0. Name (2 pts):

CS 2334: Programming Structures and Abstractions

Final Exam

Monday, December 13, 2010

General instructions:

- This examination booklet has 15 pages.
- Do not forget to write your name at the top of the page and to sign your name below.
- The exam is open book and notes, but closed electronic device.
- The exam is worth a total of 200 points (and 20% of your final grade).
- Explain your answers clearly and concisely. Do not write long essays (even if there is a lot of open space on the page). A question worth 5 points is only worth an answer that is at most one sentence.
- You have 120 minutes to complete the exam. Be a smart test taker: if you get stuck on one problem go on to the next. Don't waste your time giving details that the question does not request. Points will be taken off for answers containing excessive or extraneous information.
- Show your work. Partial credit is possible, but only if you show intermediate steps.

Problem	Topic	Max	Grade
0	Name	2	
1	Recursion	40	
2	Ethics	25	
3	Exceptions and Assertions	30	
4	Inheritance and Polymorphism	30	
5	Generic Programming and Generics	15	
6	Graphical User Interfaces and Event-Driven Programming	40	
7	Collections Framework	20	
Total		200	

On my honor, I affirm that I have neither given nor received inappropriate aid in the completion of this exam.

Signature: ________________________________

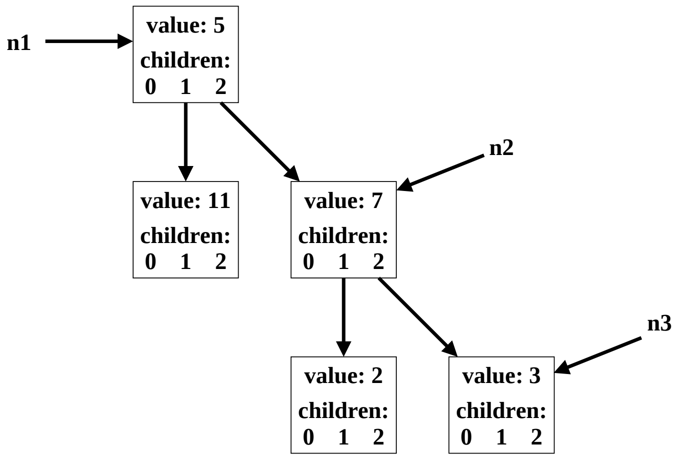
Date: ________________________________

1. **Recursion** (40 pts)

Consider the following class definition:

```
public class Node {
  private int value;
  private Node[] children = new Node[3];

  public Node(int value) {
    this.value = value;
  };

  public void addChild(Node c, int i) {
    if(i >= 0 && i < children.length){
      children[i] = c;
    };
  };

  public int baz() {
    int current = 0;
    boolean flag = false;

    for(int i = 0; i < children.length; ++i) {
      if(children[i] != null) {
        if(flag) {
          current = Math.min(current, children[i].baz());
        }else{
          current = children[i].baz();
          flag = true;
        }
      }
    }
    return(current + value);
  };
```

Assume that the following tree has already been constructed using the constructor and the **addChild()** methods, and that **n1, n2,** and **n3** are variables containing references to the indicated nodes. You may also assume that when arrows are not shown for children, their value is **null**.



(a) (5 pts) What is the value returned by **n3.baz()**?

(b) (10 pts) What is the value returned by **n2.baz()** ?

(c) (10 pts) What is the value returned by **n1.baz()**?

(d) (5 pts) **baz()** is a recursive method, what is(are) the base case(s)?

(e) (5 pts) What is(are) the recursive case(s)?

(f) (5 pts) In one sentence: what is the meaning of the integer that **baz()** returns?

2. **Ethics in Computer Science** (25 pts)

(a) (15 pts) *Build-A-Bear Workshop* is a chain of stores located in many malls. Customers move through a series of stations in which they select a "skin" for a stuffed animal (called a *friend*), have a heart inserted, have the stuffing inserted, select clothing, and finally create a "birth certificate" for the friend. Creating the certificate involves sitting down at a computer terminal clearly designed for young children: the keyboard and screen are at a very low height and have a "cute" aesthetic. At this interface, the customer enters a variety of information, including: friend name, and customer name, birth date, home address and email address. Briefly describe **three** ethical principles/ideas that *should* be brought to "bear" in the design of this computer system.

(b) (10 pts) Bob works for company Bazfoo as a software architect. In order to meet an impending software release deadline, Bob is considering the use of the OpenStreetMap package, which is licensed under a *Creative Commons* license. What factors should Bob consider in deciding whether he should move forward with this plan?

3. Exceptions and Assertions (30 pts)

Consider the following code in which Exception1 and Exception2 are checked exceptions:

```
public class exceptionTest
{
    static public void methodA() throws Exception2 { ...   // Do some stuff
    };

    static public void methodB() throws Exception1 { ...   // Do some other stuff
    };

    static public void bar() throws Exception1, Exception2{
        try {
            System.out.println(”A”);
            methodB();
            System.out.println(”B”);
        }catch(Exception1 e){
            System.out.println(”C”);
            throw e;
        }finally{
            System.out.println(”D”);
        };
        System.out.println(”E”);
    }

    static public void main(String[] args) {
        try {
            try {
                System.out.println(”F”);
                bar();
                methodA();
                System.out.println(”G”);
            }catch(Exception2 e){
                System.out.println(”H”);
            };
        }catch(Exception1 e){
            System.out.println(”I”);
        };
        System.out.println(”J”);
    };
}
```

(a) (10 pts) Suppose that methodB() throws Exception1 during execution. What output does this program produce and does the program terminate with or without an error? (assume that output is only produced by the println() calls that you see)

(b) (10 pts) Suppose that methodA() throws Exception2 during execution. What output does this program produce and does the program terminate with or without an error?

(c) (10 pts) True or False and briefly explain: Assertions can be addressed in code using try/catch statements.

4. **Inheritance and Polymorphism** (30 pts)

Consider the following implementation:

```
public class A
{
    protected String name;

    public A(String name){
      this.name = name;
    }

    public String toString(){
      return(”A: ” + getName());
    };

    public String getName() {
      return name;
    };
}

public class B extends A
{
    private String name;

    public B(String name) {
      super(”SUPER–B”);
      this.name = name;
    }

    public String getName() {
      return ”Name: ” + name;
    };
}

public class C extends B
{
    private String name;

    public C(String name){
      super(”SUPER–C”);
      this.name = name;
    };

    public String toString() {
      return ”C: ” + super.toString();
    };
};

public class driver
{
    public static void main(String args[]) {
      A[] objects = new A[4];

      objects[0] = new A(”foo”);
      objects[1] = new B(”bar”);
      objects[2] = new C(”baz”);

      for(int i = 0; i < objects.length; ++i) {
        System.out.println(objects[i]);
      };
    };
};
```

(a) (15 pts) Draw the corresponding UML diagram.

(b) (15 pts) What output does this program produce?

5. Generic Programming and Generics (15 pts)

(15 pts) Assume that classes A, B, and C are the same as from the previous question. Explain whether each of the three **display()** method calls will compile or not.

```
public class genericTest
{
    public static <E1, E2 extends E1> void display(E1 c1, E2 c2){
      System.out.println(c1 + " * " + c2);
    };

    public static void main(String[] args) {
      A a = new A("foo");
      B b = new B("bar");
      C c = new C("baz");

      display(a, b);        // Does it compile?





      display(c, b);        // Does it compile?





      display(b, b);        // Does it compile?

    };
}
```

6. **Graphical User Interfaces and Event-Driven Programming** (40 pts)

(a) (10 pts) True or False and briefly explain: the getPreferredSize() method of a JPanel requires information from its associated layout manager.

(b) (10 pts) Suppose you were to implement your own class, MyButton, that extends JButton. Briefly describe the conditions under which you would provide an implementation of paintComponent() and the types of method calls that you would make inside of this method.

(c) (10 pts) Suppose we wish to implement a program that shows an **EyeFrame**, a class that extends JFrame. A pair of eyes are painted in one corner of this frame. The eyes will: 1) blink at regular intervals and 2) appear to follow the location of the cursor. Suppose that EyeFrame has two properties: eyesOpen (boolean) and eyeOrientation (a set of doubles that indicate which direction each eye is looking). What two event source classes will be used to implement this functionality?

(d) (10 pts) At least three methods are required to implement this functionality. What are their names, what do they do and to which classes do they belong? Hint: the method names and signatures are already determined for you.

7. **Java Collections Framework** (20 pts)

Consider the following code:

```
import java.util.*;

public class driver2 {

    public static void displayMap(Map m) {
      System.out.println(”#”);
      for(Object o: m.keySet()) {
        System.out.println(o + ”: ” + m.get(o));
      };
    };

    public static void addMaps(Map<Integer, String> m_i_s,
                               Map<String, Integer> m_s_i,
                               Integer i,
                               String s) {
      m_i_s.put(i, s);
      m_s_i.put(s, i);
    };

    public static void main(String[] args) {
      Map<Integer, String> m_i_s = new TreeMap<Integer, String>();
      Map<String, Integer> m_s_i = new TreeMap<String, Integer>();

      addMaps(m_i_s, m_s_i, 1138, ”THX”);
      addMaps(m_i_s, m_s_i, 3417, ”LUH”);
      addMaps(m_i_s, m_s_i, 1042, ”THX”);
      addMaps(m_i_s, m_s_i, 5241, ”SEN”);

      displayMap(m_i_s);

      displayMap(m_s_i);
    };
}
```

(20 pts) What output does the program produce?