**CS211 Spring 2011**

Dr. Kinga Dobolyi

**Midterm Examination**

Student Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student G#: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student signature for Honor Code:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Part 1: Short Answer.

1. a. Imagine you are responsible for designing a new computer that must store information. Your boss has said to you that they have invented a new type of hardware/memory that can store up to 5 different levels: rather than simply being “on” or “off”, it could have 6 different discernible states. Your boss wants you to use as little memory space, in terms of digits, as possible to store information. Given the option between a base 4 number system for all memory storage, or a base 2 (binary) number storage system, which would better satisfy these and these requirements alone, and ***why***? (2 pts)

b. Given 5 digits, what is the largest unsigned integer your number system could represent? (2 pts)

1. What is the difference between a class and an object in Java? If you use an analogy, it should be PART of your explanation. (2 pts)
2. What is the result of the following expressions? Give the Java type and the value of the expression, or say it would result in a compilation error. Assume that all variables have been defined and assigned values, and all methods have their default **Object** implementation.

(14 pts)

* 1. **12 / 5.0** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
  2. **“happy” + “3”** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
  3. **“happy” + 3** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
  4. **p1.equals(p1)** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
  5. **p1 == p1** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
  6. **int p = 3; p++;** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
  7. **p1.toString().length()**\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Regular expressions
   1. Write a regular expression, as we use them in Java (in the form of a **String** that you could pass to **Scanner**’s **findInLine** method) that matches any line where an address is contained somewhere in the line, but the line must end in an integer (for this question it is okay if the integer has preceeding zeros such as 007). Your regular expression should only match and address if the following are true: (10 pts)
      1. It starts with a first name made up of word characters, followed by a single whitespace, followed by a last name made up of word characters
      2. After part (i), there can be any number of whitespace characters (must have at least one)
      3. After part (ii), there will be a street number (again, can have preceeding zeros) of three digits, followed by a whitespace character, followed by at least one and up to any number of non-whitespace characters.
      4. After part (iii) there will be at least one, and possibly more, whitespace characters, followed by the integer the line must end in
   2. Write YES or NO below if the regular expression, as we use them in Java (in the form of a **String** that you could pass to **Scanner**’s **findInLine** method) would match: (10 pts)

**“(**[**\\w\*)12**](file:///\\w*)12)**.4+\\$$”**

* + 1. **Hello1234** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
    2. **111234$** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
    3. **Hello 1234$** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
    4. **Hello123$**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
    5. **12.4$**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Provide the output of the following piece of code (assume it compiles): (6 pts)

**public class Exam1{**

**public int intVal;**

**private Address address;**

**public static void main(String[] args){**

**Exam1 exam = new Exam1(5, “Cherry Lane”);**

**Exam1 exam2 = update(exam, exam.getIntVal(), exam.getAddress());**

**System.out.println(exam);**

**System.out.println(exam2);**

**}**

**public Exam1(int intValIn; String addressIn){**

**System.out.println(“ctor 1”);**

**intVal = intValIn;**

**address = new Address(addressIn);**

**}**

**public Exam1(int intValIn; Address addressIn){**

**System.out.println(“ctor 2”);**

**intVal = intValIn;**

**address = addressIn;**

**}**

**public String toString(){**

**System.out.println(“intVal: “ + intVal);**

**//prints out the entire String that was**

**//passed to the Address constructor**

**System.out.println(“Address: “ + address);**

**}**

**public static Exam1 update(Exam ex, int intValArg, Address addressArg){**

**intValArg = 3;**

**addressArg = new Address(“Pine Tree Lane”);**

**ex = new Exam(intValArg, addressArg);**

**return ex;**

**}**

**//assume standard getters and setters as seen in class**

**}**

1. A. Write an ***interface*** (that cannot be instantiated to an object) called **Printable** that contains a method that is of return type void, takes a single **java.lang.Object** argument, and would be used to print out the object passed in using its default **String** representation. After printing out the object passed in, the method would print out the default string representations of all the private (and private only) attributes of the class that implements the interface. (8 pts)

B. Now implement the **Printable** interface with a public class called **Computer** that has two fields: a public **String operatingSystem** and a private primitive integer **bits** that stores whether the processor operates on 32 bits or 64 bits. In addition to implementing the interface, your class must also provide a comprehensive constructor, getter and setters for all fields, and implement the **toString** method. (14 pts)

(use this space for 6B if you need to)

Part 2: True/False (circle one) (7 pts)

1. If I have the following statements in my code,

**int x = 0;**

**return y / x;**

The Java compiler will report **a DivideByZero Exception** because it knows that

the value of x is zero in the return statement. 1.TRUE FALSE

1. Public mutable attributes of classes violate the object-oriented principle of information hiding 2.TRUE FALSE
2. An object in memory can have, at most, two aliases 3.TRUE FALSE
3. A switch statement can only have integers or characters as cases 4.TRUE FALSE
4. I can leave off an import statement for **java.util.\*** if I always refer to **ArrayList**s in my code as **java.util.ArrayList** instead of just **ArrayList** 5.TRUE FALSE
5. Code that is compiled is usually slower than code that is interpreted, which is why a lot of languages prefer interpreters to compilers 6.TRUE FALSE
6. I always write my name on exams 7.TRUE FALSE