Exam 1 Review

Array, Time Complexity, Stack

COMP128 Data Structures

Exam 1 Overview

- Covers LG 1 Arrays, LG2 Time Complexity, LG3 Stacks
- Paper Exams
- Open notes, open textbook, open Moodle resources, open prior programs from this class

Topics We Covered So Far

- Arrays
- Time Complexity Analysis
- Collections
- Stacks
  - Linked Nodes

Topics We Covered So Far

- Arrays
- Linked Nodes
- Time Complexity Analysis
- Collections
- Stacks

Basic Data Structures (Building Blocks)

Analysis Tool

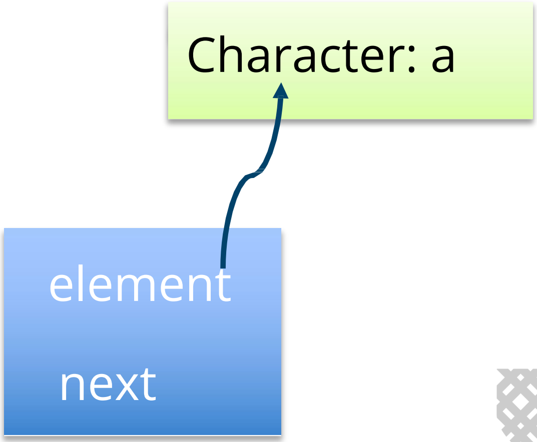
Abstract Data Types

Arrays

```
// Declare and initialize an array
    String[] words = new String[10];

// Assigning and indexing
        words[0] = "words";

// Iterating
for(int i = 0; i < words.length; i++ ){
    System.out.println(words[i]);
}

// Iterating
for (String word : words){
    System.out.println(word);
}
```

Linked Nodes

```
public class LinearNode<E> {
   private LinearNode<E> next;
   private E element;
   /**
    * Creates a node storing the specified element.
    * @param elem the element to be stored within the new node
    **/
   public LinearNode(E elem){
      next = null;
      element = elem;
   }

… getters and setters: getNext, setNext,
getElement, setElement
```



Big-O Analysis

```
// O(1)
public void swap(int[] array, int left, int right) {
    int temp = array[left];
    array[left] = array[right];
    array [right] = temp;
}
// O(n)
public void sum(int n) {
    int sum = 0;
    for (int j = 0; j < n; j++)
        sum += j;
}
// O(logn)
public void test(int[] array) {
    for(int i=array.length; i > 0; i /= 2){
        System.out.println(i);
    }
}
```

Practice Question (LG2)

What’s the time complexity of this example?

```
public void test(int[] array) {
    for(int i=array.length; i > 0; i /= 2){
        for (int k = 0; k < array.length; k = k + 2)
            System.out.println(k);
        }
    }
}
```

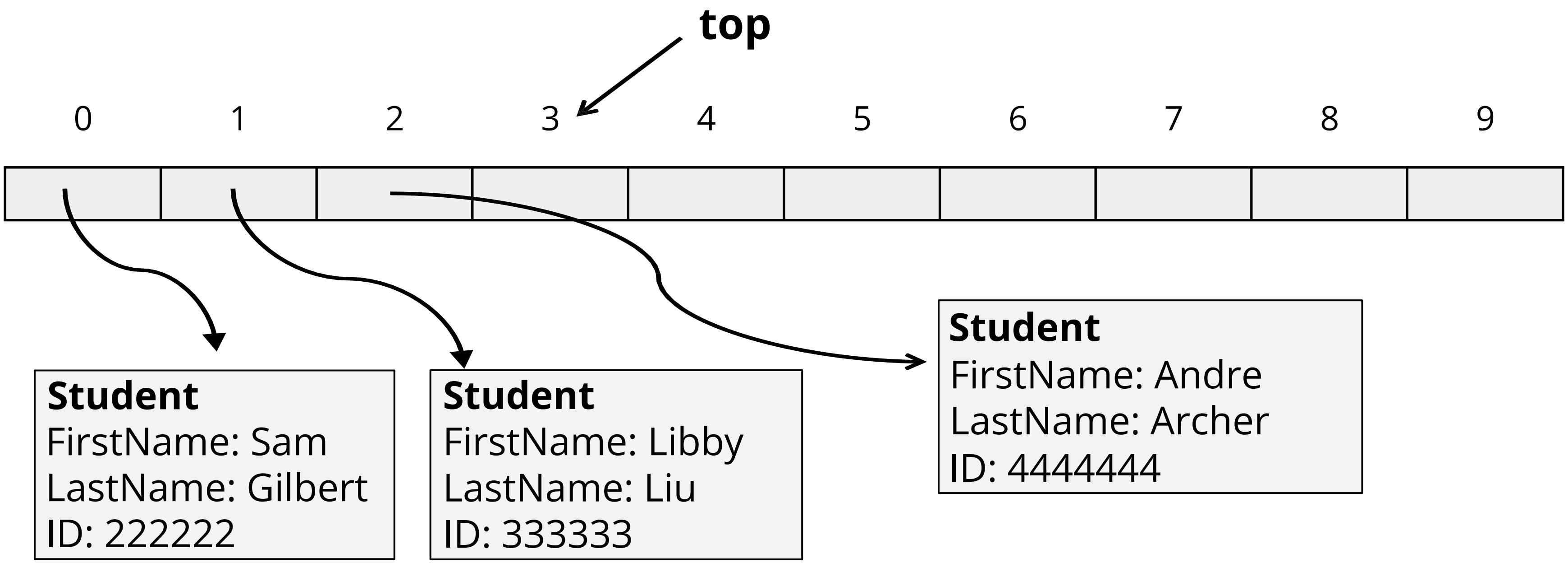
Stacks (LIFO)

Operation	Description
push(Object item)	Add an object to the top of the stack
pop()	Remove an object from the top of the stack
peek()	Return the top object in the stack
isEmpty()	Check if the stack is empty
size()	Return the size of the stack

Working with Stacks

- **Java Collections**
  - Deque Interface
  - ArrayDeque Class
- **Foundations Textbook**
  - StackADT Interface
  - ArrayStack Class
  - LinkedStack Class

Stacks Implementation: ArrayStack

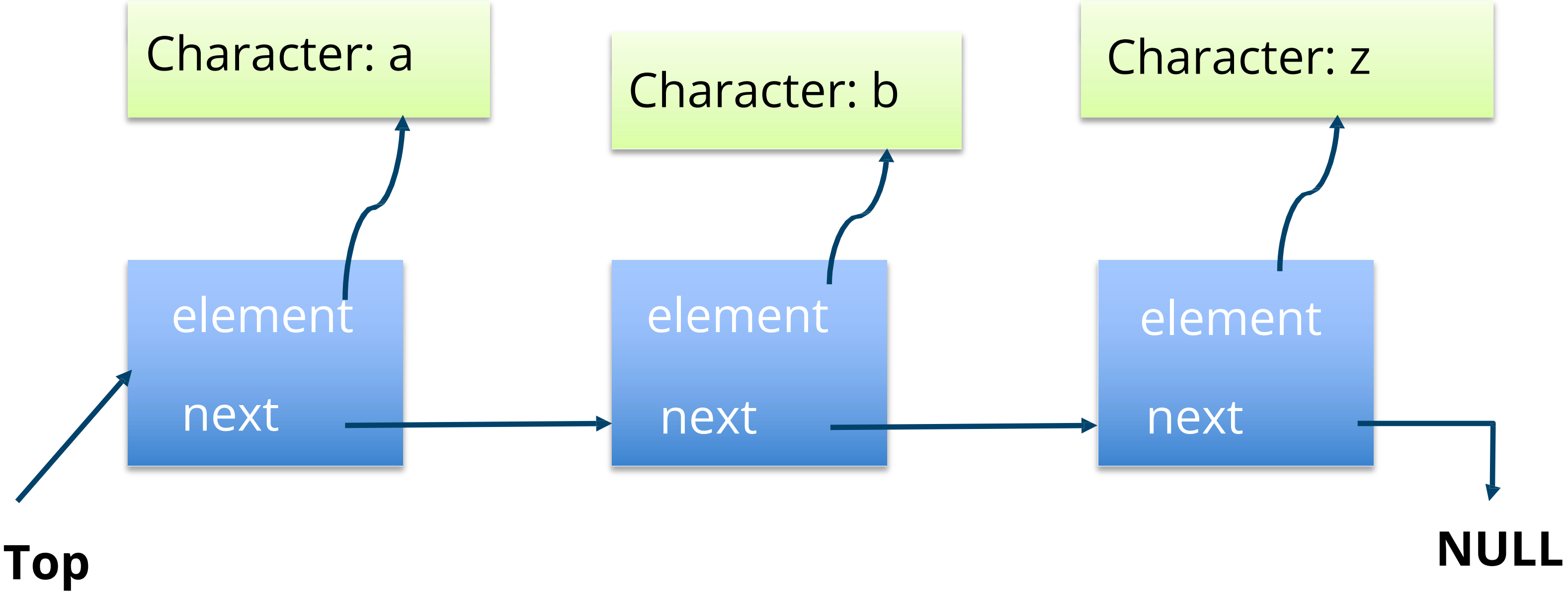


Student
FirstName: Sam
LastName: Gilbert
ID: 222222

Student
FirstName: Libby
LastName: Liu
ID: 333333

Student
FirstName: Andre
LastName: Archer
ID: 4444444

Stacks Implementation: LinkedStack



Practice Question (LG3)

Take a piece of paper see if you can write a method called removeNegatives that takes a stack of Integer objects as a parameter and returns the stack with all negative numbers removed. For example, if the original stack contained 30, -15, 20, -25 (top of stack), the new stack would look like 30, 20 (top of stack).

```
public Deque<Integer> removeNegatives(Deque<Integer> inStack){







}
```

Queues (FIFO)

Operation	Description
enqueue(Object item)	Add an object to the end of the queue
dequeue()	Remove an object from the beginning of the queue
first()	Return the first element in the queue
isEmpty()	Check if the queue is empty
size()	Return the size of the queue

Working with Queues

- **Java Collections**
  - Queue Interface
  - LinkedList Class
- **Foundations Textbook**
  - QueueADT Interface
  - LinkedQueue Class
  - CircularArrayQueue Class

In-class Activity

Concurrent Web Spider Activity

Exam 1

Practice Questions

LG1: Arrays

1. Write the Java code that creates and initializes an integer array named myIntArray with the following values: 10, 20, 30, 40.

2. Write the code for the following method that returns the maximum value found in the input array.

LG2: Time Complexity

1. What is the time complexity of the following code in big-o notation. Briefly explain your answer.

```
int sum = 0;
for(int i=0; i<n; i++) {
  for(int j=0; j<n/2; j++) {
    sum = sum + j;
  }
}
```

LG2: Time Complexity

2. What is the Big-O complexity for the following algorithm for determining who stole the cookie from the cookie jar:

> *Ask the first person if they stole the cookie; if they say it couldn't be them, you ask the second person if they stole the cookie; etc, for each person in the room.*

LG3: Stacks

1. Write a method called removeNegatives that takes a stack of Integer objects as a parameter and returns the stack with all negative numbers removed. For example, if the original stack contained 30, -15, 20, -25 (top of stack), the new stack would look like 30, 20 (top of stack).

LG3: Stacks

2. Which implementation of a stack (array-based or linked-node based) would be most efficient in terms of time complexity to use for the inStack parameter variable in the above question. Briefly justify your answer using big-o notation. If both implementations would be the same, state why.