Java for Non Majors

Final Study Guide

April 26, 2017

The test consists of

1. Multiple choice questions
2. Given code, find the output
3. Code writing questions
4. Code debugging question
5. Short answer questions

- You will have an opportunity to earn 20 extra credit points.
- Please try and attempt all questions. You get points for trying.
- The test is cumulative, but focuses on topics introduced before the second midterm.
- Anything from the slides / homeworks / quizzes is fair game.
- Code debugging is mostly syntax based (missing brackets, etc.)

Topics to study

- Basics of Java Programming. You will be expected to be familiar with Java Syntax, the basic components of a Java program, compiling and running code and the tools involved, etc. Questions will draw from (but not be limited to) the following topics:
  - Java reserved words.
  - Data types, variables and type conversions.
  - Operators and operator precedence.
- Console I/O. Basic input output
  - Printing using the System class methods - println, print and printf.
  - Reading in data from the console using the Scanner class.
- Selection and Repetition.
  - Regular sequential execution of code.
  - Logical operators and short circuit evaluations.
  - Selection statement - if, if-else, if-else ladders, switch structure (including case labels, use of break and continue)

  - Loops - while, do-while and for loops, enhanced for loops, choosing loops most suited for a particular task.

- Java Methods.
  - Method syntax - declaring and defining methods.
  - Modifiers, return types and parameter lists.
  - Pass by value and pass by reference.
  - Method overloading.
- Strings
  - String comparison.
  - Using methods of the String class
- Arrays
  - Declaring, initializing and using arrays of primitive types.
  - Passing arrays as parameters to methods and returning arrays from methods.
  - Multi-dimensional arrays.
- Classes and Objects
  - Creating an object of a pre-existing class and using the available methods.
  - Access Modifiers - public, private and protected.
  - Defining a class - data attributes, constructors, accessor and mutator methods, instance methods.
  - Instantiating a class (creating an object), and using instance methods.
  - Difference between static and instance methods.
  - Passing objects as parameters to methods and returning objects from methods.
  - The ”this” keyword.
  - Arrays of objects.
- Inheritance, Interfaces and Polymorphism
  - Concept of base (super) and derived (sub) classes.
  - The “super()” keyword.
  - Method overriding.
  - The class ”Object”
  - Abstract Classes
  - The concept of dynamic binding.
  - Casting classes
  - Interfaces
- Java Exceptions
  - Types of exceptions.
  - Claiming, throwing and catching exceptions.
- GUI programming and Graphics
  - Basic Graphics Classes in Java (Swing and AWT).
  - The paint() methods, and the Graphics, Color and Polygon classes.

Sample Questions

1. General Instructions

- The multiple choice questions and short answer questions will pretty much be like the questions on the midterms.
- The code debugging question will also follow the same pattern. You only need to point out the syntax and semantic errors. You don't need to fix typos in String literals, or attempt to fix the logic. Basically, you only need to fix the stuff that the compiler will complain about.
- Some samples for the code writing questions are included in the study guide. Please note that the sample questions are of the same difficulty level as the questions on the test. That is the only similarity between the questions here and those on the test. You will be expected to solve a different problem on the test.
- The Sample Runs provided here and on the test are *samples*. They are just a single run of the program. Your solution should work for all legal inputs. Please do not hard code your solution (think that the sample is the only possible input) to the sample run. The sample is just provided to clarify the requirement. For example, if the sample run says N=4, it does not mean N is 4 all the time. It just happens to be 4 this once. It can be any integer.

2.Examples

1. Which of the following is NOT a Java reserved word?
   (a) public
   (b) static
   (c) void
   (d) main

2. Evaluate the following expression.

   ```
   x = 5, y = 15, z = 3 (All integers)

   x * 4 + y / 2 % z
   ```

   (a) 21
   (b) 2
   (c) 2.5
   (d) None of the above

3. A Java variable can begin with
   (a) A letter
   (b) underscore
   (c) '$' sign
   (d) All of the above

4. Which of the following is NOT a Java access specifier?
   (a) public
   (b) private
   (c) abstract
   (d) protected

5. The class "Coffee" has a single data attribute "price". Which of the following methods can be private?
   (a) void calcTax()
   (b) Coffee(double p)
   (c) void setPrice (double p)
   (d) double getPrice()

6. A variable declared inside a block is available
   (a) throughout the program
   (b) only in that block
   (c) only in that class
   (d) only in that method

7. An object consists of
   (a) Name
   (b) Attributes
   (c) Methods
   (d) All of the above

8. A class is a single instance of an object
   (a) True
   (b) False

9. "IndexOutOfBoundsException" is a
   (a) Normal Exceptions (checked exceptions)
   (b) Runtime Exceptions (unchecked exceptions)
   (c) Error

10. "FileNotFoundException" is a
   (a) Normal Exceptions (checked exceptions)
   (b) Runtime Exceptions (unchecked exceptions)
   (c) Error

11.
```
Animal s1, s2, s3; // Animal is the base class
s2 = new Bird();
Bird c = new Bird(); // Bird is a derived class
```
   Please choose all correct options below.

(a) `s1 = c;`
(b) `s1 = (Animal)c;`
(c) `s3 = new Bird();`
(d) `c = s2;`
(e) `c = (Bird)s2;`

12. Which of the following is not a container?
    (a) JFrame
    (b) JPanel
    (c) JDialog
    (d) JMenu
13. How do you do exception handling?
14. What is the Helper classes? Give 3 examples for them
15. What are the Simple Dialog Box types? Explain each of them with a few sentence

3. Code Writing: Loops

Write a Java method called `armstrong` to print all the Armstrong Numbers in a given range. This method accepts 2 integer parameters - `start` and `end` and returns nothing.

An Armstrong Number is a 3 digit number where the sum of the cubes of the individual digits of the number equals the number itself. For example, 371 is an Armstrong Number, since $3^3 + 7^3 + 1^3 = 27 + 343 + 1 = 371$.

Sample Run: (start = 100, end = 999)

```
The Armstrong Numbers between 100 and 999 are:
153
370
371
407
```

Solution: Please note that only the method is required. You need not write the entire program.

4. Code Writing: String Manipulation

Write a Java method called `insert3` that reads a series of strings from the user and prints them after inserting the string "blue" at the third index of the given string. The method accepts a single integer parameter `N` that denotes the number of strings. The function returns nothing. The strings entered by the user will always be at least 4 characters long. You need to test for that.

Sample Run: (N = 4)

```
Enter the 4 strings:
Hello World
Helbluelo World
To infinity and beyond
To blueinfinity and beyond
I Wumbo You Wumbo
I Wblueumbo You Wumbo
Java Programming
Javbluea Programming
```

Solution: Once again, we only need to write the method, not the entire class. You can assume that all the necessary libraries have been imported. You can make this assumption of the test as well.

5. Code Writing: Inheritance + Array of Objects

Write a Java program with the following requirements.

- Write a class called Drink. The class has an integer attribute called size that holds the size of the drink in ounces. The class also has a parametrized constructor and a method called `print` that prints the size.
- Write a class called `Coffee` that inherits from `Drink`. The class has an integer attribute called numShots that denotes number of espresso shots in the drink. The class also has a parametrized constructor and a method called `print` that overrides the base class method to print both the size and the number of shots.
- Write a class called `Beer` that inherits from `Drink`. The class has a double attribute called percentAlcohol that denotes percentage of alcohol content in the drink. The class also has a parametrized constructor and a method called `print` that overrides the base class method to print both the size and the percentage of alcohol.
- Write a class called `ManyDrinks`. This call just contains the `main` method. In the main method, create an array of Drink of size 4. The first 2 elements are of type Coffee, and the last 2 are Beer. Give the drinks values of your choice. Then invoke the `print` method for each of them.