CSE 1020 Introduction to Computer Science I

Exam

9:00–11:00, December 20, 2010

Last name:

First name:

Instructions

- This is a closed book test. No aids are permitted.
- Answer each question in the space provided.
- Make sure that you have answered all questions (double sided).
- Manage your time carefully.
- Last page can be used as scrap paper.

Question	Maximum	Mark
1	8	
2	20	
3	8	
4	28	
5	4	
6	28	
7	14	
8	10	
Total	120	

1 (8 marks)

(a) A *static* method is invoked on a(n) ....

(b) A *non-static* method is invoked on a(n) ....

2 (20 marks)

(a) What is the *state* of an object?

(b) What is the *identity* of an object?

(c) What does the binary operator `==` check?

(d) What does the method `equals` check?

(e) Consider the following fragment of Java code:

```
Fraction f1 = new Fraction(1, 2);
Fraction f2 = new Fraction(3, 4);
Fraction f3 = f1;
```

Circle the expressions below that evaluate to `true`:

```
f1 == f1           f1 == f2           f2 == f3           f1 == f3

f1.equals(f1)      f1.equals(f2)      f2.equals(f3)      f1.equals(f3)
```

3 (8 marks)

What are the *two* differences between a `Set` and a `List`?

4 (28 marks)

Consider the following fragment of (legal) Java code:

```
1 Number num = new BigInteger("123456789123456789");
2 double val = num.doubleValue();
3 num = ((BigInteger) num).subtract(new BigInteger("1"));
```

`BigInteger` extends `Number`.
The method `doubleValue()` is polymorphic.
The cast on line 3 is necessary for the code to compile.

(a) What class or classes are searched during early binding for the method `doubleValue()`? **Justify your answer**.

(b) What class or classes are searched during early binding for the method `subtract(BigInteger)`? **Justify your answer**.

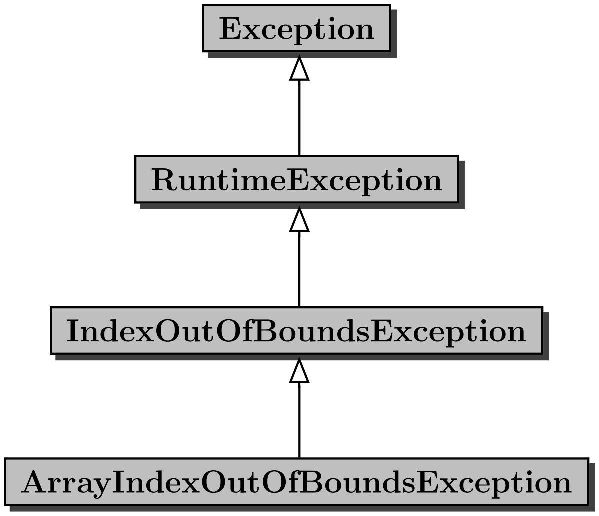
(c) What class or classes are searched during late binding for the method `doubleValue()`? **Justify your answer**.

5 (4 marks)

Explain how composition differs from aggregation.

6 (28 marks)

Consider the following UML diagram.



(a) Consider the following main method.

```
public static void main(String[] args)
{
   PrintStream output = System.out;

   try
   {
      output.println(args[-1]);
      ...
   }
   catch (ArrayIndexOutOfBoundsException e)
   {
      output.println("Invalid array index is used");
   }
   catch (Exception e)
   {
      output.println("Something went wrong!");
   }
}
```

The above main method produces the following output.

```
Invalid array index is used
```

Explain what happens when the code is run. In your explanation, use the following phrases if they are relevant (not all may be relevant):

- throw(s) an exception
- terminate(s) immediately
- determine(s) the appropriate
- substitutable/substitutability

(b) Consider the following main method.

```
public static void main(String[] args)
{
   PrintStream output = System.out;

   try
   {
      output.println(args[-1]);
      ...
   }
   catch (IndexOutOfBoundsException e)
   {
      output.println("Invalid index is used");
   }
   catch (Exception e)
   {
      output.println("Something went wrong!");
   }
}
```

The above main method produces the following output.

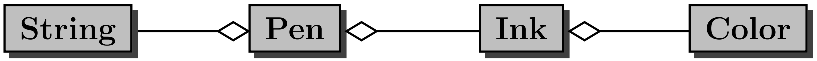
```
Invalid index is used
```

Explain what happens when the code is run. In your explanation, use the following phrases if they are relevant (not all may be relevant):

- throw(s) an exception
- terminate(s) immediately
- determine(s) the appropriate
- substitutable/substitutability

7 (14 marks)

Consider the following UML diagram.



Consider the following code snippet (which appears in the body of a main method).

```
1 Pen pen = new Pen("Papermate", new Ink(Color.BLUE));
2 Pen alias = ...;
3 Pen shallow = ...;
4 Pen deep = ...;
```

Assume that memory can be depicted by the diagram below when the execution reaches the end of line 1. Recall that the `String` class is immutable.

⋮	
100	`main` invocation
`pen`	600
`alias`	
`shallow`	
`deep`	
200	`Pen` class
300	`Ink` class
400	`Color` class
500	`String` class
600	`Pen` object
`name`	700
`ink`	800
700	`String` object
	`"Papermate"`
800	`Ink` object
`color`	900
900	`Color` object
⋮	

(a) Consider that an *alias* of `pen` is assigned to `alias` in line 2. Draw the diagram depicting memory when the execution reaches the end of line 2. *Draw only those blocks which have changed or are new.*

(b) Consider that a *shallow* copy of `pen` is assigned to `shallow` in line 3. Draw the diagram depicting memory when the execution reaches the end of line 3. *Draw only those blocks which have changed or are new.*

(c) Consider that a *deep* copy of `pen` is assigned to `deep` in line 4. Draw the diagram depicting memory when the execution reaches the end of line 4. *Draw only those blocks which have changed or are new.*

8 (10 marks)

(a) What is a loop invariant?

(b) Searching a collection for a particular element is performed using a loop. Suppose that we want to know whether or not a list contains at least one string that starts with `z`. Consider the following attempt:

```
// t is a List<String>
boolean found = false;
for (int i = 0; i < t.size(); i++)
{
   found = t.get(i).startsWith("z");
}
```

What is a useful loop invariant in the code fragment shown above?

(c) The loop invariant in Part (b) does not correctly solve the search problem. What is a useful loop invariant that correctly solves the search problem?

(d) Using big-O notation, state the complexity of the loop shown in Part (b) where N is the number of elements in the list.

(e) Suppose we want to solve a different search problem: Does a set contain all of the elements of another set? Consider the following attempt:

```
// set1 is a Set<String>
// set2 is a Set<String>
boolean containsAll = true;
for (String s1 : set1)
{
   boolean found = false;
   for (String s2 : set2)
   {
      found = found || s1.equals(s2);
   }
   containsAll = containsAll && found;
}
```

Using big-O notation, state the complexity of the code fragment shown above. Explain your answer (but do not formally prove your answer). You can assume that both sets always have N elements.