# CSE 331 Final Exam 12/10/12

Name _____

There are 12 questions worth a total of 100 points. Please budget your time so you get to all of the questions. Keep your answers brief and to the point.

The exam is closed book, closed notes, closed electronics, closed telepathy, etc.

You may remove pages during the exam while you're working, but you must return all pages at the end.

Many of the questions have short solutions, even if the question is somewhat long. Don't be alarmed.

If you don't remember the exact syntax of some command or the format of a command's output, make the best attempt you can. We will make allowances when grading.

Relax, you are here to learn.

Please wait to turn the page until everyone is told to begin.

Score _____ / 100

1. _____ / 16
2. _____ / 6
3. _____ / 5
4. _____ / 9
5. _____ / 6
6. _____ / 6
7. _____ / 6
8. _____ / 10
9. _____ / 8
10. _____ / 6
11. _____ / 10
12. _____ / 12

**Question 1.** (16 points) A design exercise. One of the key data structures in Dijkstra's algorithm is the *priority queue*. If the Java library did not provide this ADT we could specify and implement it ourselves.

A *priority queue* is a data structure that provides the following two operations:

- *insert* – add a new element to the priority queue.
- *deleteMin* – find, return, and remove the minimum item in the priority queue.

For this problem, assume that we are only interested in implementing priority queues that hold integers (Java `int`s). Also, for simplicity, we will assume that a priority queue has a maximum size and the elements are stored in a sorted array of `int`s, and not a fancier data structure like a binary heap.

(a) Describe the *representation* (rep) for the priority queue, including, of course, the sorted array containing the queue elements, and any other variables that are part of the rep. Don't give the representation invariant here – that's next.

(b) What is the *representation invariant* (RI) for your priority queue, given the representation you've chosen in part (a)?

**Question 1. (cont.)** (c) What is the *abstraction function* (AF) for your priority queue, given the representation and RI from parts (a) and (b)?

(d) Give the Java code for a suitable `checkRep()` function for your priority queue. If it matters, assume it should be suitable for use during testing and debugging.

**Question 2.** (6 points) Representation exposure. We want to create a new ADT (class) to represent an interval of time. The beginning and end of an interval will be represented by Java `Date` objects, which store a `long` integer value representing the number of milliseconds since the beginning of time (which Java defines as 00:00:00 GMT on Jan. 1, 1970). Class `Date` includes a constructor with a `long` integer parameter to specify the initial time stored in a new `Date` object, and includes other methods to set this time, `setTime(long n)`, and to read it, `long getTime()`.

Here is the code for our `Interval` class. In the code below:

(a) Indicate where there are potential representation exposure problems, if any, by pointing out places in the original code that are problematic and describing how these could lead to the exposure, and

(b) if there are representation exposure problems, fix the code to prevent them without altering the interface to class `Interval`.

```
public class Interval {

     private final Date start;

     private final Date end;


     public Interval(Date start, Date end) {

          this.start = start;

          this.end = end;

     }

     public Date getStart() {

          return this.start;

     }

     public Date getEnd() {

          return this.end;

     }
}
```

**Question 3.** (5 points) Specifications. The Ghost of Midterms Past.

On the midterm exam we had a question about the relative strength of different specifications for the function `double sqrt(double x)`, a method that returns the square root of its argument. The four specifications were:

A
@requires x >= 0
@return y such that |y*y - x| < 0.001

B
@requires x >= 0
@return y such that |y*y - x| < 0.001
@throws IllegalArgumentException if x < 0

C
@return y such that |y*y - x| < 0.001 if x >= 0, and 0.0 if x < 0

D
@requires x >= 0
@return y such that |y*y - x| < 0.00001

We had to throw out this question because at least one of the specifications had technical errors that made it invalid. Which specification or specifications above are invalid and why? (Keep it short and to the point, and a technical reason is much better than general hand-waving.)

**Question 4.** (9 points) Hashcodes. It's almost time for holiday gift giving, and we've been working on an application to handle inventory for a toy store. We've created a class to represent individual toys that the store has in stock.

```
public class Toy {
  private String name;         // Toy name, example "Furby"
  private String description;  // Description, example "annoying"
  private int price;           // price in pennies
  private int sku;             // stock (inventory) number
```

We need to provide `equals` and `hashCode` methods for this class and are considering these possible implementations.

```
  public boolean equals(Object other) {  // equals version 1
    if (!(other instanceof Toy)) return false;
    Toy t = (Toy) other;
    return this.sku == t.sku;
  }

  public boolean equals(Object other) {  // equals version 2
    if (!(other instanceof Toy)) return false;
    Toy t = (Toy) other;
    return this.name.equals(t.name) && this.sku == t.sku;
  }

  public boolean equals(Object other) {  // equals version 3
    if (!(other instanceof Toy)) return false;
    Toy t = (Toy) other;
    return this.name.equals(t.name) &&
                this.description.equals(t.description);
  }

  public int hashCode() {          // hashcode version 1
    return sku;
  }

  public int hashCode() {          // hashcode version 2
    return name.hashCode()+sku;
  }

  public int hashCode() {          // hashcode version 3
    return name.hashCode();
  }
```

(Continued next page. You may remove this page while you work on the question if you wish.)

**Question 4.** (cont.) Complete the following table by putting an X in each space where the corresponding `hashCode` and `equals` methods given on the previous page are compatible – that is, put an X in the space if the given combination of `hashCode` and `equals` methods would work correctly if they were used in class `Toy`.

| | `equals` version 1 | `equals` version 2 | `equals` version 3 |
|---|---|---|---|
| `hashCode` version 1 | | | |
| `hashCode` version 2 | | | |
| `hashCode` version 3 | | | |

**Question 5.** (6 points) Exception chaining. Sometimes we catch exceptions, only to immediately throw a different exception. For example:

```
void doSomething() throws SomeOtherException {
   try {
      code that might cause an IOException
   } catch (IOException e) {
      throw new SomeOtherException(e);
   }
}
```

Why is this sometimes a useful design strategy and when should it be used? (Be brief)

**Question 6.** (6 points) Some typing. We've found some really old code that copies lists. The existing code is a static method that has the following method signature:

```
static void copyElements(List src, List dst);
```

(a) Give a new version of this signature by adding generic type parameter(s) so the method will type-check and accept only `List` parameters that have the same element types. In other words, the new signature should accept the same type for both the `src` and `dst` arguments, say either both `List<Number>` or both `List<Integer>`, but not `List<Integer>` for one and `List<Number>` for the other.

(b) Give a better version of this new signature using generic type parameters (with bounded wildcards – extends/super) so that the method will type-check and can accept parameters where the source list (`src`) elements can be stored in the destination list (`dst`), even if the source list elements are a subtype of the destination list elements.

**Question 7.** (6 points) The subtyping rules for generic types sometimes seem surprising, at least at first. For example, `List<Integer>` is not a subtype of `List<Number>` even though `Integer` is a subtype of `Number`.

Supposed we changed the type system to specify that generic types are covariant with respect to their arguments. In other words, in this new system, since `Integer` is a subtype of `Number`, we would have that `List<Integer>` is a subtype of `List<Number>`.

Give a sequence of Java statements that would be legal using this new typing rule, but would nevertheless produce incorrect behavior if executed. (In other words, show that this new typing rule is wrong by giving an example of an erroneous sequence of statements that would be accepted by the new rule.)

**Question 8.** (10 points) More generic questions. Suppose we have the following classes defined:

```
class Shape { … }
class Circle extends Shape { … }
class Rectangle extends Shape { … }
```

Now suppose we have a program that contains the following objects and list:

```
Object o;
Shape s;
Circle c;
Rectangle r;
List<? extends Shape> les;
```

For each of the following, circle Error if there is a type error in the assignment. Circle OK if the assignment has the correct types and will compile without errors.

OK Error `les.add(s);`

OK Error `les.add(c);`

OK Error `c = les.get(0);`

OK Error `s = les.get(0);`

OK Error `o = les.get(0);`

Now, suppose we have another list

```
List<? super Shape> lss;
```

As before, circle Error if there is a type error in each of the following assignments. Circle OK if it is correct.

OK Error `lss.add(s);`

OK Error `lss.add(c);`

OK Error `c = lss.get(0);`

OK Error `s = lss.get(0);`

OK Error `o = lss.get(0);`

**Question 9.** (8 points) There are methods in this madness. Suppose we have the following classes and methods:

```
class A {
  void f()      { System.out.println("A.f()"); }
  void f(int n) { System.out.println("A.f(int)"); }
  void g(int n) { System.out.println("A.g(int)"); }
}

class B extends A {
  void f() { System.out.println("B.f()"); }
  void h() { System.out.println("B.h()"); }
}

class C extends B {
  void g(int n)   { System.out.println("C.g(int)"); }
  void g(char c)  { System.out.println("C.g(char)"); }
  void h(int n)   { System.out.println("C.h(int)"); }
}
```

For each of the following groups of statements, indicate the output produced or, if there is a compile-time or runtime type error, explain in a sentence what is wrong.

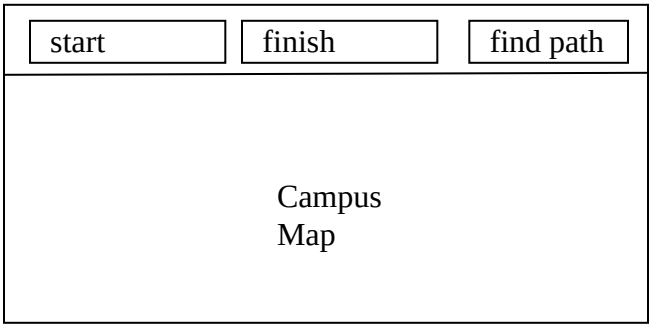
(i)
```
A a = new C();
a.f();
```

(ii)
```
A a = new C();
a.g(10);
```

(iii)
```
A a = new C();
a.g('x');
```

(iv)
```
B b = new B();
C c = b;
c.h(17);
```

**Question 10.** (6 points) Debugging strategies. After lots of work you almost have a finished graphical user interface for the campus maps program. The GUI code is built on top of the working model and path-finding code from previous projects. But it doesn't quite work. When you run the program, you see the user interface you expect:




There's a pair of drop-down menus, *start* and *finish*, with the building names in them, there's a *find path* button that is supposed to compute and display the path, and below the controls is the picture of the campus map.

Unfortunately when you select start and finish building names in the menus and click the "find path" button, nothing seems to happen – no path is drawn and nothing apparently changes.

Describe your strategy for figuring out where the problem is. What we're looking for here is a terse description of how you go about diagnosing the problem quickly and with a minimal amount of wasted effort. (And with a minimum amount of excess writing in your explanation. ☺)

**Question 11.** (10 points) GUI things. After your great success in 331 one of your colleagues has come to you for help with a small Java graphics program. The code is given on the next page. It is supposed to draw a green rectangle, then, each time the mouse is clicked, draw a red circle at the location of the click. The green rectangle appears, but for some reason the circles do not. We've verified that the `mouseClicked` method is being called correctly each time the mouse is clicked, and the `getX()` and `getY()` methods return the correct coordinates after each mouse click. We've also verified that the various graphics commands like `setColor` and `fillRect` have the proper parameters. The problem is with the program logic.

What is (are) the problem(s) and how should it (they) be fixed? You should give a brief description of the problem(s) below. Then you can either describe the fixes at the bottom of this page, or write code corrections on the next page. In either case, your description of the fix(es) must include specific code and be easy to follow (i.e., a novice programmer should be able to understand exactly how to repair the code by reading your corrections).

Description of the problem(s):

Changes needed to fix the problem(s): (either here or on the code on the next page)

**Question 11. (cont.)** The buggy code. Give a brief summary of the problem on the previous page, Then show how to fix it, either by writing on the code below or by writing specific details on the previous page.

```
public class Click {
  /** Drawing panel for green background with red dots */
  private static class Drawing extends JPanel {
    private Graphics gsave;     // graphics context for drawing

    public Drawing() {
      this.addMouseListener(new MouseAdapter() {
        public void mouseClicked(MouseEvent e) {
          gsave.setColor(Color.red);
          gsave.fillOval(e.getX()-10, e.getY()-10, 20, 20);
        }
      });
    }

    public void paintComponent(Graphics g) {
      g.setColor(Color.green);
      g.fillRect(0,0,300,300);
      gsave = g;
    }
  }  // end of Drawing nested class


  public static void main(String[] args) {
    JFrame frame = new JFrame("Exam Question");
    Drawing d = new Drawing();
    d.setPreferredSize(new Dimension(300,300));
    frame.add(d);
    frame.pack();
    frame.setVisible(true);
  }
}
```

**Question 12.** (12 points) Design patterns. We discussed several patterns during the quarter. For reference, here is a list of some of them:

- Creational: Factory, Singleton, Builder, Prototype, Interning, Flyweight
- Structural: Adapter, Composite, Decorator, Flyweight, Proxy
- Behavioral: Interpreter, Observer, Iterator, Strategy, Model-View-Controller, Visitor

For each of the following situations, write down the name of the design pattern that would be an appropriate solution to the given design problem. You do not need to justify your answer – just pick the best one. If more than one pattern seems appropriate, pick the best match or most specific one.

a) You have a set of objects that is unlikely to change, but the set of operations on them is likely to change. You want to keep the code for each operation together in a single module.

b) You have a class with a many optional parameters, and want to avoid creating a large number of constructors.

c) You have a Point class whose interface uses polar coordinates and you want to provide a wrapper so it can be used to implement a version of Point that uses rectangular coordinates.

d) You want to wrap a library class that manages data connections so that it logs connection attempts as they occur.

e) You want to save memory costs of creating many duplicate objects of a class which has only a few distinct abstract values.

f) When new objects of your class are created the new objects should actually have a more appropriate subtype.

*Have a great winter break! See you in the New Year!!*