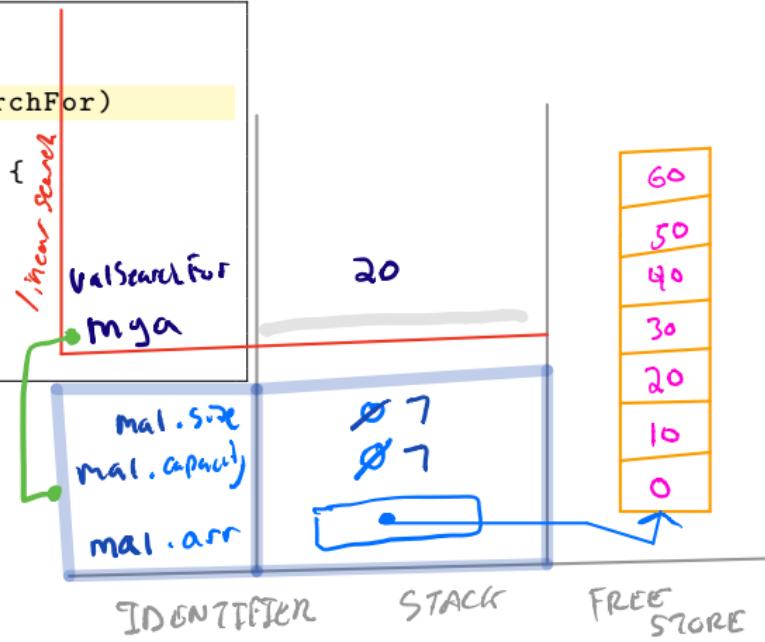


## linearSearch()

```
1 #include "linearSearch.h"
2
3 int linearSearch(MyArray const &mya, int valSearchFor)
4 {
5     for (unsigned int i = 0; i < mya.size; ++i) {
6         if (mya.arr[i] == valSearchFor)
7             return i;
8     }
9     return -1;
10 }
```

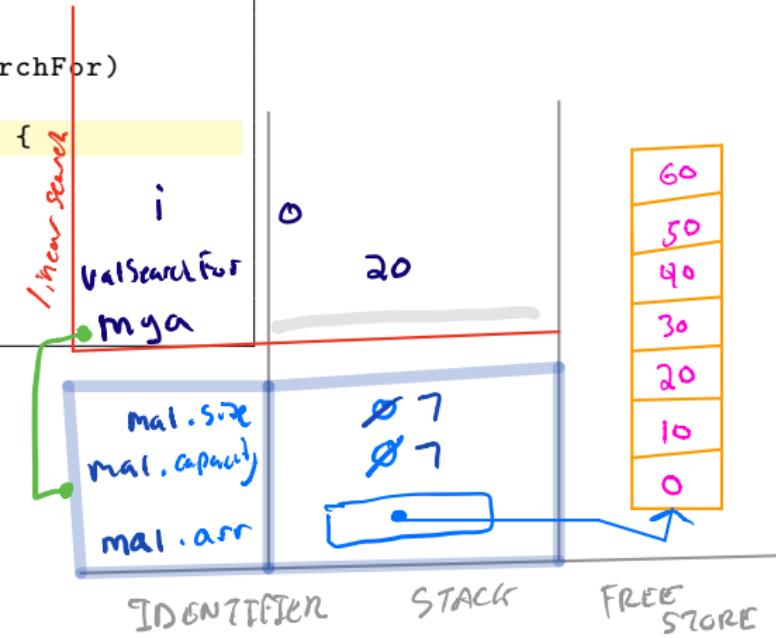
## linearSearch()

```
1 #include "linearSearch.h"
2
3 int linearSearch(MyArray const &mya, int valSearchFor)
4 {
5     for (unsigned int i = 0; i < mya.size; ++i) {
6         if (maya.arr[i] == valSearchFor)
7             return i;
8     }
9     return -1;
10 }
```



## linearSearch()

```
1 #include "linearSearch.h"
2
3 int linearSearch(MyArray const &mya, int valSearchFor)
4 {
5     for (unsigned int i = 0; i < mya.size; ++i) {
6         if (maya.arr[i] == valSearchFor)
7             return i;
8     }
9     return -1;
10 }
```



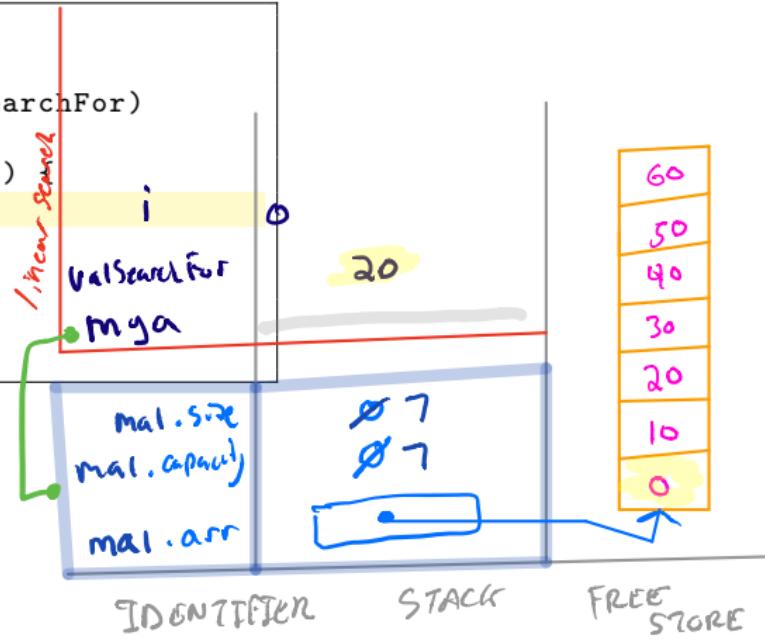
## linearSearch()

```
1 #include "linearSearch.h"
2
3 int linearSearch(MyArray const &mya, int valSearchFor)
4 {
5     for (unsigned int i = 0; i < mya.size; ++i)
6         if (maya.arr[i] == valSearchFor)
7             return i;
8     }
9     return -1;
10 }
```

Linear Search

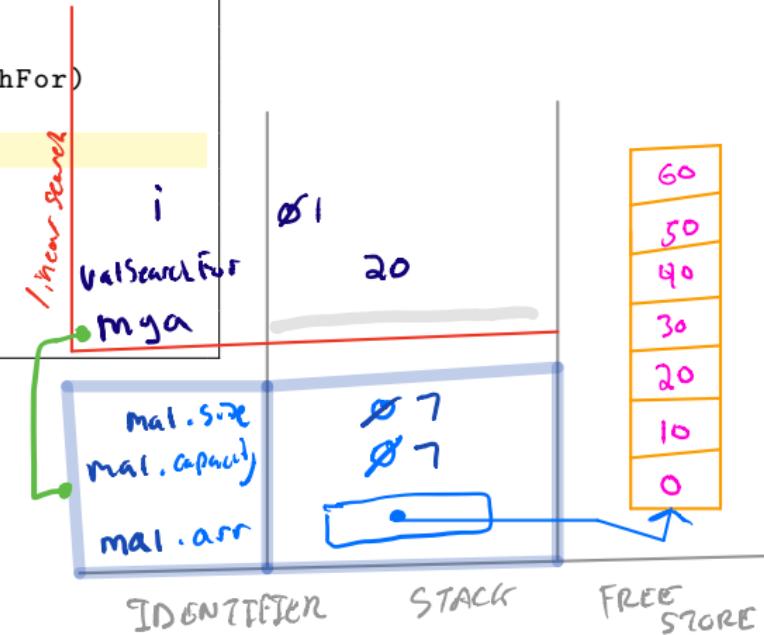
Value

My



## linearSearch()

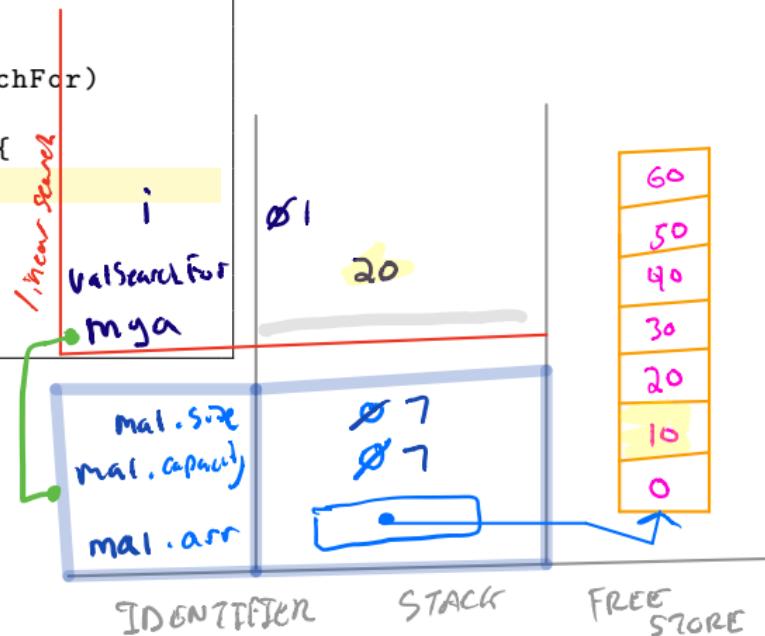
```
1 #include "linearSearch.h"
2
3 int linearSearch(MyArray const &mya, int valSearchFor)
4 {
5     for (unsigned int i = 0; i < mya.size; ++i) {
6         if (mya.arr[i] == valSearchFor)
7             return i;
8     }
9     return -1;
10 }
```



## linearSearch()

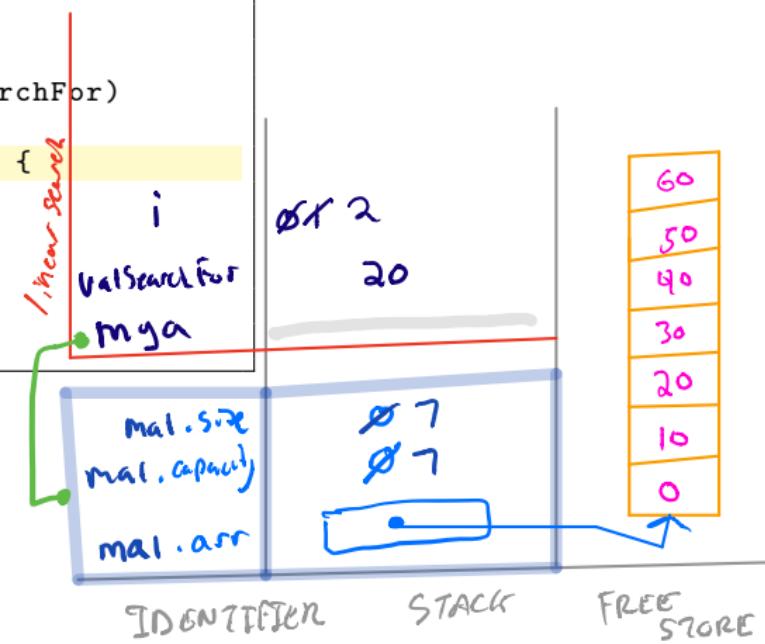
```
1 #include "linearSearch.h"
2
3 int linearSearch(MyArray const &mya, int valSearchFor)
4 {
5     for (unsigned int i = 0; i < mya.size; ++i) {
6         if (mya.arr[i] == valSearchFor)
7             return i;
8     }
9     return -1;
10 }
```

Linear Search



## linearSearch()

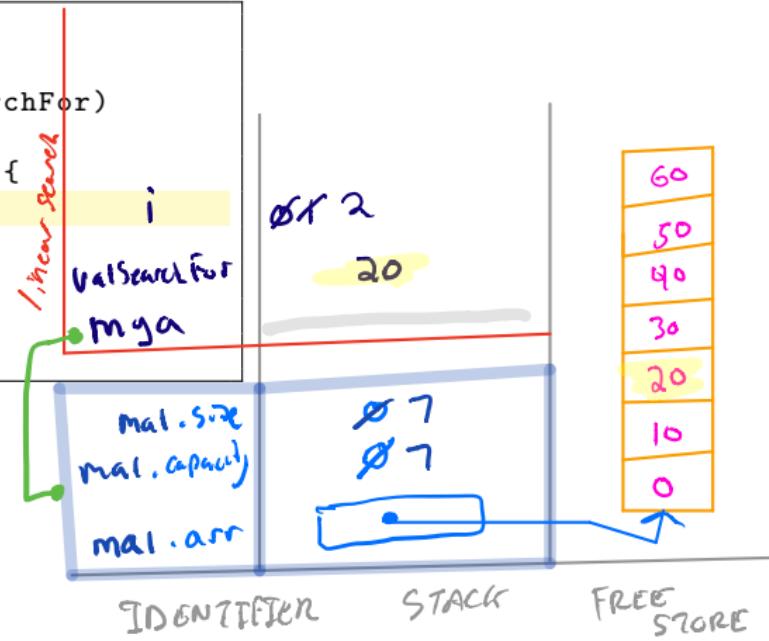
```
1 #include "linearSearch.h"
2
3 int linearSearch(MyArray const &mya, int valSearchFor)
4 {
5     for (unsigned int i = 0; i < mya.size; ++i) {
6         if (mya.arr[i] == valSearchFor)
7             return i;
8     }
9     return -1;
10 }
```



## linearSearch()

```
1 #include "linearSearch.h"
2
3 int linearSearch(MyArray const &mya, int valSearchFor)
4 {
5     for (unsigned int i = 0; i < mya.size; ++i) {
6         if (maya.arr[i] == valSearchFor)
7             return i;
8     }
9     return -1;
10 }
```

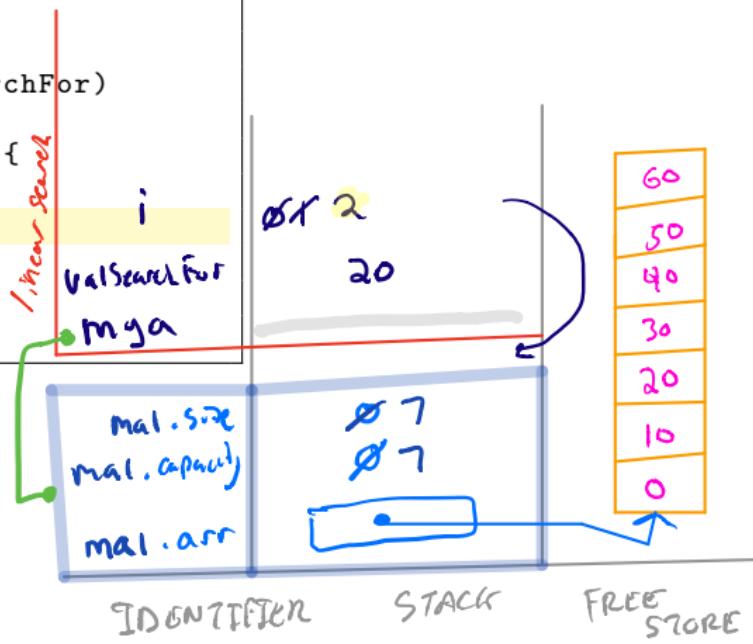
Linear Search



## linearSearch()

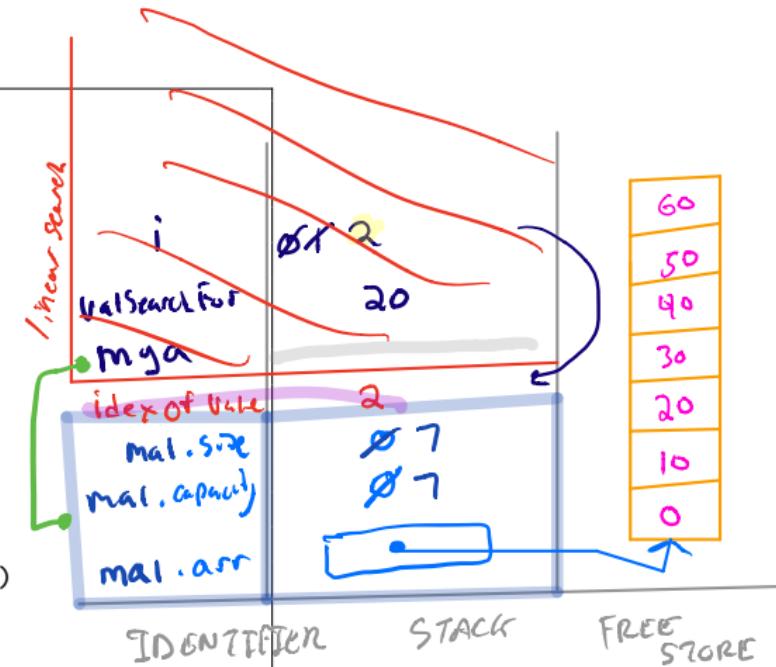
```
1 #include "linearSearch.h"
2
3 int linearSearch(MyArray const &mya, int valSearchFor)
4 {
5     for (unsigned int i = 0; i < mya.size; ++i) {
6         if (maya.arr[i] == valSearchFor)
7             return i;
8     }
9     return -1;
10 }
```

Linear Search



## Returned to main

```
1 #include <iostream>
2 #include "MyArray.h"
3 #include "linearSearch.h"
4 #include "binarySearch.h"
5
6 using namespace std;
7
8 int main()
9 {
10     MyArray ma1;
11     ma1.arr = new int[7];
12     ma1.capacity = 7;
13     ma1.size = 7;
14     for (unsigned int i = 0; i < ma1.size; ++i)
15         ma1.arr[i] = i * 10;
16
17     unsigned int idxofvalue = linearSearch(ma1, 20);
18
19     unsigned int idxofvalue2 = binarySearch(ma1, 20);
20
21     delete [] ma1.arr;
22
23     return 0;
24 }
```



# Overview

MyArray

Searching for some value in an array

Linear Search

Binary Search



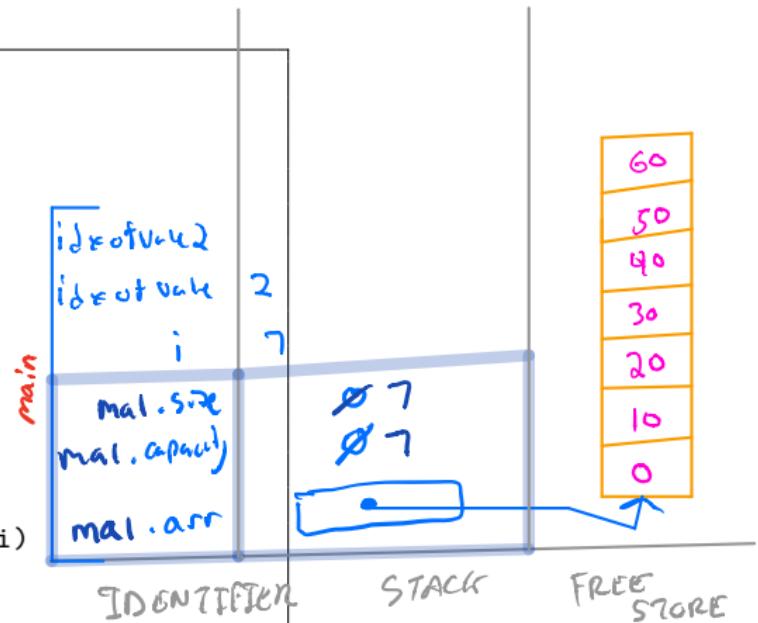
→ more Complete Picture of  
main's activation record

# Binary Search

- Requirement: data is sorted!

## Calling our binary search function

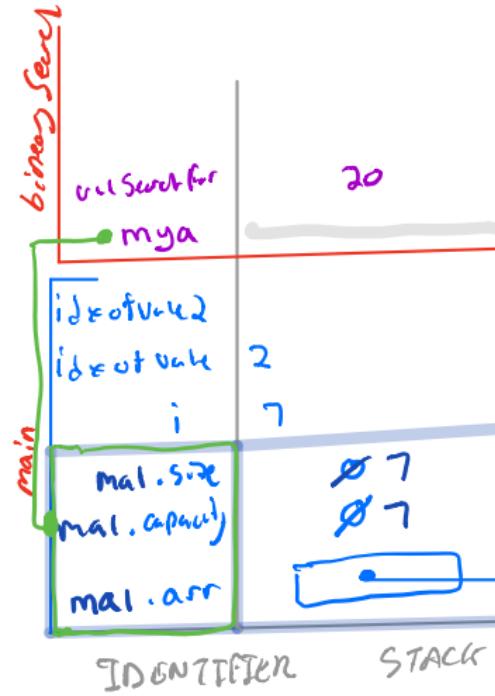
```
1 #include <iostream>
2 #include "MyArray.h"
3 #include "linearSearch.h"
4 #include "binarySearch.h"
5
6 using namespace std;
7
8 int main()
9 {
10     MyArray ma1;
11     ma1.arr = new int[7];
12     ma1.capacity = 7;
13     ma1.size = 7;
14     for (unsigned int i = 0; i < ma1.size; ++i)
15         ma1.arr[i] = i * 10;
16
17     unsigned int idxofvalue = linearSearch(ma1,
18
19     unsigned int idxofvalue2 = binarySearch(ma1,
20
21     delete [] ma1.arr;
22
23     return 0;
24 }
```



## binarySearch()

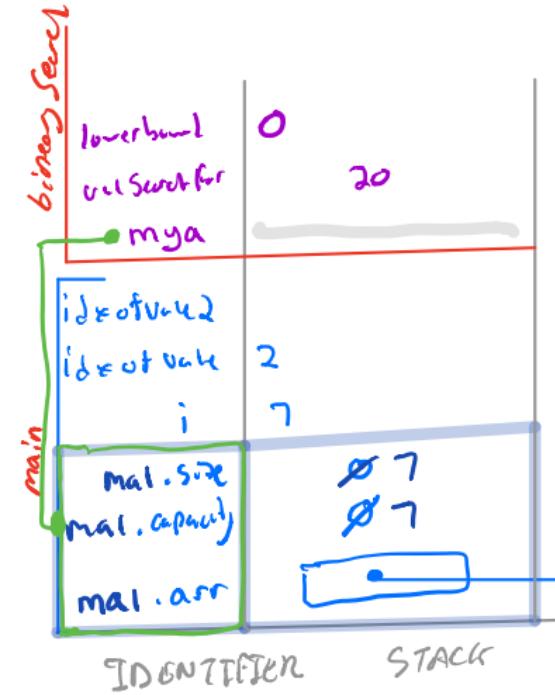
```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &mya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (mya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (mya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17     }
18     return -1;
19 }
```

## binarySearch()



```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &mya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (maya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (maya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17     }
18     return -1;
19 }
```

## binarySearch()



```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &mya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (mya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (mya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17     }
18     return -1;
19 }
```

## binarySearch()

binarySearch

upperbound 6  
lowerbound 0  
valSearchFor 20  
mya

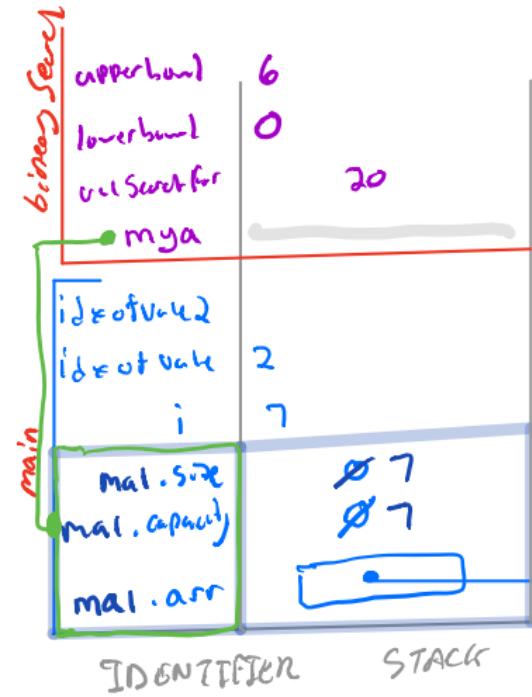
main  
indexofValue  
idxtutake  
i 7  
MyArray  
mat.size 7  
mat.capacity 7  
mat.arr

IDENTIFIER

STACK

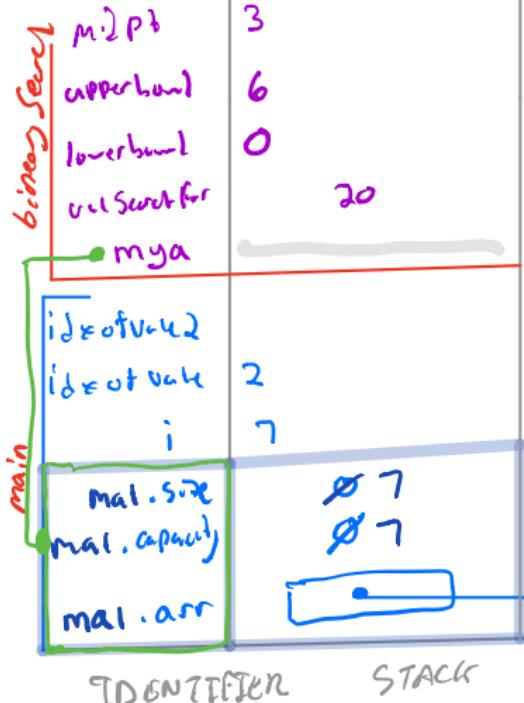
```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &mya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (maya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (maya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17     }
18     return -1;
19 FREE STORE
```

## binarySearch()



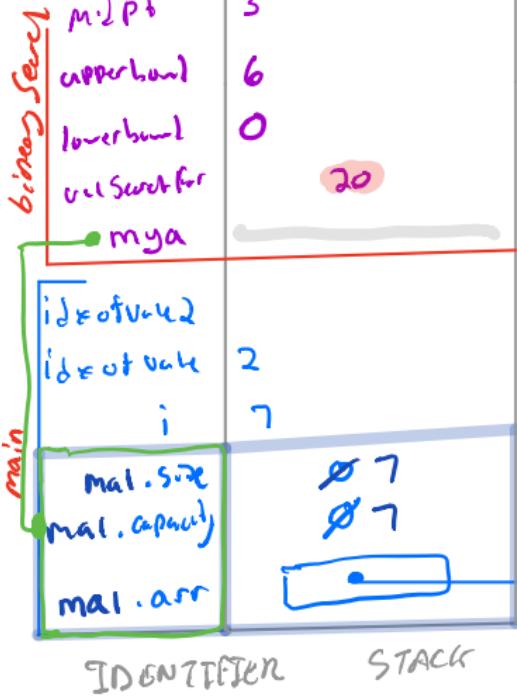
```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &mya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (maya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (maya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17     }
18     return -1;
19 }
```

## binarySearch()



```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &mya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (mya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (mya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17     }
18     return -1;
19 }
```

## binarySearch()



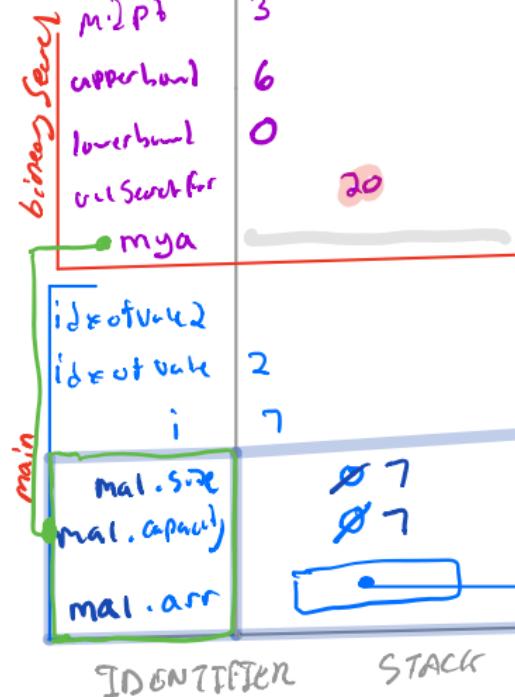
```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &mya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (maya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (maya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17     }
18     return -1;
19 }
```

## binarySearch()



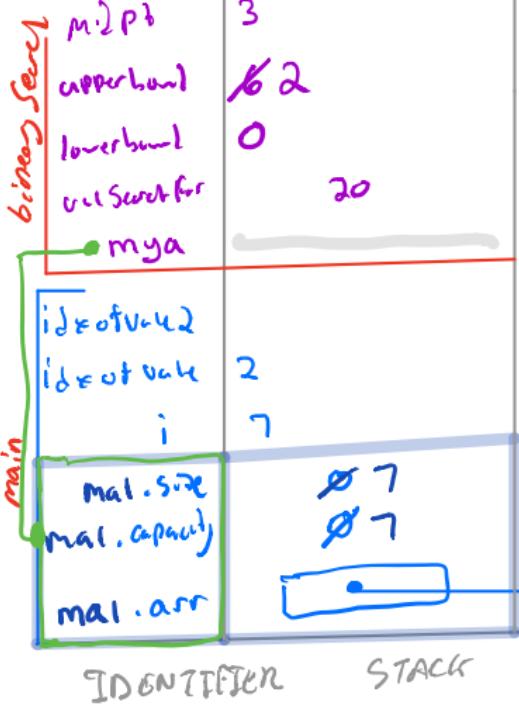
```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &mya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (mya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (mya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17     }
18     return -1;
19 }
```

## binarySearch()



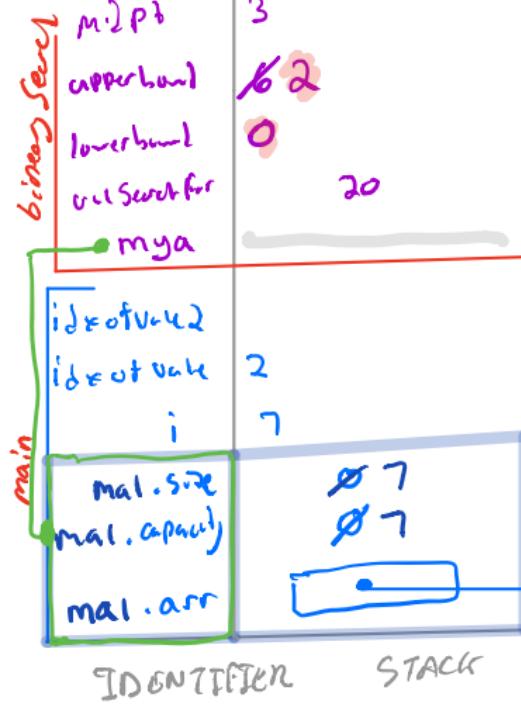
```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &mya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (mya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (mya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17     }
18     return -1;
19 FREE STORE
```

## binarySearch()



```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &mya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (mya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (mya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17     }
18     return -1;
19 }
```

## binarySearch()



```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &mya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (maya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (maya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17     }
18     return -1;
19 }
```

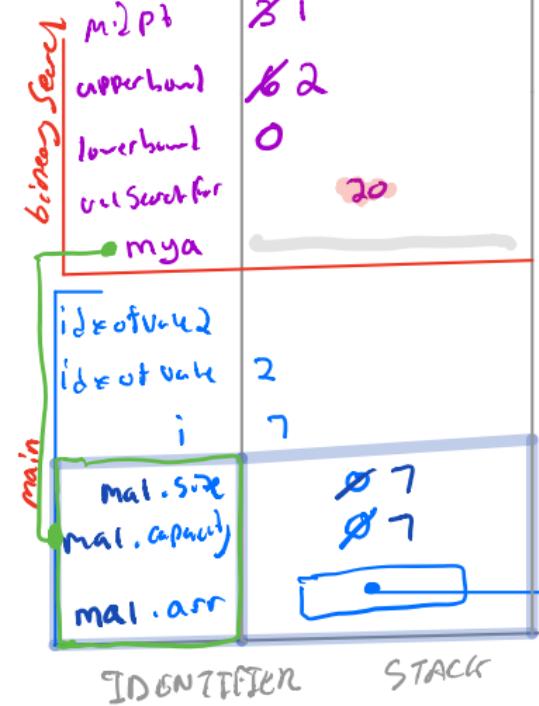
## binarySearch()

binarySearch  
main



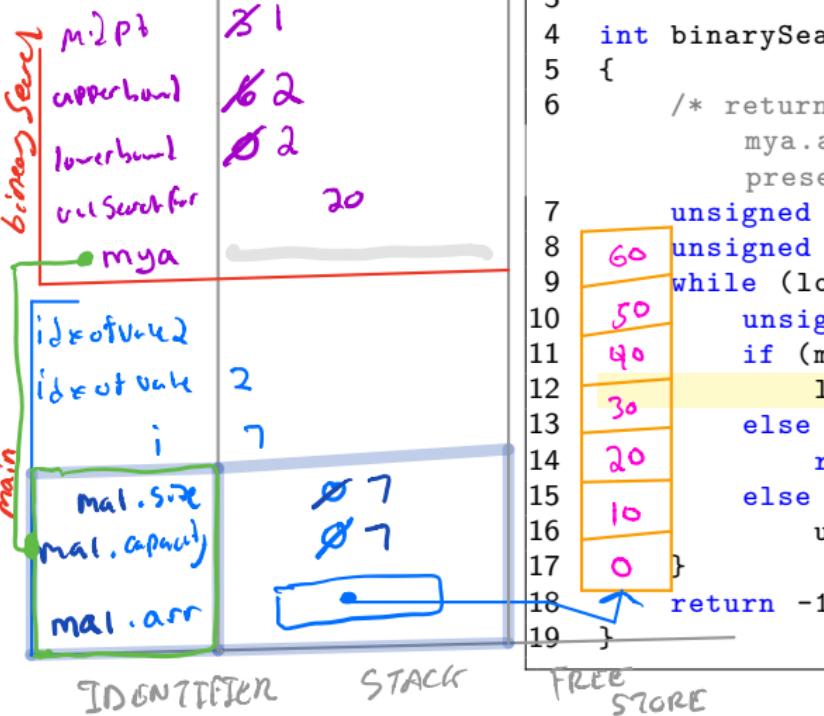
```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &maya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (maya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (maya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17     }
18     return -1;
19 }
```

## binarySearch()



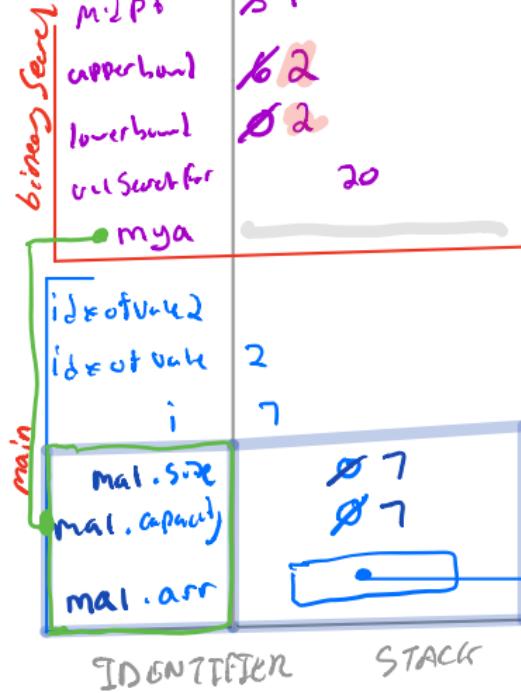
```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &mya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (maya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (maya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17     }
18     return -1;
19 }
```

## binarySearch()



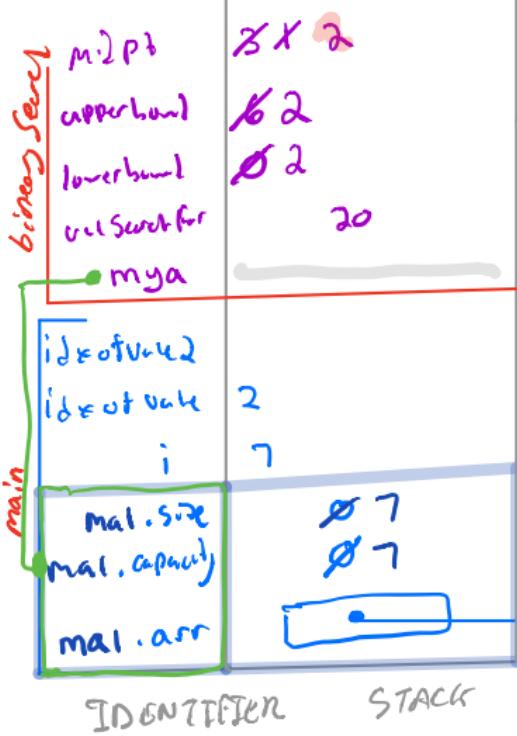
```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &mya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (mya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (mya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17     }
18     return -1;
19 }
```

## binarySearch()



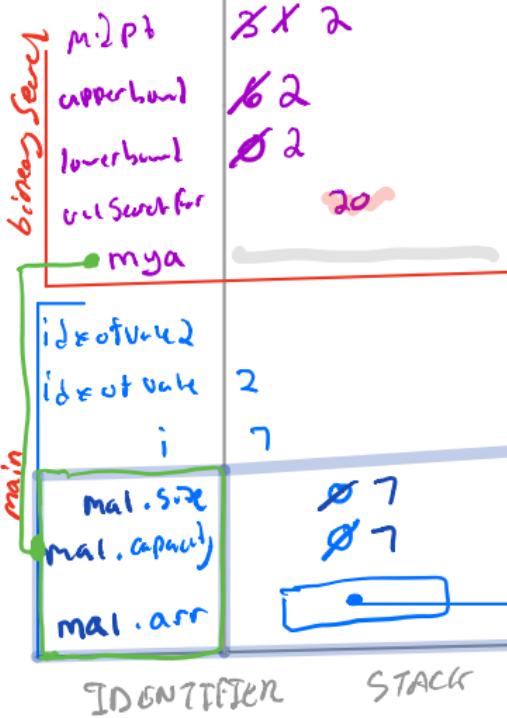
```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &mya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (mya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (mya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17     }
18     return -1;
19 }
```

## binarySearch()



```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &mya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (mya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (mya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17
18     return -1;
19 }
```

## binarySearch()



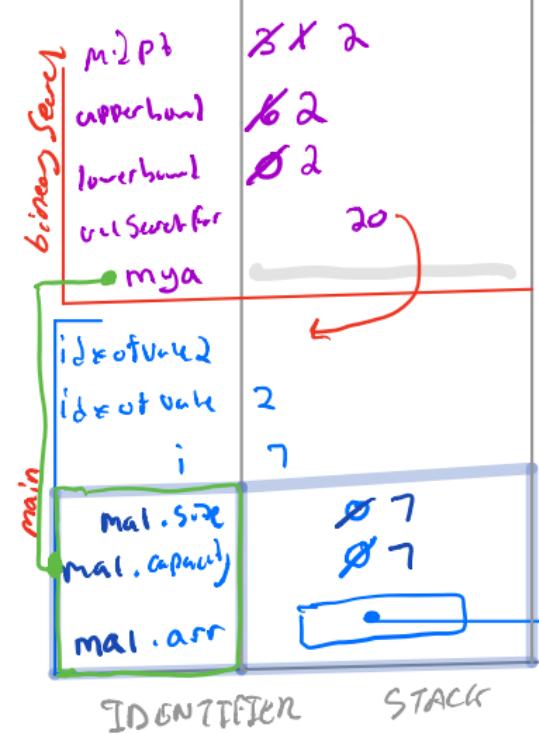
```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &mya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (maya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (maya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17     }
18     return -1;
19 }
```

## binarySearch()



```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &mya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (maya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (maya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17     }
18     return -1;
19 }
```

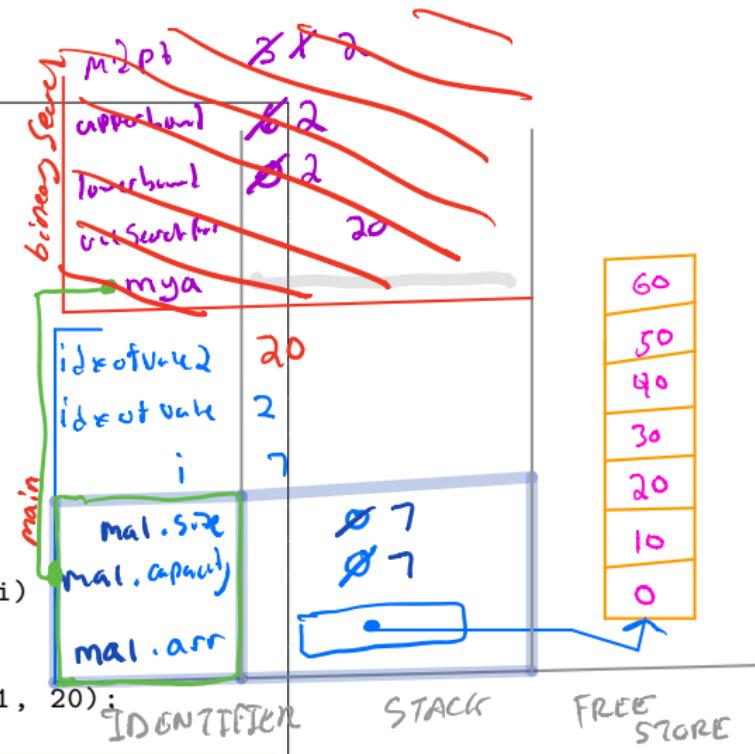
## binarySearch()



```
1 #include "binarySearch.h"
2 #include <iostream>
3
4 int binarySearch(MyArray const &mya, int valSearchFor)
5 {
6     /* return (any) position if valSearchFor is in sorted
      mya.arr[0..size-1] or -1 if valSearchFor is not
      present */
7     unsigned int lowerBound = 0;
8     unsigned int upperBound = mya.size - 1;
9     while (lowerBound <= upperBound) {
10         unsigned int midpt = (lowerBound + upperBound) / 2;
11         if (maya.arr[midpt] < valSearchFor)
12             lowerBound = midpt + 1;
13         else if (maya.arr[midpt] == valSearchFor)
14             return midpt;
15         else /* mya.arr[midpt] > valSearchFor */
16             upperBound = midpt - 1;
17     }
18     return -1;
19 }
```

Returned to [main](#)

```
1 #include <iostream>
2 #include "MyArray.h"
3 #include "linearSearch.h"
4 #include "binarySearch.h"
5
6 using namespace std;
7
8 int main()
9 {
10     MyArray ma1;
11     ma1.arr = new int[7];
12     ma1.capacity = 7;
13     ma1.size = 7;
14     for (unsigned int i = 0; i < ma1.size; ++i)
15         ma1.arr[i] = i * 10;
16
17     unsigned int idxofvalue = linearSearch(ma1,
18
19     unsigned int idxofvalue2 = binarySearch(ma1
20
21     delete [] ma1.arr;
22
23     return 0;
24 }
```



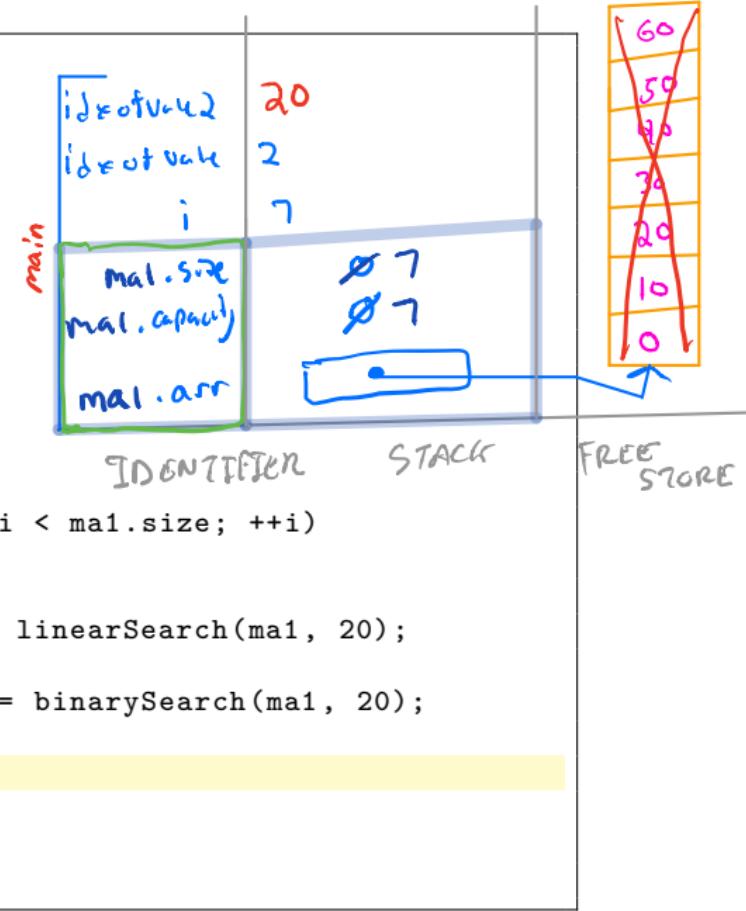
Returned to [main](#)

```
1 #include <iostream>
2 #include "MyArray.h"
3 #include "linearSearch.h"
4 #include "binarySearch.h"
5
6 using namespace std;
7
8 int main()
9 {
10     MyArray ma1;
11     ma1.arr = new int[7];
12     ma1.capacity = 7;
13     ma1.size = 7;
14     for (unsigned int i = 0; i < ma1.size; ++i)
15         ma1.arr[i] = i * 10;
16
17     unsigned int idxofvalue = linearSearch(ma1, 20);
18
19     unsigned int idxofvalue2 = binarySearch(ma1, 20);
20
21     delete [] ma1.arr;
22
23     return 0;
24 }
```

The diagram illustrates the state of variables in memory across different scopes:

- main**: A red box labeled "main". Inside, there is a green box labeled "ma1". The "ma1" box contains three entries: "ma1.size", "ma1.capacity", and "ma1.arr".
- idxofvalue2**: A blue box labeled "idxofvalue2". It contains the value "20".
- idxofvalue**: A blue box labeled "idxofvalue". It contains the value "2".
- i**: A blue box labeled "i". It contains the value "7".
- ma1.arr**: A blue box labeled "ma1.arr". It contains a pointer to memory.

Handwritten annotations include "IDENTIFIER" written below the "main" box and "ST" written next to the "ma1.arr" box.



## Returned to main

```
1 #include <iostream>
2 #include "MyArray.h"
3 #include "linearSearch.h"
4 #include "binarySearch.h"
5
6 using namespace std;
7
8 int main()
9 {
10     MyArray ma1;
11     ma1.arr = new int[7];
12     ma1.capacity = 7;
13     ma1.size = 7;
14     for (unsigned int i = 0; i < ma1.size; ++i)
15         ma1.arr[i] = i * 10;
16
17     unsigned int idxofvalue = linearSearch(ma1, 20);
18
19     unsigned int idxofvalue2 = binarySearch(ma1, 20);
20
21     delete [] ma1.arr;
22
23     return 0;
24 }
```

