COMP 401
Spring 2013
Midterm 1

I have not received nor given any unauthorized assistance in completing this exam.

Signature: _____

Name: _____

PID: _____

Please be sure to put your PID at the top of each page.

PID: _____

This page left intentionally blank. Do not write on it.

PID: _____

Part I: Multiple choice (45 points total)

Directions:
For each multiple choice question below, indicate all correct answers among the choices provided. More than one answer may be correct and at least one answer will be correct. Each question is worth 3 points. Selecting all of the choices or selecting none of the choices will result in no points awarded.

1) Which of these are valid built-in Java data types?
* a) `int`
b) `unsigned long`
* c) `String`
d) `null`
* e) `boolean`

2) Which of these are reference types?
* a) `double[]`
b) `byte`
* c) `String`
d) `char`
e) `boolean`

3) Which of these statements about Java arrays is true?
a) Arrays are indexed numerically starting at 1.
b) The number of elements in an array can change.
* c) Each element of an array must be of the same type.
* d) The "new" keyword is used to create an array.
e) Negative index values access elements from the end of an array.

4) Which of these are legal Java variable names?
* a) `a_var`
* b) `_b_var`
c) `1var`
* d) `$var`
e) `while`

5) Which of these are NOT legal Java expressions?
* a) `false && (1-8)`
b) `4+3*2`
c) `Math.PI`
d) `"hello" + 3`
* e) `"hello" * 3`

PID: _____

6) Which of these statements about Java statements and blocks are true?
* a) A statement must end in a semicolon.
b) A statement must result in a value.
* c) A single statement can be placed anywhere a block of statements is expected.
d) A block of statements is enclosed by parentheses.
* e) A block of statements can be empty.

7) Which of these statements are correctly related to the principle of encapsulation?
* a) Object state should be maintained in private instance fields.
b) An object should provide direct access to its properties.
c) Every object property must have both a getter and a setter.
d) Java is a strongly-typed language.
* e) Setters should validate values if possible.

8) Which of these statements about Java interfaces are true?
a) Interfaces define a set of private methods that an object implements.
b) Interface names are not valid data types for variables.
c) Interfaces provide default implementations for specified methods.
* d) Interfaces can be extended using inheritance.
* e) Interfaces never specify constructors.

9) Which of these statements about polymorphism is true?
* a) Constructor overloading is an example of polymorphism.
b) Method overloading requires that different versions of the same method have the same method signature.
c) An overloaded method can only invoke another version of the same method if it does so on the first line of code in the body of the method.
d) Polymorphism hides object state in order to support abstraction.
* e) Polymorphism allows methods to be context-specific.

10) Which of these statements about enumerations is true?
* a) Enumerations define a new data type.
b) Symbols can be added to an enumeration after it has been declared.
c) The symbols of an enumeration can be used as integer values in expressions.
* d) Enumeration symbols can be used in a "switch" statement.
* e) One advantage of using an enumeration is type safety.

PID: _____

11) Which of these statements properly declares a named constant?

a) `public int A_CONSTANT;`

b) `static public int A_CONSTANT;`

* c) `static final public int A_CONSTANT = 3;`

* d) `static final int A_CONSTANT = 3;`

e) `static final public int A_CONSTANT;`

12) Which of these statements about the iterator design pattern are true?

* a) An iterable collection should provide a method that creates a new iterator instance.

b) The state of an iteration is encapsulated within the iterable collection.

c) If two iterators are created for the same iterable collection, advancing one will advance the other.

* d) An iterator generally assumes that the underlying collection is not altered while being traversed.

e) A single iterator can be used to traverse more than one collection.

13) Which of these statements is true of the String data type?

a) String is equivalent to an array of char elements.

* b) Strings are immutable.

* c) Strings are reference types.

d) The most appropriate way to compare to Strings is to use the == operator.

e) An empty string (i.e., a string of length 0) is the same as the value null.

14) Which of the following statements is true of objects?

* a) Every object is associated with a class.

b) Every object is associated with at least one interface.

* c) Two object references that have the same value refer to the same object.

* d) The state of an object is determined by the current values of its instance fields.

e) A new copy of an object is created when it is passed to a method as a parameter.

15) Which of the following are legal method signatures?

* a) `public void int foo()`

* b) `private static int foo()`

c) `double foo(a, b)`

* d) `String[] foo(String a, String b)`

* e) `String foo(int a, String[] b)`

PID: _____

Part II: Short answer / calculation (25 points total)

Directions:
Calculate the value of the variable or expression indicated. Each question is worth 5 points.

16) What is the value of the variable d after the code below executes?

Answer: 6

```
int a[] = new int[] {1, 2, 3};
int b[] = new int[] {4, 5, 6};
int c[] = new int[] {a[0], b[1], a[2]};
b = a;
a = c;
c = b;
int d = a[0] + b[1] + c[2];
```

17) What is the value of bar after the code below executes?

Answer: 25

```
int foo = 30;
int bar = 15;

if (foo < 90) {
  bar += 10;
  foo *= 3;
} else if (foo < 80) {
  bar += 5;
} else if (foo > 70)
  bar -= 5;
}
```

PID: _____

Questions 18 and 19 use the definitions of the functions *foo* and *bar* given below:

```
int foo(String s) {
    if (s.equals("hello")) {
        bar(s);
        return s.length;
    } else {
        return bar(s).length;
    }
}

String bar(String t) {
    String s = "good";
    if (t.equals("bye")) {
        return s;
    } else {
        return s+t;
    }
}
```

18) What is the value of the expression `foo("hello")`?

Answer: 5

19) What is the value of the expression `foo("bye")`?

Answer: 4

20) What is the value of sum after the following code executes?

Answer: 2

```
int sum = 0;

for (int i=0; i<4; i++) {
  for (int j=i; j<i*2; j++) {
    if (j<4) {
       sum++;
   } else {
       sum--;
   }
}
```

PID: _____

Part III: Reading and understanding code (15 points total)

Directions:
All of the questions in this section concern the code for the `Scoreboard` class given below (note that the class definition continues on to the next page). This class is intended to be an abstraction for keeping track of the score of a basketball game. In this abstraction, teams can score either 2-point baskets or 3-point baskets that are recorded by using the methods `scoreTwo` and `scoreThree` respectively. Each question is worth 3 points.

```
class Scoreboard {

     enum Team {HOME, VISITOR};

     private String home_name;
     private String visitor_name;
     private String home_mascot;
     private String visitor_mascot;

     private int home_score;
     private int visitor_score;

     Scoreboard(String home_name,
                String home_mascot,
                String visitor_name,
                String visitor_mascot) {

          this.home_name = home_name;
          this.home_mascot = home_mascot;
          this.visitor_name = visitor_name;
          this.vistor_mascot = visitor_mascot;

          home_score = 0;
          visitor_score = 0;
      }

     public String getCurrentLeader() {
          if (home_score > visitor_score) {
               return home_name;
          } else {
               return visitor_name;
          }
      }

      public int getHomeScore() {
           return home_score;
      }
```

```
    public int getVisitorScore() {
         return visitor_score;
    }

    public int scoreTwo(Team who) {
        if (who == Team.HOME) {
             home_score += 2;
             return home_score;
        } else {
             visitor_score += 2;
             return visitor_score;
        }
    }

    public int scoreThree(Team who) {
        if (who == Team.HOME) {
             home_score += 3;
             return home_score;
        } else {
             visitor_score += 3;
             return visitor_score;
        }
    }
}
```

21) How many parameters does the `Scoreboard` constructor accept?

4

22) Which method(s), if any, act as getters following the JavaBeans convention?

getCurrentLeader, getHomeScore, getVisitorScore

23) Which method(s) act as a getter for a derived property?

getCurrentLeader

24) Which method(s), if any, act as setters following the JavaBean convention?

None

25) Is the following statement true or false: A `Scoreboard` object is immutable (i.e., has no state that changes after creation). False

PID: _____

Part IV: Writing Code (20 points total)

Directions:
In this section, you will create and use an abstraction for the immutable team information (i.e., does not change after creation) encapsulated by the `Scoreboard` class as a new class called `Team`. Question 26 is worth 3 points, Question 27 is worth 7 points, and Questions 28 is worth 10 points.

26) Identify which object fields originally part of `Scoreboard` should be encapsulated by the class `Team`.

```
home_name, home_mascot, visitor_name, visitor_mascot
```

27) Write a definition for the class `Team` that includes an appropriate constructor and getter methods.

```
public class Team {
    private String name;
    private String mascot;

    Team (String name, String mascot) {
        this.name = name;
        this.mascot = mascot;
    }

    public String getName() {
        return name;
    }

    public String getMascot() {
        return mascot;
    }
}
```

PID: _____

28) Rewrite the `Scoreboard` class to use the new `Team` class you developed as the answer for Question 27. You are free to change the implementation of any or all of the existing methods, but some version of the existing methods must still be provided. Furthermore, be sure to include a new constructor that accepts two `Team` objects and to rewrite the existing constructor to chain from this new constructor.

```
class Scoreboard {

     private Team home;
     private Team visitor;

     private int home_score;
     private int visitor_score;

     Scoreboard(Team home, Team visitor) {
          this.home = home;
          this.visitor = visitor;
     }

     Scoreboard(String home_name,
                String home_mascot,
                String visitor_name,
                String visitor_mascot) {

          this(new Team(home_name, home_mascot),
               new Team(visitor_name, visitor_mascot));
          home_score = 0;
          visitor_score = 0;
      }

     public String getCurrentLeader() {
          if (home_score > visitor_score) {
               return home.getName();
          } else {
               return visitor.getName();
          }
      }

      public int getHomeScore() {
           return home_score;
      }

      public int getVisitorScore() {
           return visitor_score;
      }

      public int scoreTwo(Team who) {
```

```
            if (who == home) {
                 home_score += 2;
                 return home_score;
            } else {
                 visitor_score += 2;
                 return visitor_score;
            }
       }

       public int scoreThree(Team who) {
            if (who == home) {
                 home_score += 3;
                 return home_score;
            } else {
                 visitor_score += 3;
                 return visitor_score;
            }
       }
}
```