Lec 19

Conclusion
• Turn in extra credit by Friday.
• Projects due midnight Dec 2\textsuperscript{nd}
• Next class in CSE 1202. Project demos

• Practice final posted by Thursday
• Final review Friday in Spencer’s discussion
• Final exam in this room, Monday 7-10pm
Honorable Mentions

Jonah Cruz, Crystal Jiao, Ziqi Deng, Yaxuan Li, Qiaoran Dong, Xiaowei Lu, Mengran Wang, Jesse and Sharon, Cesar Lopez, Nayeli Ruiz, Edward Guo, Zhe Wang, Abhi and Brian, Richard San Miguel, Jiyuiae and Wei, Allen Chang, John Lieu
Vote

A) Black and green
B) Light blue
C) Dissolves
Vote

A) dissolves
B) Black and red and drag and drop
C) Black and blue
D) pokemon
5. Given the following directory structure:

```
/  
  /home/  /etc/  /proc/
    /cse8b/  /cse11/  
      /hw1/  /hw2/  
```

Typing `pwd` into the terminal returns `/home/cse8b/hw1`

a) What is the command, using relative paths, to change directory into `hw2` ________________

b) What is the command, using absolute paths, to change directory into `cpu` ________________

c) What is the command to list all files in the current directory ____________________________

d) You just ran `pwd` and you’re in `/home/cse8b/hw2` and you have the files `Question5.java` and `apache.jar`. `Question5.java` makes use of class definitions in apache.jar. What is the command to compile `Question5.java`? ______________________________

e) What is the command to run the Java program created from (d) ____________________________
public class Test {
    public static void newMethod(int[] i, int j) {
        i[0]++;
        j++;
        return;
    }
    public static void main(String[] args) {
        int[] i = {10, 20};
        newMethod(i, i[1]);
        System.out.println(i[0] + " " + i[1]);
    }
}

A) 10 20
B) 11 20
C) 10 21
D) 11 21
E) None
import java.util.Scanner;

public class Ex3 {
    public static void main(String[] args) {
        String[] students = new String[2];
        getStudents(students);
        System.out.println(students[0]);
    }
    public static void getStudents(String[] students) {
        Scanner scnr = new Scanner(System.in);
        int i = 0;
        String s;
        while(true) {
            System.out.print("enter name (Quit to exit): ");
            s = scnr.next();
            if(s.equals("Quit")) {
                break;
            }
            else {
                students[i] = s;
                i++;
            }
        }
    }
}

Run with: Adam Ali Uma Quit
I/O

• When/why would you use:
  – Scanner
  – FileInputStream
  – FileOutputStream
  – BufferedInputStream
  – BufferedOutputStream
  – PrintWriter
Exceptions
public class Test {
    public static void main(String[] args) {
        writeArray();
    }

    public static void writeArray() throws IOException {
        FileOutputStream fs = null;
        PrintWriter pw = null;
        int[] a = {0, 1, 2};
        fs = new FileOutputStream("out.txt");
        pw = new PrintWriter(fs, true);
        for (int i = 0; i < a.length; i++) {
            pw.println(a[i]);
        }
    }
}

What gets printed

What happens if:
- We cannot open “out.txt” to write to due to permission issues
  - We can open “out.txt” to write to

A) - exit gracefully, nothing printed
    - out.txt gets expected output
B) - program exits with run time error
    - program exits with run time error
C) - exit gracefully, nothing printed
    - exit gracefully, nothing printed
D) - Compiler error
E) - None of the above
Search, sort, hash, and big-O
7. Draw the UML diagram for the following class:

```java
public class Student {
    public String university = "UCSD";
    private String name;
    private int age;
    public Student() {
        this("Adam", "24");
    }
    public Student(String name, int age) {
        this.name = name;
        this.age = age;
    }
    public String getName() {
        return name;
    }
    private void birthday() {
        age++;
    }
}
```
Stacks, heaps, and static memory
9. Given the following code:

```java
import java.util.Scanner;

public class Midterm {
    final static int MAX GRADE = 100;
    public static void main(String[] args) {
        int[] page_scores = {0, 0, 0, 0, 0};
        Scanner scanner = new Scanner(System.in);
        int total = 0;
        for (int page = 0; page < page_scores.length; page++) {
            System.out.print("Enter score for page "+(page+1)+": ");
            page_scores[page] = scanner.nextInt();
            total += page_scores[page];
        }
        int percentage = (total/MAX GRADE)*100;
        System.out.println("Your percentage is: "+percentage+\%);
    }
}
```

In which part of memory are the **contents** of the following variables stored (static, stack, or heap)?

a) page_scores
b) page
c) total
d) MAX GRADE
e) scanner
Draw the order of methods on the stack up to the indicated point of program execution, method6. Note: the stack may not fill completely, also Stacks fill from the bottom to the top:

```java
public class Driver {
    public static void main(String[] args) {
        SomeClass sc = new SomeClass();
        sc.method4();
        sc.method1();
    }
}
public class SomeClass {
    public void method1() {
        method5();
    }
    public void method2() {
        return;
    }
    public void method3() {
        method6();
    }
    public void method4() {
        method2();
    }
    public void method5() {
        method3();
    }
    public void method6() {
        /* What is the stack here? */
    }
```
IDE & JUnit Testing

• `assertTrue(condition)`
• `assertEquals(expected, actual)`
• `assertArrayEquals(expected, actual)`
public class XXX extends Car
Polymorphism

```java
public class Person {
    public void identity2() {
        System.out.println("Person1");
    }
    public void identity3() {
        System.out.println("Person2");
    }
}

public class Professor extends Person {
    public void identity1() {
        System.out.println("Professor1");
    }
    public void identity2() {
        System.out.println("Professor2");
        identity1();
    }
}

public class Adam extends Professor {
    public void identity1() {
        System.out.println("Adam1");
    }
    public void identity2() {
        System.out.println("Adam2");
        super.identity2();
    }
    public void identity3() {
        System.out.println("Adam3");
    }
}
```

What is the output of the following code:

A. `Person jundt = new Adam();
jundt.identity1();`

B. `Person jundt = new Adam();
jundt.identity2();`
Abstract classes and Interfaces

How would you implement the following interface:

```java
public interface cse8b {
    public int getGrade(String ID);
}
```
JavaFX - GUIs

- **Stage**: Top level Java FX Component
- **Scene**: A container for all the items in the scene
- **Border, Hbox, Vbox, FlowPane, GridPane**: Helper classes that govern where components appear in the Scene

A very rough guide to creating a simple GUI:

1. Create a Layout Manager
2. Create and add components
3. Make a Scene
4. Set its properties

You don’t always have to do this in this order. Some of the steps can have substeps (e.g., creating a Layout Manager to organize other components)
public class BankAcct extends Application {
    private int balance = 100;

    public void start(Stage primaryStage) {
        Button deposit = new Button("Deposit $20");
        Button withdraw = new Button("Withdraw $20");
        Label label = new Label("Balance: ");
        TextField balanceField = new TextField("$" + balance);

        deposit.setOnAction(e -> {
            balance += 20;
            balanceField.setText("$" + balance);
            withdraw.setDisable(false);
        });

        withdraw.setOnAction(e -> {
            balance -= 20;
            balanceField.setText("$" + balance);
            if(balance <= 0) {
                withdraw.setDisable(true);
            }
        });

        // rest of code
    }

    // What happens if user hits : withdraw X 5, then deposit X 1

    A) Balance = 20, withdraw is disabled
    B) Balance = 0, withdraw is disabled
    C) Balance = 20, withdraw is enabled
    D) Balance = 0, withdraw is enabled
public class Question25 {
    public static void main(String[] args) {
        mystery(5);
    }

    public static int mystery(int x) {
        int y = x + 5;
        int z = x + 2;

        if (y > 4) {
            System.out.println(x + " " + y + " " + z);
            z = x + mystery(x - 3);
            System.out.println(x + " " + y + " " + z);
        } else {
            y = z - x;
            System.out.println("Stop!");
            System.out.println(x + " " + y + " " + z);
        }

        return y;
    }
}
Threads

```java
public class Sum extends Thread {
    int start, stop, sum;
    public Sum(int start, int stop) {
        //initialize vars.
    }
    public void run() {
        for(int i = start; i <= stop; i++) {
            sum += i;
        }
    }
}

public class Example {
    private static final int SUM_NUMBERS = 5;
    public static void main(String[] args) {
        Thread t1 = new Sum(0, SUM_NUMBERS);
        t1.start();
        System.out.println(((Sum) t1).sum); //note: this is ok!
    }
}
```
Goals for course

• Independence
• Mentorship
• Clarity
http://www.cape.ucsd.edu/