Write the answers to question 1 and 2 on this sheet. Write the other answers in the exam booklet.

1. True/False. Please circle the correct answer on this sheet.

   (a) T In Java, the expression \( x \text{ instanceof Object} \) returns true whenever \( x \) is a variable that refers to an object of some user-defined class.
   (b) F A Java interface can extend a Java class.
   (c) F A static method of a class cannot create objects of that class.
   (d) F A UML sequence diagram models the structural relationship (inheritance, containment, etc) between object types.
   (e) T If class B extends class A in Java, then an object of class B is also an object of class A.
   (f) T If a class \( C \) contains the main method and the class is defined in the package \( A.B \), then you execute the program by typing \( java A.B.C \).
   (g) F static synchronized methods, e.g.

   ```java
   static synchronized void f() { .... }
   ```

   synchronize upon the \( this \) object.
   (h) F If you extend the \texttt{Applet} class, you must override the \texttt{init()}, \texttt{start()}, \texttt{stop()}, and \texttt{paint()} methods in order for your code to compile.
   (i) T \texttt{Javac} compiles the Java source code to Java Byte Code, which then gets interpreted by the Java Virtual Machine.
   (j) T The programming language we have studied so far derives its name, ultimately, from an island in the Pacific Ocean.

2. Multiple Choice. Circle the letter next to the correct answer.

   (a) Which \textbf{one} of the following is true?

   i. If an exception is raised in a method \( f \) and there is no try block in \( f \) that handles the exception, then \( f \) exits and the exception propagates to the method that called \( f \).
   ii. If an exception is raised in a method \( f \) and there is no try block in \( f \) that handles the exception, then if \( f \) is \textit{not} declared as throwing the exception, the entire program terminates.
   iii. If an exception is raised in method \( f \) and the exception is handled by a \texttt{catch} clause within \( f \), then \( f \) resumes executing at the point in the code that the exception was raised.
   iv. The \texttt{finally} clause of a try block is executed only if an exception is not caught by a \texttt{catch} clause of that try block.

   (b) Which one of the following is \textbf{not} a necessary feature of an object oriented programming language?

   i. encapsulation of data and code into a single structure
   ii. \texttt{threads}
   iii. inheritance with subtyping
   iv. method overriding

   (c) Complete this sentence: A Java thread is

   i. any object that has a \texttt{run()} method
   ii. an object containing code that can run concurrently with other parts of a program.
   iii. required in order to display moving graphics inside a frame.
   iv. any object that implements the \texttt{Runnable} interface.
(d) Given the interface

```java
interface myInterface { void foo(); void bar(int x); }
```

which one of the following will **not** compile?

i. interface yourInterface extends myInterface {}

ii. class myClass implements myInterface { void foo() {}

```java
void bar(int x) {}
```

iii. class myClass implements myInterface { public void foo() {} public void bar(int x) {}

```java
public void bar(int x, int y) {} }
```

iv. interface yourInterface extends myInterface { void bar(int x, int y); }

(e) Which of the following is not true about UML class diagrams?

i. An arrow with a triangular head denotes an “is” relationship between classes.

ii. A line denotes a “uses” relationship between classes.

iii. A arrow with a diamond head denotes a “contains” relationship between classes.

iv. **None of the above are true.**

3. (a) Write the definition of a class **Person** that contains a **name** field of type **String** and an **age** field of type **int**. Then define two classes, **Dancer** and **DonutMaker**, that extend **Person** such that **Dancer** has an additional field called **Company** of type **String**, and **DonutMaker** has an additional field called **Shop** of type **String**.

```java
class Person {
    int age = 10;
    String name = "Jane Doe";
}
class Dancer extends Person {
    String Company = "Bolshoi";
}
class DonutMaker extends Person {
    String Shop = "Dunkin Donuts";
}
```

(b) Define a class **Counting** that contains a method **DancersAges(Person[] a)**, such that **DancersAges** returns the sum of the **age** fields of the **Dancer** objects referenced by the array **a**. There is no need for an object of class **Counting** to be created in order for **DancersAges** to be called.

```java
class Counting {
    static int DancersAges(Person[] a) {
        int sum = 0;
        for(int i=0; i<a.length; i++)
            if (a[i] instanceof Dancer)
                sum = sum + a[i].age;
        return sum;
    }
}
```

(c) Define a class **MainClass** containing the main method (for the entire program). **main** should create an array with 10 elements, namely 5 **Dancer** objects and 5 **DonutMaker** objects in alternating order, and pass the array to the **DancersAges** method, above. You can initialize the fields of the objects as you see fit.

```java
class MainClass {
    public static void main(String[] args) {
        Person[] a = new Person[10];
        for(int i = 0; i < 10; i = i+2) {
            a[i] = new Dancer();
            a[i+1] = new DonutMaker();
        }
    }
```
System.out.println(Counting.DancersAges(a));
}
}

4. What will be displayed when the program below runs?

class MyThread extends Thread {
    static Object o = new Object();
    int n;
    MyThread(int x) { n = x; }
    public synchronized void run() {
        for(int i = n; i<n+5; i++) {
            System.out.println(i + " A");
            yield();
            synchronized(o) {
                System.out.println(i + " B");
                yield();
                System.out.println(i + " C");
            }
            yield();
        }
    }
}

class test {
    public static void main(String args[]) {
        MyThread t1 = new MyThread(0);
        MyThread t2 = new MyThread(10);
        t1.start();
        t2.start();
    }
}

There are a number of possibilities. One important thing to note is that initially, the same thread will print “B” and “C” consecutively. The yield() in between those print statements will not affect the output since it is inside a block that both threads are synchronizing on. Thus, one possible output sequence is:

```
0 A
10 A
0 B
0 C
10 B
1 A
10 C
1 B
11 A
1 C
11 B
2 A
11 C
2 B
12 A
2 C
12 B
3 A
12 C
3 B
```
13 A
3 C
13 B
4 A
13 C
14 A
4 B
14 B
4 C
14 C