CSE 114

Exam 2 Review

Exam Format

- 80 minutes
- In-person on Tuesday (October 20)
    - Can use 1 page of handwritten A4 page of notes (both sides)
    - No other class materials allowed
    - No computer / phone / internet
- Problems can consist of:
    - True/false, multiple choice, short answer, code analysis, writing code, fix or modify existing code examples, memory diagrams

New Topics Covered Since Exam 1

Arrays

- Creating, accessing, and using arrays
- Passing arrays to functions
- How arrays differ from primitive data types
- Searching arrays
- Sorting arrays
- 2D Arrays

Memory Diagrams

- How Java handles storing variables
  - Stack vs Heap
- Effect of method calls, variable scope
- How objects and primitive data types are handled
- Creating and reading memory diagrams

Classes & Objects

- Adding static fields to a class
- Using multiple files
- Creating and using objects
  - Creating constructors
  - Using dynamic variables and methods

Some Additional Points Covered

- Using command line arguments
- Using break/run/continue
- How to benchmark algorithms

Further Refresher On Some Points

Pass by value

- Java uses Pass-by-value to pass arguments to methods
- For Primitive Types (int, float, etc)
  - Actual value or contents of the variable is passed
  - Changing value inside method does NOT affect the caller’s copy of the variable
- For Array Type
  - Reference (memory address) is passed to method
  - Changes to elements of array affect the caller’s copy

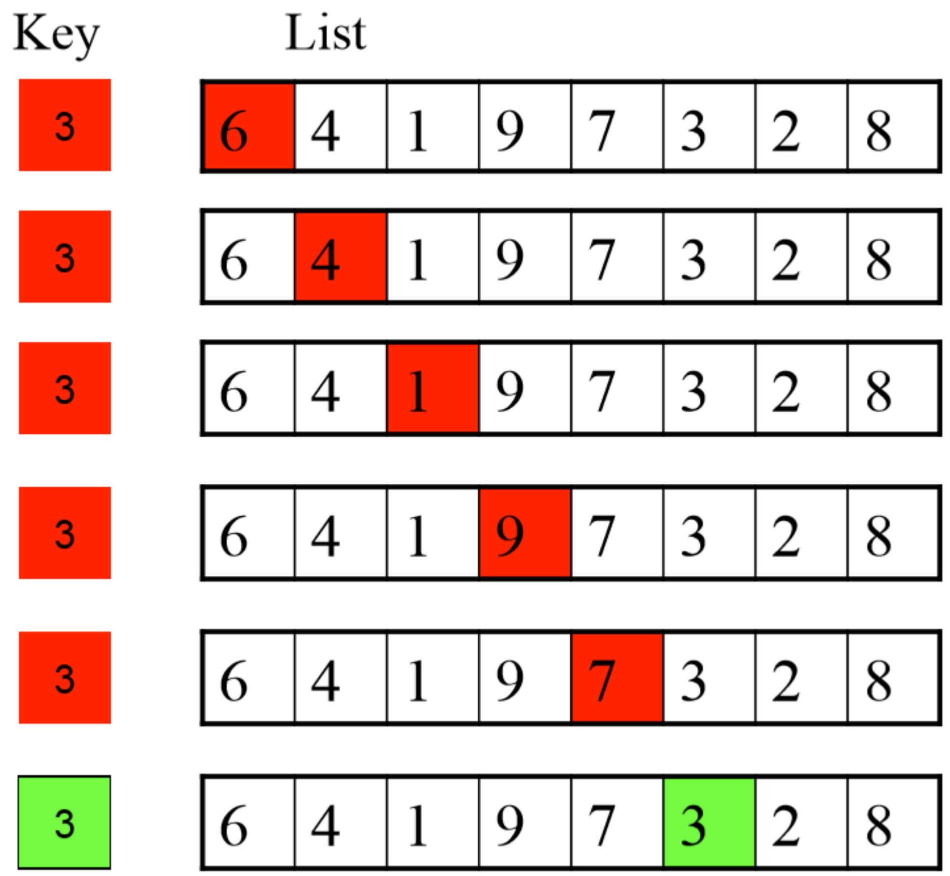
The length of an array

- After an array is created
  - its size is fixed and cannot be changed
  - Can find its size using
    - arrayRefVar.length
  - Note ➔ not .length()

Linear Search

- Linear Search Algorithm
  1. Compare the element being searched with element 0 in array
  2. If it matches, exit returning index where element was found
  3. If it does not match, advance to next element
  4. Compare element being searched
  5. Repeat 2-4 until an element matches or array is exhausted
  6. If element was not found, return -1

Linear Search Animation



Finding the largest element

```
double max = myList[0];
for (int i = 1; i < myList.length; i++) {
    if (myList[i] > max) {
        max = myList[i];
    }
}
```

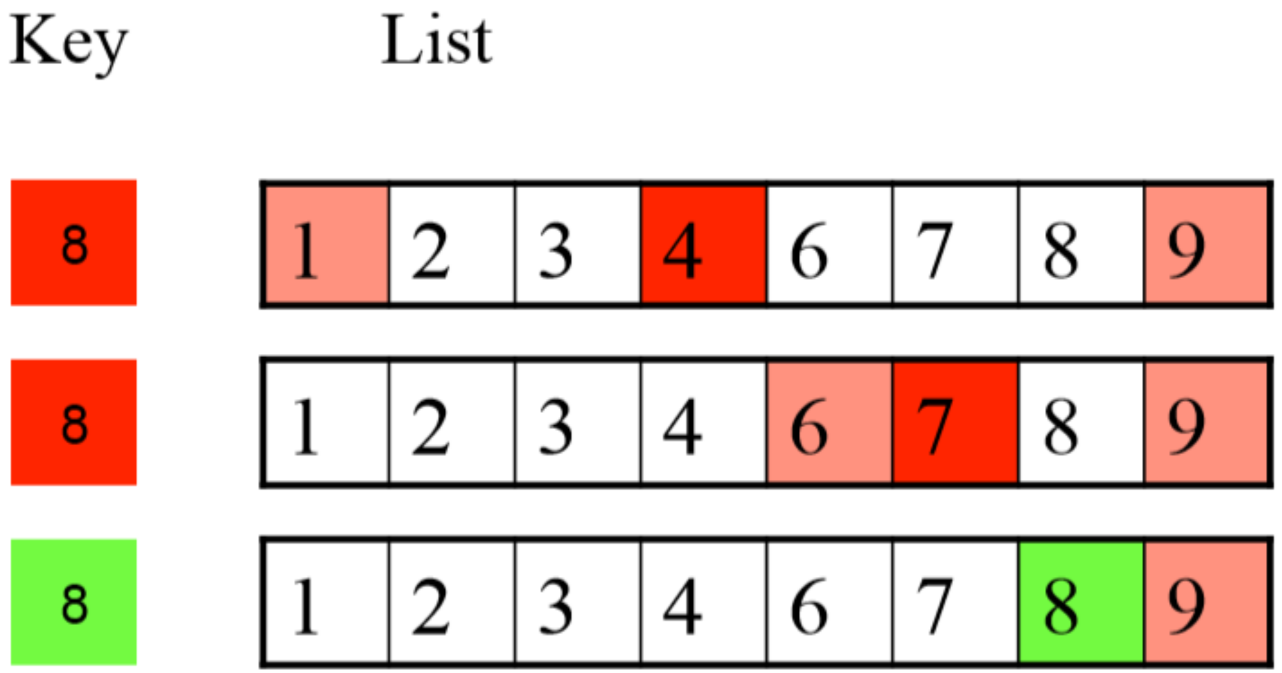
Binary Search

- Binary Search Requirements
    - Elements must be already ordered
    - Prefer ascending order
    - e.g. 2 4 7 10 11 45 50 59 60 66 69 70 79

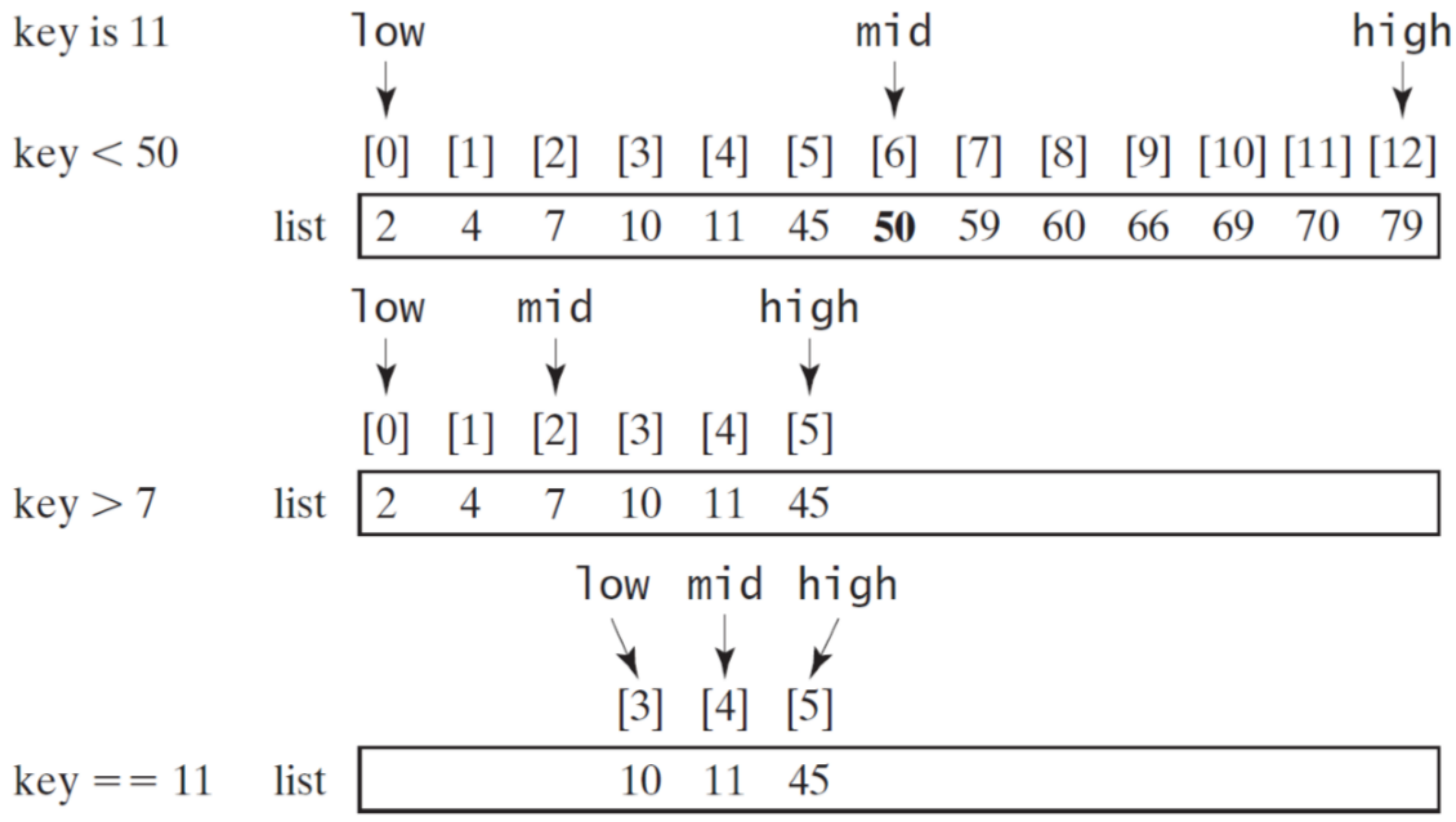
Binary Search Algorithm

- Binary Search Algorithm
  1. Select middle element
  2. Compare key to element
  3. If key matches the element, return the element’s index
  4. If key is less than element, key is in first half of array or array section if present at all. Divide element count in 2 to select middle of first half
  5. If key is greater than element, key is in latter half of array section, Divide latter half count by 2 to select middle of latter half
  6. Repeat 2-5 until element found or no elements left in the remaining segment

Binary search



Binary search (cont.)



Sorting arrays

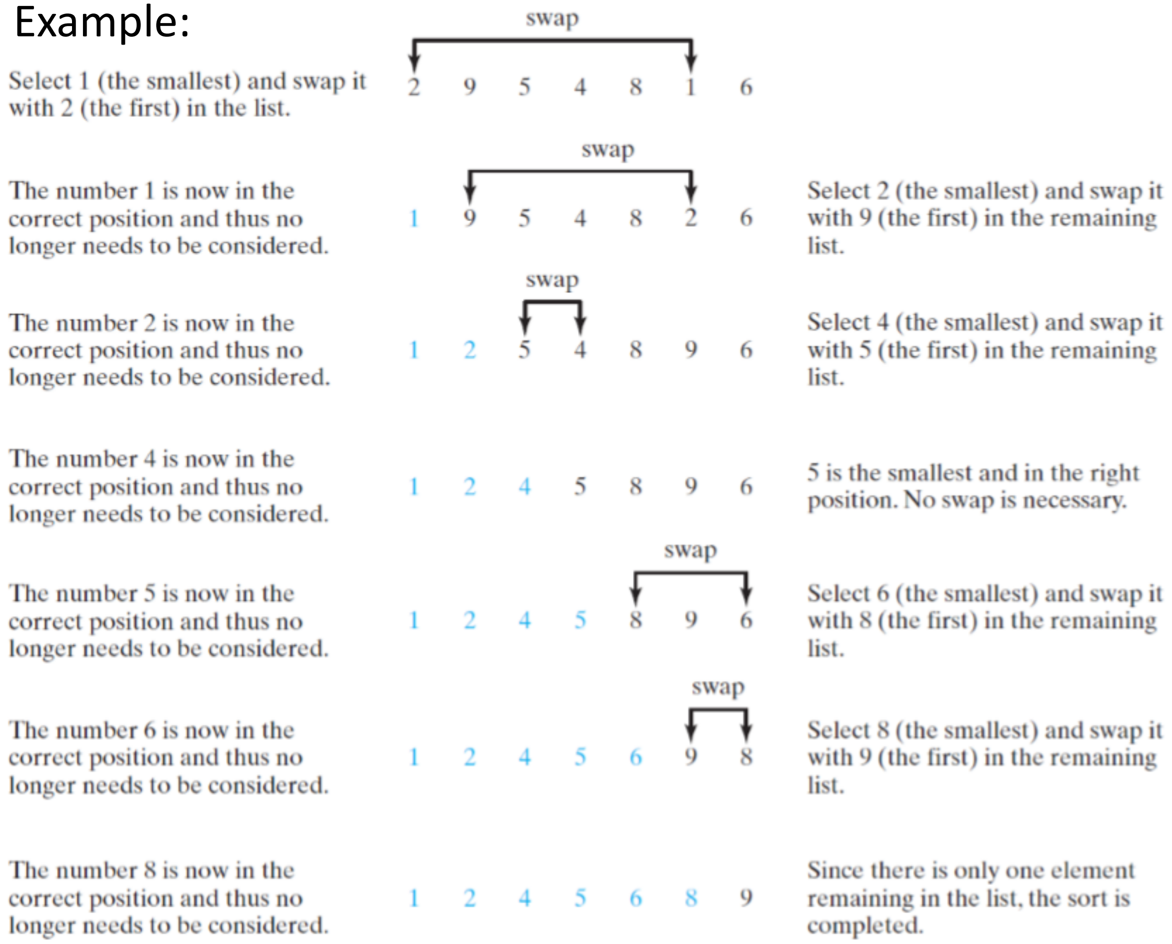
- Sorting is a common task in computer programming.
- Many different algorithms have been developed for sorting.
  - We covered selection sort, insertion sort, and bubble sort

Selection sort

- Selection Sort Algorithm
    1. Find the smallest number in the list
    2. Swap the smallest number with the first item in array
    3. Find the smallest number remaining
    4. Swap it with the next item (e.g. second, third…)
    5. Repeat 3-4 until array exhausted

Selection sort

Example:



Select 1 (the smallest) and swap it with 2 (the first) in the list.

2 9 5 4 8 1 6

The number 1 is now in the correct position and thus no longer needs to be considered.

1 9 5 4 8 2 6

Select 2 (the smallest) and swap it with 9 (the first) in the remaining list.

The number 2 is now in the correct position and thus no longer needs to be considered.

1 2 5 4 8 9 6

Select 4 (the smallest) and swap it with 5 (the first) in the remaining list.

The number 4 is now in the correct position and thus no longer needs to be considered.

1 2 4 5 8 9 6

5 is the smallest and in the right position. No swap is necessary.

The number 5 is now in the correct position and thus no longer needs to be considered.

1 2 4 5 8 9 6

Select 6 (the smallest) and swap it with 8 (the first) in the remaining list.

The number 6 is now in the correct position and thus no longer needs to be considered.

1 2 4 5 6 9 8

Select 8 (the smallest) and swap it with 9 (the first) in the remaining list.

The number 8 is now in the correct position and thus no longer needs to be considered.

1 2 4 5 6 8 9

Since there is only one element remaining in the list, the sort is completed.