MIDDLE EAST TECHNICAL UNIVERSITY
Department of Computer Engineering

CEng 546 – Object-Oriented Programming Languages and Systems

2004-2005 Spring **Take-Home Final Exam**
Dr. C. Şener **Section 01**

IMPORTANT

- Submit a hard-copy of your answers till 10.00 a.m. on June 10, Friday, to my office, B-206. You may throw into the box provided there.
- You will lose one point per minute for submissions later than 10.00 a.m. on June 10.
- Soft-copies and any submissions with an answer copied from somewhere or someone will just be ignored as a whole.
- Please, write your *name* and *section number* at the top of your answer sheet.
- I strictly suggest you to test your Java codes using a compiler before you submit.
- You are allowed to make use of the books, other references, Internet etc. However, the codes and sentences in your answers are strictly required to be original and proprietary.
- Please, be brief and clear as much as possible.

Question 1. (5 pts.)
Considering the Java language, assume that *X* extends *SuperX*, both of them define own versions of *methodABC( )*, and *this* object is an instance of *X*. Can

```
((SuperX)this).methodABC()
```

call an overridden *methodABC( )* of a superclass *SuperX*? Explain with a sentence.

Question 2. (5 pts.)
Why do you think Java does not automatically make all classes Serializable? Won't that make it easy to output objects?

Question 3. (5 pts.)
Which EJB type you prefer for each of the following cases:
a) If something needs to be accessed across multiple sessions …
b) If something shouldn't be shared across multiple session, but needs to be shared within a session (i.e. across multiple method calls) …
c) If something shouldn't be shared within a session (i.e. across multiple method calls) …

Question 4. (5 pts.)
Assume that someone is developing a Java program in which it needs to access a CORBA environment. When should he prefer using Java IDL over RMI-IIOP, and when not?

Question 5. (10 pts.)
Explain briefly the complete life-cycle of a statefull session bean.

Question 6. (10 pts.)
Consider the following Java class

```
public class Person {
  protected String firstName;
  protected String lastName;
  public Person(String first, String last){
    firstName = first;
    lastName = last;
  }
  public String whoAmI(){
    return lastName + ", " + firstName;
  }
}
```

and the following code segment

```
for (int i = 0; i < people.length; i++) {
  Person p = (Person) people[i];
  System.out.println("Person " + i + " is " + p.whoAmI());
}
```

Is it guaranteed that all Persons will identify themselves by returning a string composed of last name followed by a comma followed by first name (but, for example, not by returning a string composed of social-security number) when the code in the above for loop calling the method *whoAmI( )* executes –assuming that the code does not produce any error or exception? Explain very briefly.

Question 7. (15 pts.)
Write a *thread-safe* class `Combinatorial` with the instance methods:

- `void pool(int x)`
  pools the value `x` into the instance
- `int min(int n)`
  blocks if the number of pooled values is less than `n`, then returns the minimum of the pooled `n` values.
- `int max(int n)`
  blocks if the number of pooled values is less than `n`, then returns the maximum of the pooled `n` values.
- `double average(int n)`
  blocks if the number of pooled values is less than `n`, then returns the average of the pooled `n` values.

For example, the segment

```
Combinatorial comb = new Combinatorial();
...
minOfThem = comb.min(7);
maxOfThem = comb.max(7);
```

will block the caller if seven integers are not pooled by some (possibly seven) threads into `comb` yet. If or when they are all pooled, it will return the min and max of them.

Also, you are required to write and include a multi-threaded program to test it.

Question 8. (15 pts.)
Write a Java program to perform the following tasks:

- It will check whether the class file "`Calculator.class`" is available at run-time or not.
- If exists, it will try to locate its method with the signature
  `int calculate(int code,double op1,double op2);`
- If located, it will call
  `result = calculate(2,3.5,-4.0);`
  and print out the `result`.
- At each step, it should print messages about the errors or exceptions, if any.

Question 9. (15 pts.)

Create a new version of the following program that has the same behavior but uses the model-view-controller pattern. Indicate separately and clearly the MVC sections in your code.

```
import java.io.*;
public class AMonolithicProgram {
  public static void main (String args[]) throws java.io.IOException {
     System.out.print("Enter the side: ");
     BufferedReader read=new BufferedReader(new InputStreamReader(System.in));
     int side = Integer.parseInt(read.readLine());
     int areaOfSquare = side*side;
     System.out.println("Area of the square is "+ areaOfSquare);
  }
}
```

Question 10. (15 pts.)

Consider the classes *Part* and *SubPart*. Assume that we need to serialize to make their instances persistent. An instance of the class *Part* contains the fields:

- *id* (of type `int`)
- *name* (of type `String`)
- *density* (of type `double`)
- *volume* (of type `double`)
- *mass* (of type `double`) – calculated field; never saved or retrieved; so never serialized;
  when an instance is formed, its *mass* field is set to its *density* times its *volume*
- *itsPart* (of type *SubPart*)

and also an instance of the class *SubPart* contains the fields:

- *id* (of type `int`)
- *name* (of type `String`)
- *position* (of type `long`) – calculated during the execution; never saved or retrieved;
  so never serialized; also no need to initialize this field

Write the Java classes *Part* and *SubPart* containing their field definitions and the code required for their serialization process as described above, only.