Computer Science 136

Spring 1997
Professor Bruce

Final Examination

May 26, 1997

Question	Points	Score
1	10	____
2	16	____
3	14	____
4	10	____
5	6	____
6	10	____
7	20	____
TOTAL	86	____

Your name (Please print) _____

I have neither given nor received aid on this examination.

1. In your second version of the Josephus problem, you implemented a class, `DblyCircularList`, representing a doubly-linked circular list. Please write a method for that class to delete the last element in the list.

   You may assume that doubly-linked nodes have the following methods:

```
public DoublyLinkedListElement next()
// post: returns the element that follows this

public DoublyLinkedListElement previous()
// post: returns element that precedes this

public Object value()
// post: returns value stored here

public void setNext(DoublyLinkedListElement next)
// post: establishes a new reference to the next value

public void setPrevious(DoublyLinkedListElement prev)
// post: establishes a new reference to a prev value

public void setValue(Object value)
// post: sets a new value for this object
```

   The `DblyCircularList` class has the following instance variables:

```
protected DoublyLinkedListElement head;
protected int count;
```

   Be sure to take care of special cases:

```
public Object removeFromTail()
// pre: list is not empty
// post: removes value from tail of list
```

2. A simple implementation of a set can be given using a Collection class to hold the elements:

```
public interface Collection extends Container {
    public boolean contains(Object value);
    public void add(Object value);
    public Object remove(Object value);
    public Iterator elements();
}
```

An extract of the code for the class is given below:

```
public class Set {
    protected Collection setRep;

    public Set()
    // post: constructs a new, empty set
    {
        setRep = new ????;
        // Replace ???? by a specific class constructor
    }

    public boolean isEmpty()
    // post: returns true iff set is empty

    public void add(Object e) {
    // pre: e is non-null object
    // post: adds element e to interface
       if (!setRep.contains(e)) setRep.add(e);
    }

    public Object remove(Object e) {
    // pre: e is non-null object
    // post: e is removed from set, value returned
        setRep.remove(e);
    }

    public boolean contains(Object e)
    // pre: e is non-null
    // post: returns true iff e is in set

    public Iterator elements(){
    // post: returns iterator to traverse the elts of set
        return setRep.elements();
    }
}
```

a. Please write Java code to implement the following method:

```
public Object intersection(Set other)
// pre: other is non-null reference to set
// post: returns set intersection between this & other
```

b. Suppose we replace the ???? in the constructor by `SinglyLinkedList()`. What is the complexity of your implementation of intersection above if the receiver has n elements and `other` has m elements?

c. Suppose we replace the ???? in the constructor by `HashTable(N)`, for `N` an integer. If the load factor of the table is very low, then what will the average complexity of your implementation of intersection be (in terms of m, n, and N)?

d. Suppose the set is restricted to contain elements which are Comparable. If we replace the ???? in the constructor by `SplayTree()`, then what will the average complexity of your implementation of intersection be?

3a. Describe briefly the division method for hash coding. Which kind of table sizes work best with this method?

b. What is a hash clash (or collision)?

c. Define briefly:
   i. primary clustering

   ii. secondary clustering

d. Describe briefly the open addressing and chaining methods of dealing with hash clashes.

e. Define the load factor, α, of a hash table.

f. Suppose we use open addressing to deal with hash clashes. One of my texts describes double hashing as "one of the best methods available for open addressing." Please briefly explain double hashing and discuss its advantages and disadvantages over linear and quadratic probing. Be sure to compare them with respect to both primary and secondary clustering.

4. Concurrency and threads:

a. What is a thread and what does it have to do with concurrency?

b. Please explain what the impact of sending the following messages to a thread would be:

   i. start()

   ii. sleep(50);

c. Suppose an object, `testObj`, with `int` field `value`, has methods,

```
public void set(int n){
   value = n;
}

public int get(){
   return value;
}

public void write(int n){
   set(n);
   int temp = get();
   System.out.println("The value of n is "+temp);
}
```

Normally, executing the `write` method results in printing the value of n.

(i) What Java language construct can we use to ensure that only one thread is executing in these three methods at a time? Show how it would be used with these methods.

(ii) What might happen if two threads could be executing in these methods at the same time? Describe a scenario in which `write` would not behave the same way as if only one thread were allowed in it at a time.

5. Consider the following code fragment where `infile` is a variable of type `DataInputStream` which has been successfully created (and attached to an existing file).

```
try
{
   while (true)
   {
      int number = infile.readInt();
      System.out.println("The number is "+number);
   }
} catch (EOFException eofEx){
   System.out.println("EOF exception");
} catch (IOException ioEX){
   System.out.println("IO Exception");
}
System.out.println("End of Code");
```

a. What is printed out by the code fragment if 24, 37, and 16 were originally written to the file (when created as a `DataOutputStream`)?

b. What is printed out by the code fragment if 24, 33, "Hello", 18, and 29 were originally written to the file?

6. Java and object-oriented languages:

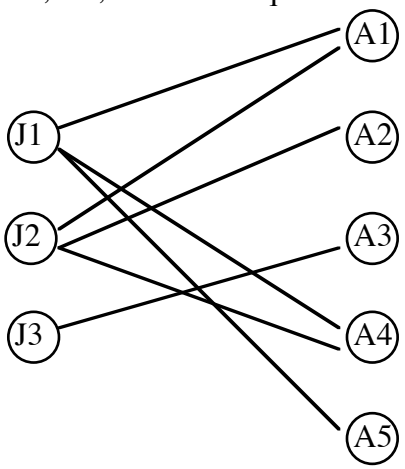
a. What is the difference between an interface and an abstract class in Java?

b. Please explain the life cycle of an applet in terms of the three methods `init()`, `start()`, and `action(Event evt)`.

c. Please explain the restrictions Java applies to applets that do not apply to applications. Why are those restrictions there?

d. Question 1 of this exam involves the class `DblyCircularList`. Please write the complete Java code for a subclass of that class which includes a method `printCount()` which prints the current value of the field `count`.

7. A bipartite graph is a special graph in which the vertices are divided into two sets, and the only edges are between vertices in different sets. For example we can images a graph with vertices representing applicants and jobs. Two vertices will be connected if one of them represents an applicant and the other a job for which that person is qualified. The following graph represents such a graph in which, for example, only applicants A1, A4, and A5 are qualified for job J1.



The program should allow the user to specify one of the following options:

1. Insert (a) a new applicant or (b) a new job position into the graph.
2. Remove (a) an existing applicant or (b) job position from the graph.
3. Add the fact that a particular applicant is qualified for a particular job.
4. List (on the screen) all persons qualified for a specified job position.
5. List (on the screen) all job positions for which a specified person is qualified.

You may assume that all applicant and job information is held in objects of class `Applicant` and `Job`, each of which includes methods for ordering, equals, and finding hash codes. Similarly, you may presume that a class `Edge` is available with methods to return the applicant and job involved. We will presume that the interface `BipartiteIntface` specifies the names and behavior of all of the methods. In this problem we will discuss two possible implementations of this interface.

a. In this subproblem you will implement parts of a class implementing a bipartite graph in a way analogous to that for an *adjacency matrix*. **Be sure to use the fact that no edges go between applicants or jobs to save space!**

i. Draw a picture corresponding to the graph above using the representation you design.

ii. Write the class header, all field definitions necessary to hold all of the associated information, and a class constructor which takes parameters representing the maximum number of jobs, nJobs, and applicants, nApps, in use at once, and creates an empty graph. You may omit the "freelist" of unused rows and/or columns for simplicity.