Scope, Pass-by-Value, Static

Exam-Level 01

Announcements

- Welcome to CS 61B!
- Please read our Ed guidelines before you post to make sure everything follows the rules
- Pre-Semester Survey: due Monday 1/22 at 11:59pm PT
- Homework 0B: due Monday 1/22 at 11:59pm PT
- Project 0: due Monday 1/29 at 11:59pm PT

Meet Your TA!

Add an introduction here. Make sure to make your own copy of the slides before editing, and change the location to your own Drive (not our shared 61B one).

Some things you can include:

- Your name
- Your pronouns
- Your email address
- Your major and year
- Maybe your hobbies, interests, favorites, etc so students can relate to you as a human being
- Maybe a fun picture of you that shows your ✨sparkle ✨

Content Review

Quick Java Basics

```
public class Hello {
    public static void main(String[] args) {
        System.out.println(“Hello world!”);
    }
}
```

- In Java, pretty much everything is defined in a class
- Type declarations: Java is statically typed, so we have to tell the computer what type of value every variable holds and what every function returns (ie. `int, void`)
- Don’t forget the brackets and semicolons!

Structure of a Class

```
public class CS61BStudent { // Class Declaration
    public int idNumber; // Instance Variables
    public int grade;
    public static String instructor = "Hug"; // Class (Static) Variables
    public CS61BStudent (int id) { // Constructor
        this.idNumber = id; // this refers to the instance of the CS61BStudent we are in
        this.grade = 100;
    }

    public void watchLecture() { // Instance Method
        ...
    }
    public static String getInstructor() { // Class (Static) Method
        ...
    }
}
```

Instantiating Classes

```
public class CS61BLauncher {
    public static void main(String[] args) {
        CS61BStudent studentOne; // Declare a new variable of class CS61BStudent
        studentOne = new CS61BStudent(32259); // Instantiate and assign to our new instance
        CS61BStudent studentTwo = new CS61BStudent(19234); // Both at once

        studentOne.watchLecture(); // Instance methods are called on instance

        CS61BStudent.getInstructor(); // Static methods can be called on the class OR the
                                      instance

        CS61BStudent.watchLecture(); // Fails. Which student is watching lecture?
        studentOne.getInstructor(); // Works, though is seen as bad practice.
    }
}
```

Overview: Static vs. Instance

Static variables and functions belong to the whole class.
Example: Every 61B Student shares the same instructor, and if the instructor were to change it would change for everyone.

Instance variables and functions belong to each individual instance.
Example: Each 61B Student has their own ID number, and changing a student's ID number doesn't change anything for any other student.

Check for understanding: can you reference instance variables in static methods? Can you reference static variables in instance methods?

*Don't worry if you don't fully understand the difference right now! We'll talk more about this in future discussions

Worksheet

1A Quik Maths

```
public static void multiplyBy3(int[] A) {
    for (int i = 0; i < A.length; i += 1) {
        int x = A[i];
        x = x * 3;
    }
}

public static void multiplyBy2(int[] A) {
    int[] B = A;
    for (int i = 0; i < B.length; i+= 1) {
        B[i] *= 2;
    }
}

public static void swap (int A, int B) {
    int temp = B;
    B = A;
    A = temp;
}
```

```
public static void main(String[] args) {
    int[] arr;
    arr = new int[]{2, 3, 3, 4};
    multiplyBy3(arr);

    /* Value of arr: {_____________________} */

    arr = new int[]{2, 3, 3, 4};
    multiplyBy2(arr);

    /* Value of arr: {_____________________} */

    int a = 6;
    int b = 7;
    swap(a, b);

    /* Value of a: _______,  Value of b: _______ */
}
```

1A Quik Maths

```
public static void multiplyBy3(int[] A) {
    for (int i = 0; i < A.length; i += 1) {
        int x = A[i];
        x = x * 3;
    }
}

public static void multiplyBy2(int[] A) {
    int[] B = A;
    for (int i = 0; i < B.length; i+= 1) {
        B[i] *= 2;
    }
}

public static void swap (int A, int B) {
    int temp = B;
    B = A;
    A = temp;
}
```

```
public static void main(String[] args) {
    int[] arr;
    arr = new int[]{2, 3, 3, 4};
    multiplyBy3(arr);

    /* Value of arr: {2, 3, 3, 4} */

    arr = new int[]{2, 3, 3, 4};
    multiplyBy2(arr);

    /* Value of arr: {4, 6, 6, 8} */

    int a = 6;
    int b = 7;
    swap(a, b);

    /* Value of a: 6, Value of b: 7 */
}
```

Java Visualizer Link

1B Quik Maths

Write code to simulate “swapping” of primitive values.

```
class IntWrapper {
    int x;
    public IntWrapper(int value) {
        x = value;
    }
}

public class SwapPrimitives {
    public static void main(String[] args) {
        int a = 6;
        int b = 7;
        ______________________________________;
        ______________________________________;
        swap(__________, __________);
        a = ___________________________________;
        b = ___________________________________;
    }
}
```

```
public static void swap(__________, __________) {
    ______________________________________;
    ______________________________________;
    ______________________________________;
}
```

1B Quik Maths

Write code to simulate "swapping" of primitive values.

```
class IntWrapper {
    int x;
    public IntWrapper(int value) {
        x = value;
    }
}

public class SwapPrimitives {
    public static void main(String[] args) {
        int a = 6;
        int b = 7;
        IntWrapper first = new IntWrapper(a);
        IntWrapper second = new IntWrapper(b);
        swap(__________, __________);
        a = ____________________________________;
        b = ____________________________________;
    }
}
```

```
public static void swap(__________, __________) {
    ____________________________________;
    ____________________________________;
    ____________________________________;
}
```

1B Quik Maths

Write code to simulate "swapping" of primitive values.

```
class IntWrapper {
    int x;
    public IntWrapper(int value) {
        x = value;
    }
}

public class SwapPrimitives {
    public static void main(String[] args) {
        int a = 6;
        int b = 7;
        IntWrapper first = new IntWrapper(a);
        IntWrapper second = new IntWrapper(b);
        swap(__________, __________);
        a = ____________________________________;
        b = ____________________________________;
    }
}
```

```
public static void swap(IntWrapper first,
IntWrapper second) {
    ____________________________________;
    ____________________________________;
    ____________________________________;
}
```

1B Quik Maths

Write code to simulate "swapping" of primitive values.

```
class IntWrapper {
    int x;
    public IntWrapper(int value) {
        x = value;
    }
}

public class SwapPrimitives {
    public static void main(String[] args) {
        int a = 6;
        int b = 7;
        IntWrapper first = new IntWrapper(a);
        IntWrapper second = new IntWrapper(b);
        swap(__________, __________);
        a = ___________________________________;
        b = ___________________________________;
    }
}
```

```
public static void swap(IntWrapper first,
IntWrapper second) {
    int temp = first.x;
    first.x = second.x;
    second.x = temp;
}
```

1B Quik Maths

Write code to simulate "swapping" of primitive values.

```
class IntWrapper {
    int x;
    public IntWrapper(int value) {
        x = value;
    }
}

public class SwapPrimitives {
    public static void main(String[] args) {
        int a = 6;
        int b = 7;
        IntWrapper first = new IntWrapper(a);
        IntWrapper second = new IntWrapper(b);
        swap(first, second);
        a = ___________________________________;
        b = ___________________________________;
    }
}
```

```
public static void swap(IntWrapper first,
IntWrapper second) {
    int temp = first.x;
    first.x = second.x;
    second.x = temp;
}
```

1B Quik Maths

Write code to simulate "swapping" of primitive values.

```
class IntWrapper {
    int x;
    public IntWrapper(int value) {
        x = value;
    }
}

public class SwapPrimitives {
    public static void main(String[] args) {
        int a = 6;
        int b = 7;
        IntWrapper first = new IntWrapper(a);
        IntWrapper second = new IntWrapper(b);
        swap(first, second);
        a = first.x;
        b = second.x;
    }
}
```

```
public static void swap(IntWrapper first,
IntWrapper second) {
    int temp = first.x;
    first.x = second.x;
    second.x = temp;
}
```

2A Static Books

```
class Book {
    public String title;
    public Library library;
    public static Book last = null;

    public Book(String name) {
        title = name;
        last = this;
        library = null;
    }

    public static String lastBookTitle() {
        return last.title;
    }
    public String getTitle() {
        return title;
    }
}
```

```
class Library {
    public Book[] books;
    public int index;
    public static int totalBooks = 0;

    public Library(int size) {
        books = new Book[size];
        index = 0;
    }

    public void addBook(Book book) {
        books[index] = book;
        index++;
        totalBooks++;
        book.library = this;
    }
}
```

For each modification, determine whether the code of the Library and Book classes will compile or error. Treat each modification independently.

1. Change the `totalBooks` variable to non `static`
2. Change the `lastBookTitle` method to non `static`
3. Change the `addBook` method to `static`
4. Change the `last` variable to non `static`
5. Change the `library` variable to `static`

2A Static Books

```
class Book {
    public String title;
    public Library library;
    public static Book last = null;

    public Book(String name) {
        title = name;
        last = this;
        library = null;
    }

    public static String lastBookTitle() {
        return last.title;
    }
    public String getTitle() {
        return title;
    }
}
```

```
class Library {
    public Book[] books;
    public int index;
    public static int totalBooks = 0;

    public Library(int size) {
        books = new Book[size];
        index = 0;
    }

    public void addBook(Book book) {
        books[index] = book;
        index++;
        totalBooks++;
        book.library = this;
    }
}
```

For each modification, determine whether the code of the Library and Book classes will compile or error. Treat each modification independently.

1. Compile
2. Compile
3. Error
4. Error
5. Compile

2B Static Books

```
System.out.println(Library.totalBooks);
System.out.println(Book.lastBookTitle());
System.out.println(Book.getTitle());

Book goneGirl = new Book("Gone Girl");
Book fightClub = new Book("Fight Club");

System.out.println(goneGirl.title);
System.out.println(Book.lastBookTitle());
System.out.println(fightClub.lastBookTitle());
System.out.println(goneGirl.last.title);

Library libraryA = new Library(1);
Library libraryB = new Library(2);
libraryA.addBook(goneGirl);

System.out.println(libraryA.index);
System.out.println(libraryA.totalBooks);

libraryA.totalBooks = 0;
libraryB.addBook(fightClub);
libraryB.addBook(goneGirl);

System.out.println(libraryB.index);
System.out.println(Library.totalBooks);
System.out.println(goneGirl.library.books[0].title);
```

Using the Book and Library classes from before, write the output of each System.out.println() line. If a line errors, put the precise reason it errors and continue execution.

2B Static Books

```
0
Error, NullPointerException
Error, does not compile

Gone Girl
Fight Club
Fight Club
Fight Club

1
1

2
2
Fight Club
```

Java Visualizer Link