CSCI 136:
Data Structures
and
Advanced Programming

Lecture 16

Search

Instructor: Dan Barowy

Williams

Announcements

Midterm exam

Wednesday during your lab period in assigned lab

Exam review session: Tonight 7-8pm in TCL 202

No class Wednesday

No class Friday

We are going to try to get Lab 4 back before Wed

Outline

Search

Binary search

100	101	322	365	423	478	499	504
0	1	2	3	4	5	6	7

Want to know **whether** the array contains the value **322**, and if so, what its **index** is.

Binary search is a **divide-and-conquer** algorithm that solves this problem.

Binary search is **fast**: in the **worst case**, it returns an answer in **$O(\log_2 n)$** steps.

Binary search

100	101	322	365	423	478	499	504
0	1	2	3	4	5	6	7

Important precondition: array must be **sorted**.

Binary search

Looking for the value **322**.

100	101	322	365	423	478	499	504
0	1	2	3	4	5	6	7

Binary search

Looking for the value **322**.

100	101	322	365	423	478	499	504
0	1	2	3	4	5	6	7

Binary search

Looking for the value **322**.

100	101	322	365	423	478	499	504
0	1	2	3	4	5	6	7

Binary search

Looking for the value **322**.

100	101	322	365	423	478	499	504
0	1	2	3	4	5	6	7

Binary search

Looking for the value **322**.

100	101	322	365	423	478	499	504
0	1	2	3	4	5	6	7

322 = 365? **no**

322 < 365? **yes**

Binary search

Looking for the value **322**.

100	101	322	365	423	478	499	504
0	1	2	3	4	5	6	7

Binary search

Looking for the value **322**.

100	101	322	365	423	478	499	504
0	1	2	3	4	5	6	7

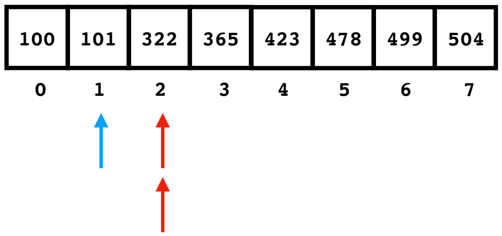
322 = 101? **no**

322 < 101? **no**

322 > 101? **yes**

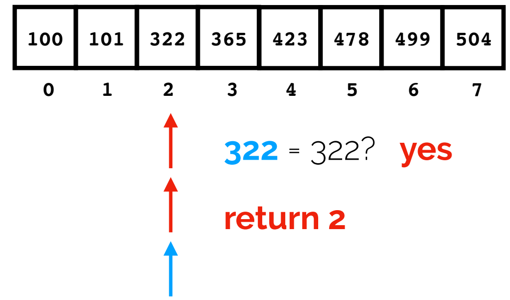
Binary search

Looking for the value **322**.



Binary search

Looking for the value **322**.



Binary search

(code)

Binary search

```
public static int search(int[] a, int value) {
    return searchRec(a, value, 0, a.length - 1);
}


protected static int searchRec(int[] a, int value, int low, int high) {
    if (low > high) {
        return -1;
    }
    int mid = (high - low)/2 + low;
    if (value == a[mid]) {
        return mid;
    } else if (value < a[mid]) {
        return searchRec(a, value, low, mid - 1);
    } else {
        return searchRec(a, value, mid + 1, high);
    }
}
```

Binary search

Binary search is **fast**: in the **worst case**, it returns an answer in **$O(\log_2 n)$** steps.

How can we **prove** this claim?

Principle of Mathematical Induction (weak induction)

Let **P(n)** be a **predicate** that is defined for **integers n**, and let **a** be a **fixed integer**.

If the following two statements are **true**:

1. **P(a)** is **true**.
2. For all integers **$k \geq a$**, **if P(k)** is **true then P(k + 1)** is **true**.

then the statement

for all integers **$n \geq a$**, **P(n)** is **true**

is **also true**.

Principle of Mathematical Induction (strong induction)

Let **P(n)** be a **predicate** that is defined for **integers n**, and let **a** be a **fixed integer**.

If the following two statements are **true**:

1. **P(a)** is **true**.
2. Whenever **P(0),P(1),...,P(k)** are **true then P(k + 1)** is **true**.

then the statement

for all integers **$n \geq a$**, **P(n)** is **true**

is **also true**.

Binary search

(proof)

Binary search

(code: count calls)

Recap & Next Class

Today we learned:

Search

Next class:

Midterm