Sample problems midterm 1, CS:2230

Prof. Mauricio Monsalve

September 24, 2015

Exams will contain a variety of problems, including *some* of the following:

- Choosing alternatives
- Naming, defining, and contrasting concepts
- Providing examples and counterexamples
- True/false, with justification
- Drawing and interpreting diagrams
- Modeling real world problems (abstraction, polymorphism)
- Writing code
- Reading and interpreting code

The difficulty of problems will range from easy to challenging. Midterm 1 will cover all the material seen until one week before the testing time.

There will be several forms for each exam. This is to make cheating difficult.

Notes and books are allowed during the exam. Talking to your peers is not allowed. The use of computers is not allowed either.

1 Short problems

1.1 Contrasting concepts

Indicate similarities and differences between the following concepts. The space provided is not representative of the space provided in the exam–more space will be available.

1. Modularity v. Encapsulation.
   Similarity: _____
   Difference: _____

2. Abstract Class v. Interface.
   Similarity: _____
   Difference: _____

3. Array v. Linked List.
   Similarity: _____
   Difference: _____

4. Object v. Type.
   Similarity: _____
   Difference: _____

5. Package v. Class.
   Similarity: _____
   Difference: _____

6. Private v. Protected.
   Similarity: _____
   Difference: _____

7. Object Oriented Programming v. Functional Programming.
   Similarity: _____
   Difference: _____

8. Static methods v. Non-static methods (aka Dynamic).
   Similarity: _____
   Difference: _____

1.2 Defining concepts

1. Error tolerance: _____
2. State of an object: _____
3. Adaptability: _____
4. Multiple subtyping; _____
5. Class diagram: _____
6. JVM: _____

2 Code problems

2.1 Counterexamples

Provide code counterexamples for the following statements. Explain how the code refutes the statement.

1. Polymorphism fully covers abstraction.
2. We could do perfectly well without using *super*.
3. You do not lose anything by using only *implements* for subtyping.
4. A subclass cannot subtype two concrete superclasses.
5. The *null* value is absolutely irreplaceable.

2.2 General coding exercises

General coding exercises are similar to those of homeworks. The ones below are not exactly the simplest, but they are a good exercise. In the exams, you won't face problems this long, but the difficulty could be similar (depending on the rest of the problems).

1. Implement the following static method that returns a verbose version of integer numbers:
   `public static String textInt(int x)`
   The method should describe hundreds, thousands,and negative numbers. Be mindful of numbers such like 10, 11, ..., 19. For the following cases, the method should return:
   (a) `textInt( 0 )` $\longrightarrow$ `"zero"`
   (b) `textInt( 25007 )` $\longrightarrow$ `"twenty five thousand seven"`
   (c) `textInt( -1054 )` $\longrightarrow$ `"negative one thousand fifty four"`
   (d) `textInt( 812 )` $\longrightarrow$ `"eight hundred twelve"`
2. Implement the following static method that returns the divisors of a number:
   `public static int[] divisors(int x)`
   The method should return an array with all the positive integer divisors of argument x. Note that a d is a divisor of x if $x\%d$ equals 0. Also note that $1 \leq d \leq x$. The method should return an empty array for when $x < 1$. For the following cases, the method should return:
   (a) `divisors( 20 )` $\longrightarrow$ `{1, 2, 4, 5, 10, 20}`

(b) `textInt( 31 )` $\longrightarrow$ `{1, 31}`

(c) `textInt( -1054 )` $\longrightarrow$ `{}`

(d) `textInt( 1 )` $\longrightarrow$ `{1}`

3. Implement the following immutable complex numbers class:

```
class Complex{
public Complex(double real, double imag)
public Complex() // sets the number to 0 + 0i
public Complex add(Complex x)
public Complex sub(Complex x)
public Complex mult(Complex x)
public double norm()
}
```

A complex number consists of two parts: a real part, and an imaginary part. For complex number $a + bi$, number a is the real part, b is the imaginary part, and $i = \sqrt{-1}$ is the imaginary unit. For example, $10 - 5i$, 20, and $27.5i$ are complex numbers. The operations of complex numbers are:

(a) Addition: $(u + vi) + (x + yi) = (u + x) + (v + y)i$

(b) Subtraction: $(u + vi) - (x + yi) = (u - x) + (v - y)i$

(c) Multiplication: $(u + vi) \cdot (x + yi) = (ux - vy) + (uy + xv)i$

(d) Norm: $|u + vi| = \sqrt{u^2 + v^2}$

3 Modeling problems

3.1 Comic strip representation

Consider the problem of representing the parts of a newspaper-style comic strip. You can see an example of one[1] below.

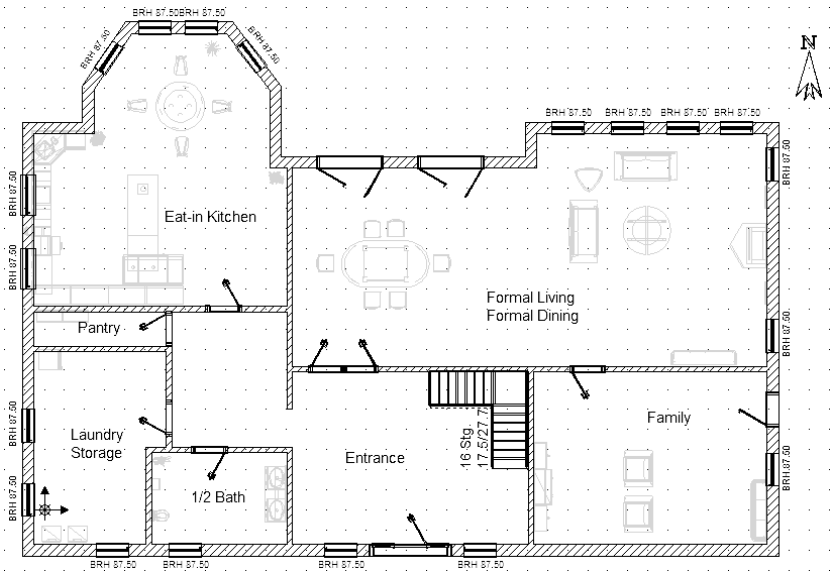
[1] Author: *Ring Lardner*. Source: *The Washington times. (Washington [D.C.]), 21 Oct. 1922. Chronicling America: Historic American Newspapers. Lib. of Congress.* Public domain.

Besides from the drawings themselves, a cartoon strip is composed of several components and identification data (e.g., title, author, date, etc.).

Design an object oriented description of comic strips through a class diagram sketch. As stated, the drawings themselves might be of type Image, but encode as much as possible as classes/objects.

3.2 Floor plan representation

Consider the problem of representing the parts of an architectural floor plan. You can see an example of one[2] below.



More than in the exact geometric details of the floor plan, we are interested in the components of a floor plan. For example, the rooms, which rooms are adjacent (and connected), the furniture, the windows (sizes and other characteristics), doors, stairs, etc.

Design an object oriented description of floor plans through a class diagram sketch.

[2] Author: *Boereck*. Source: *Wikimedia Commons*. Public domain.