### Solutions

CS 141: Introduction to (Java) Programming: Exam 2

Jenny Orr • Willamette University • Fall 2013

<table>
<thead>
<tr>
<th></th>
<th>(max 12)</th>
<th></th>
<th>(max 18)</th>
<th></th>
<th>(max 32)</th>
<th></th>
<th>(max 23)</th>
<th></th>
<th>(max 15)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td></td>
<td>4.</td>
<td></td>
<td></td>
<td></td>
<td>5.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td></td>
<td></td>
<td>Total:</td>
<td></td>
<td></td>
<td></td>
<td>(max 100)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. (12 pts total) 2-Dimensional Arrays:
   a. (4 pts) Write code that declares and creates a 2-dimensional array of integers called `myNums` with 6 rows and 4 columns:

   ```java
   int[][] myNums = new int[6][4];
   ```

   b. (2 pts each, 8 pts total) What is the value of each of the following (or state if the item doesn’t make sense and, if so, why)

   ```java
   myNums[0][4]  ________________________________
   Makes no sense – invalid index - can’t have a column index of 4
   ```

   ```java
   myNums.length   __6____________________________
   ```

   ```java
   myNums[2].length  __4_____________________________
   ```

   ```java
   myNums[2][3].length  ________________________________
   Makes no sense – `myNums[2][3]` is an integer. It doesn’t have a length.
   ```
2. (1 pt each, 32 pts total) True and False: Please circle T or F

1) T or F: Object parameters are passed by value.

2) T or F: Integer (int) parameters are passed by value.

3) T or F: Arrays parameters are passed by reference.

4) T or F: The keyword word static is used to indicate instance methods and variables.

5) T or F: If one changes the value of a class variable, the value is changed for all objects of that type.

6) T or F: A binary file can easily be read by any text editor.

7) T or F: A try-catch is used to handle exceptions.

8) T or F: If a program tries to open a file that doesn’t exist, the program will throw an exception.

9) T or F: It is never ok for two methods in a class to have the same name.

10) T or F: An ascii file can only contain letters of the alphabet.

11) T or F: The catch part of a try-catch is used to indicate what to do if no errors are generated.

12) T or F: In general, instance member variables should be public.

13) T or F: The name of a constructor must be the name of the class.

14) T or F: A constructor should always have a void return value.

15) T or F: The process of hiding object data and providing methods for data access is called encapsulation.

16) T or F: An object’s accessor method is called when the keyword new is used.

17) T or F: An object’s member variable exists for as long as the object exists.

18) T or F: Once an object is garbage collected, it can still be retrieved if needed again.

19) T or F: It is possible for a method to have multiple return statements in its implementation.

20) T or F: Private methods can be called outside of the class by using setters and getters.

21) T or F: Private instance variables hide the implementation of a class from the class user.
22) **T or F:** The terms setters and accessors are used interchangeably.

23) **T or F:** A method with a **void** return type must never have a return statement.

24) **T or F:** A variable declared within a block of code can be accessed from outside of the block.

25) **T or F:** The `toString` method must always be declared as **public**.

26) **T or F:** The declaration:

   ```
   Card c;
   ```

   creates a new Card object.

27) **T or F:** Stepwise refinement is the process of breaking complex problems down into smaller, manageable steps.

28) **T or F:** Unit testing should always be done.

29) **T or F:** It is never ok for two different variables to have the same name in a class.

30) **T or F:** A stub is a method that acts as a placeholder and returns a simple value so another method can be tested.

31) **T or F:** Suppose `setValue` is a method with one parameter of type `int`. When **calling** the method, you need to **provide a formal parameter**, e.g. `setValue(int x)`. And when **declaring** the method, you need to provide an **actual parameter**, e.g., `setValue(5)`.

32) **T or F:** Methods can have multiple arguments and can return multiple return values.
3. (5 pts each, 15 pts total) **Object Diagram:** Assume there exists a `Die` class containing an instance member variable which stores the number of sides. The `Die`'s `toString` method prints the word “Die” followed by the number of sides, e.g. “Die 6”.

Given the code below, **draw the object diagram at the lines 3, 7, and 10. Also indicate at each of these lines, what is printed and what, if anything, is garbage collected.**

Follow the drawing style used in class, e.g. use rectangular boxes to indicate object references; use rounded boxes to indicate objects as shown below on the right.

```java
public static void main(String[] args) {
    Die[] d = new Die[2];
    Line 3: System.out.println(d[0] + ", " + d[1]);
    Line 4: Die dd = new Die(10);
    Line 5: d[0] = new Die(6);
    Line 6: d[1] = dd;
    Line 7: System.out.println(d[0] + ", " + d[1] + ", " + dd);
    Line 8: d[0] = null;
    Line 9: dd = null;
    Line 10: System.out.println(d[0] + ", " + d[1] + ", " + dd);
    Line 11: }
```

Please use the space below as scratch paper. Once you have worked out the diagrams, please copy them as neatly as possible to the next page.
**Line 3:** output is null, null

Object diagram:

What if anything is garbage collected? None

---

**Line 7:** output is Die 6, Die 10, Die 10

Object diagram:

What if anything is garbage collected? none

---

**Line 10:** output is null, Die 10, null

Object diagram:

What if anything is garbage collected? Die 6
4. (18 pts) Create a Person class consisting of the following:
- Two instance member variables for the person’s name and age.
- A constructor which sets the value of both instance variables.
- A setter & getter for the age variable.
- A toString method.
- A method called birthday which increases age by 1 and returns the message “Happy Birthday”.

```java
public class Person {

    // Instance member variables for name and age:
    private String name = "";
    private int age = 0;

    // Constructor:
    public Person(String n, int a) {
        name = n;
        age = a;
    }

    // Getter and Setter for age
    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    // toString
    public String toString() {
        return name + " is " + age + " years old."
    }

    // birthday
    public String birthday() {
        age = age + 1;
        return "Happy Birthday";
    }
}
```
5. (23 pts) For each variable name listed at top, indicate a "D" on the line where the variable is declared. Mark an x in boxes to indicate scope. Circle "P" at the top if the variable is formal parameter, "M" if it is a member variable and "L" if it is a local variable.

```java
public class DoSomething {

    public static void main(String[] args) {
        int n = 3;
        double b = 5.2;
        result = bigInt + bigTwo(b);
        System.out.println(result);

        if (y > 5) {
            int x = 4;
            int i = y;
            p = y + i;
        } else {
            return p;
        }

        for (int n = 1; n < t; n++) {
            c = c + n;
        }

        return bigTwo(c);
    }

    public static double bigInt(int t) {
        double c = 1.0;
        return bigTwo(c);
    }
}
```