CS 2110, SP24

Discussion 6: Prelim 1 Review

Topics

- Procedural programming in Java
- Compile-time and runtime
- Classes
- Testing
- Object-oriented programming
- Exceptions
- Data Structures
- Efficiency

Procedural programming in Java

Classify the following as either a *primitive type*, a *reference type*, or *not a type name*:

- `Object`
- `char`
- `5`
- `String`
- `null`
- `int[]`

Predict the result of running the below program

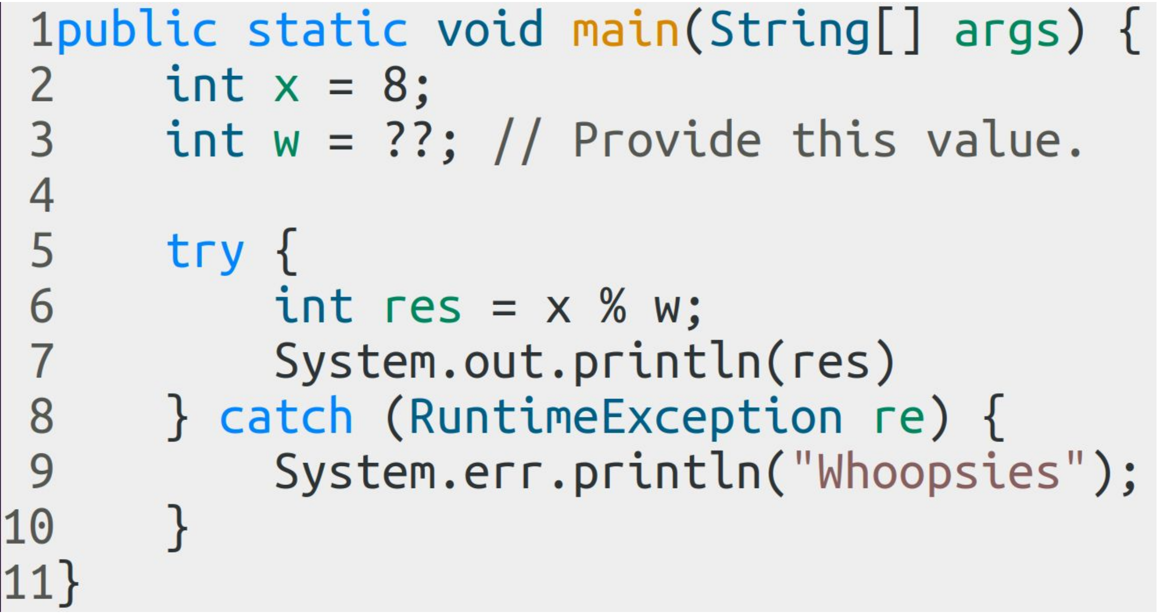
```
int[] arr = new int[] {1, 2, 4, 8, 16, 32, 64, 128};
for (int i = 0; i < arr.length; i += 1) {
   int temp = arr[arr.length - i - 1];
   arr[arr.length - i - 1] = arr[i];
   arr[i] = temp;
}
```

Complete this short method given the specification

```
/** Returns a new String with the characters of s in reverse order.
  * ex. reverseString("hello") => "olleh".
  * Requires: s is not null.
  * You may not use any Java methods or classes beyond length(),
  * charAt(), and concatenation operators. */

public static String reverseString(String s) {

    // Your code here!

}
```

Compile-time and run-time

Give an initialization value of `w` that…?



```java
public static void main(String[] args) {
    int x = 8;
    int w = ??; // Provide this value.

    try {
        int res = x % w;
        System.out.println(res)
    } catch (RuntimeException re) {
        System.err.println("Whoopsies");
    }
}
```

1. Causes a compile-time error.
   a. In this case, do any of our print statements run?
2. Causes an `ArithmeticException` to be thrown.
   a. In this case, what gets printed?
3. Causes 0 to be printed.

Given the following class hierarchy and code:

```
interface I1 { }

interface I2 { }

class A implements I2 { }

class B extends A implements I1, I2 { }
```

```
// Main Method

B b = new B();

I2 i2 = b;
```

Determine if the following code compiles, and if not, specify whether there is a runtime or compile-time error.

a) `I1 k = (I2) b;`

b) `I1 k2 = b;`

c) `I1 k3 = i2;`

d) `String s = i2.toString();`

Classes in Java

Class Diagrams

Given the following class, please draw a class diagram:

```java
public class Student {
    private String name;
    private String netId;
    private int credits;

    public String name() {
       return name;
    }

    public String netId() {
       return netId;
    }

    public void modifyCredits(int creditChange) {
       credits += creditChange;
    }
}
```

Label the return type, parameters, specification, keywords, types and literals in the method below:

```
 /**
  * This method returns true if every character in String word consists of
  * lowercase english alphabet ('a' - 'z'), and false if otherwise.
  * Requires: word is not null or empty ("").
  */
 public static boolean isAllLowerCase(String word) {
   for (int i = 0; i < word.length(); i++) {
     char currentChar = word.charAt(i);
     if (currentChar < 'a' || currentChar > 'z') {
       return false;
     }
   }
   return true;
 }
}
```

Implement isSolved() according to the specification

```
/** A class representing a single row of cells in a Sudoku game */
public class SudokuRow {
    /** The values in each of the cells in the row.
     * Each element is either filled with a number 1-9 or is an empty cell, marked by a 0
     * Invariant: Only contains values in the range 0-9 inclusive.
     * Invariant: Each number in range 1-9 inclusive can only appear at most once in the row.
     */
    private int[] cells;

    // Other fields, constructors, and methods omitted

    /** Returns whether the row has been solved. A row has been solved if there are no empty cells
in the row
    */
    public boolean isSolved() {
        //TODO
    }
}
```

Testing

Given the method specification, write at least three **black box tests**, stating the input and expected output

Recap: Black box testing is a technique of testing where the functionality of the software is tested by only looking at the specifications and without looking at the code.

```
 /**
  * Returns the average sum of the first k elements of arr. If arr is empty,
  * returns 0, and if k > arr.length, returns the average sum of all elements in
  * arr.
  *
  * Requires: k > 0, arr is not null
  */
 public double averageOfFirstKElements(int[] arr, int k) {
     //implementation here
}
```

Object-oriented programming in Java

What will happen when we try to compile and run A and B?

```
public class Animal {
        public void makeNoise() {
                System.out.println(“This animal is making its call”);
                call();
        }

        public void call() {
                System.out.println(“Grunt”);
        }
}

public class Cat extends Animal {
        public void call() {
                System.out.println(“Meow”);
        }

        public void pet() {
                System.out.println(“Purr”);
        }
}
```

A

```
public static void main(String args[]) {
        Animal oliver = new Cat();
        oliver.makeNoise();
}
```

B

```
public static void main(String args[]) {
        Animal oliver = new Cat();
        oliver.pet();
}
```

Does the following equals() method for the Player class satisfy all the properties of an equivalence relation? If not, which ones does it violate

```
public class Player {

    public String playerName;

    public int jerseyNo;

    public String team;

        public boolean equals(Object obj) {

            if (!obj instance of Player) {return false;}

            Player pl = (Player) obj;

            if (this.jerseyNo > pl.jerseyNo) {

              return this.playerName.equals(pl.playerName)

                  && this.team.equals(pl.team);

            }

            return this.playerName.equals(pl.playerName);

        }

}
```

Does Class SuperSonics implement Interface NBATeam? Are there any compile-time errors?

(There are no specifications, so we can’t say whether the implementation is *correct*; we’re just interested in whether it compiles for now.)

```java
public interface NBATeam {

    public double winPercent();

    public String nextGame();

}
```

```java
public class SuperSonics implements NBATeam {
    int gamesPlayed;
    double winPercent;
    String[] schedule;
    public SuperSonics(){
        gamesPlayed = 0;
        this.winPercent = 0.0;
        this.schedule = null;
        // the team no longer exists, so the schedule will
        always be null
    }
    public double winPercent() {
        return winPercent;
    }
    public String nextGame() {
        return schedule[gamesPlayed];
    }
}
```

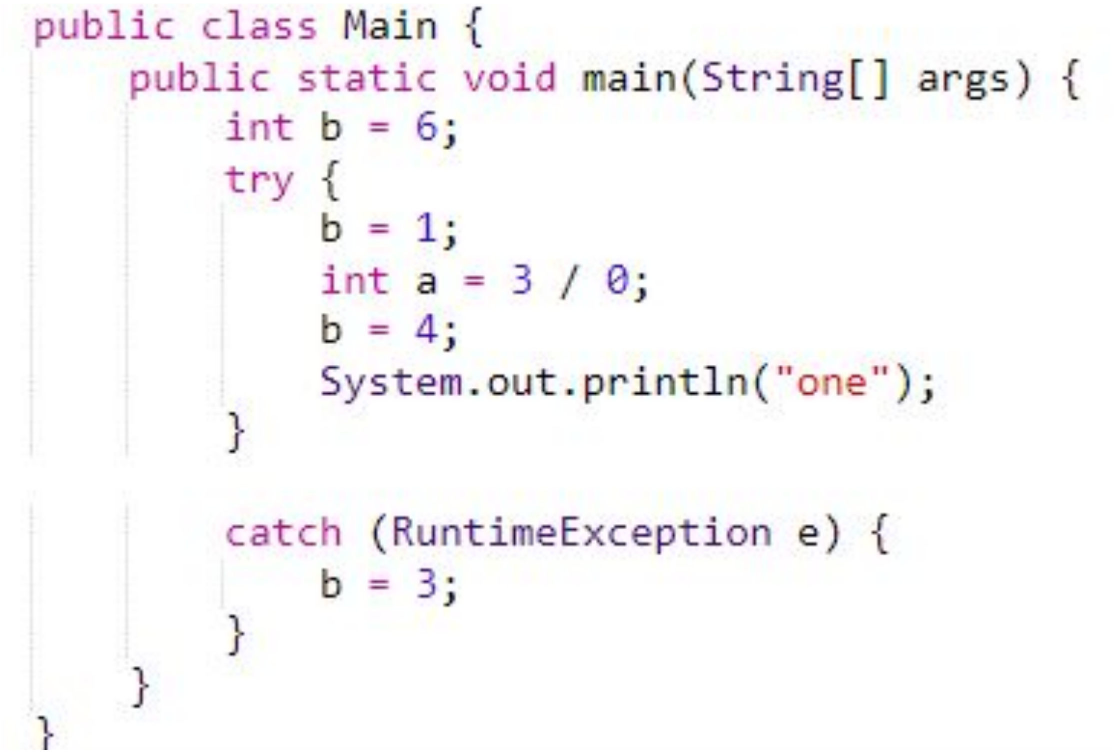
Exceptions

Exceptions: Try-Catch

(1) Does this try block throw an exception? If so what exception?

(2) What is the final value of the variable b (if the program does not crash)?

(3) What is printed out?



```
public class Main {
    public static void main(String[] args) {
        int b = 6;
        try {
            b = 1;
            int a = 3 / 0;
            b = 4;
            System.out.println("one");
        }

        catch (RuntimeException e) {
            b = 3;
        }
    }
}
```

Convert the following method to throw an `Exception` instead of returning -1:

```java
public int indexOf(char input) {
    // Iterate over each character in String
    for (int i = 0; i < this.length(); i++) {
        // If current character equals input character
        if (this.charAt(i) == input) {
            return i; // Return the current index
        }
    }

    return -1; // Character not found, return -1
}
```

Data structures

Examine the following Java class for a linked node:

```
public class Node<T> {
    private Node<T> next;
    private T data;
    public Node(T init, Node<T> nextNode) {
        data = init;
        next = nextNode;
    }
    // No other methods exist.
}
```

Complete the following tasks:

1. Create (with Java code) a chain of 3 Nodes that contain the strings “Lorem”, “Ipsum”, and “Dolor” in order.
2. Create (with Java code) a chain of 2 Nodes that point to the same String array (i.e. they reference the exact same object); the array should contain {“Lorem”, “Ipsum”, “Dolor”}.

Explain why the following real-world data / ADT pairs would be unsuitable.

1. The items in a student's backpack / List
2. Tasks that need to be completed for a project / Bag
3. The line to order flatbreads at Mac's / List

Match the following real world data to the most appropriate ADT Options (Bag, List, Stack, Queue)

1. The previous web pages visited by a user which is used by the browser when they click the back button
2. The jobs needed to be completed by a printer

Time complexity

What is the best case and worst case time complexity for the following? Let N denote the size of the list

1. Adding an element at a specified position in a singly linked list
2. Adding an element at a specified position in a doubly linked list
3. Getting the previous node in a singly linked list (given the current node)
4. Getting the previous node in a doubly linked list (given the current node)
5. Getting an element at a specified row and column in a table implemented as a singly linked list (M rows) of singly linked lists (up to N columns)
6. Appending an element to a fixed-capacity queue implemented with a circular array
7. Appending an element to an unbounded queue implemented with a dynamic array

Implementation of a Stack

Using a **linked structure** approach, a **Stack** can be represented by a `Node<T>` field called head that is the most recent item that was added to the stack. The `Node<T>` class has methods `data()` which returns the node's data and `next()` which returns the node containing the item that was added before it. An empty stack has a null head.

- Implement the pop operation `pop()` which removes the node at the top of the stack and returns that node's data as a result. Throws an `EmptyStackException` if the stack is empty.

Efficiency

Big Oh Notation

- Show that $5x^2+2x+1$ is in $O(n^2)$
- Show that $10+10x$ is in $O(n)$
- Show that $x+5$ is in $O(n^2)$

Given the following problems, state what quantity describes the problem’s size and state the algorithm’s worst case time complexity (in terms of that size) in Big Oh notation

1. Computing the mean of an array of integers
2. For some Set, enumerate every subset of size 2