CSCI 135 Exam #1 **Name: _____**
Fundamentals of Computer Science I
Fall 2012

This exam consists of 6 problems on the following 7 pages.

You may use your two-sided hand-written 8 ½ x 11 note sheet during the exam. No calculators, computers, or communication devices of any kind are permitted.

If you have a question, raise your hand and I will stop by. Since partial credit is possible, **please write legibly and show your work**.

Problem	Points	Score
1	8	
2	10	
3	10	
4	14	
5	12	
6	14	
Total	68	

1. Arrays and static methods (8 points). Consider the following program:

```
public class Squares
{
    public static int square(int i)
    {
        return i * i;
    }

    public static void main(String [] args)
    {
        final int N = Integer.parseInt(args[0]);
        double [] vals = new double[N];
        for (int i = 0; i < vals.length; i++)
            vals[i] = square(i);
        for (int i = 0; i < vals.length; i++)
            System.out.printf("vals[%d] = %.2f\n", i, vals[i]);
    }
}
```

Give the output of the following command:

% java Squares 4

2. Classes and debugging (10 points). Consider the following program:

```
01 public class Person
02 {
03     public String name;
04     public int age;
05
06     public Person(String name, int age)
07     {
08         name = name;
09         age = age;
10     }
11
12     public String toString()
13     {
14         return name + " " + age;
15     }
16
17     public static void main(String [] args)
18     {
19         Person p = new Person(args[0],
20                               args[1]);
21         System.out.println("Person: " + p);
22     }
23 }
```

A. Which bug prevents the program from ***compiling*** successfully? Identify the line number where the bug appears and give a correct version of this line of code.

Line number _____

Corrected code:

B. After fixing the compile error, you run your program and get the following:

```
% java Person Bob 21
Person: null 0
```

Which line(s) are responsible for this bug? Provide corrected code.

Line number(s) _____

Corrected code:

C. Which line(s) are violating the principle of data encapsulation? Provide corrected code.

Line number(s) _____

Corrected code:

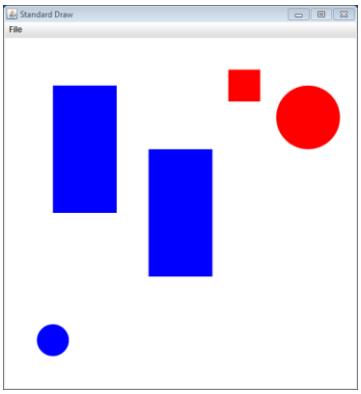
3. Recursion (10 points). Give the sequence of integers printed by a call to `mystery(6)`:

```
public static void mystery(int n)
{
   if (n <= 0)
      return;
   System.out.println(n);
   mystery(n-2);
   mystery(n-3);
   System.out.println(n);
}
```

4. Standard input and drawing (14 points). You are creating a program that reads a file from standard input and draws red and blue circles and rectangles. Here is an input file `shapes.txt` and its `StdDraw` output:

```
rectangle 0.5 0.5 0.10 0.2
rectangle 0.2 0.7 0.10 0.2
circle    0.1 0.1 0.05
circle    0.9 0.8 0.10
rectangle 0.7 0.9 0.05 0.05
```

```
% java CircleRect < shapes.txt
```



If a line starts with "rectangle", it is followed by the rectangle's (x, y) center then its width and height. If a line starts with "circle", it is followed by the circle's (x, y) center then its radius. Fill in a ***single*** letter into each box corresponding to the code that when combined makes a program ***capable*** of producing the above output.

```
public class CircleRect
{
   public static void main(String [] args)
   {
      [          ]
      {
         String s = StdIn.readString();

         StdDraw.setPenColor(StdDraw.BLUE);
         [          ]
         [          ]

         [          ]
         {
            StdDraw.filledCircle(
                                  [          ]
                                 );

         }

         [          ]
         {
            StdDraw.filledRectangle(
                                    [          ]
                                   );
         }
      }
   }
}
```

A	`StdIn.readDouble(), StdIn.readDouble()`
B	`StdIn.readDouble(), StdIn.readDouble(), StdIn.readDouble()`
C	`StdIn.readDouble(), StdIn.readDouble(), StdIn.readDouble(), StdIn.readDouble()`
D	`if (Math.random(0.5))`
E	`if (Math.random() < 0.5)`
F	`while (StdIn.isEmpty())`
G	`while (!StdIn.isEmpty())`
H	`while (StdIn.readString().length() > 0)`
I	`StdDraw.setPenColor(StdDraw.RED);`
J	`StdDraw.setPenColor(StdDraw.BLUE);`
K	`StdDraw.setPenColor(Math.random() % 2);`
L	`if (s.equals("circle"))`
M	`if (s == "circle")`
N	`if (StdIn.readString().equals("circle"))`
O	`else if (s.equals("rectangle"))`
P	`else if (s == "rectangle")`
Q	`else if (s != "rectangle")`

5. Multiple choice (12 points, 2 points each). Circle the ***best*** single answer.

I. Which of the following assignment statements results in a compile error?

a) `int     i = (short) (40 / 2.0)`
b) `boolean b = (1 > 2) && (3 < 2.0);`
c) `double  d = Integer.parseInt("123");`
d) `int     i = Double.parseDouble("3");`

II. Consider a library that provides the following public static methods:

```
int    uniform(int N)
double uniform(double lo, double hi)
```

All of the following methods could be added to the library successfully ***EXCEPT***:

a) `int    uniform(int lo, int hi)`
b) `double uniform(int num)`
c) `double uniform()`
d) `void   uniform(int [] nums, int N)`

III. The `private` keyword when added to the declaration of an instance variable of a class helps enforce which of the following principles?

a) data encapsulation
b) immutability
c) avoiding repeated code
d) divide-and-conquer

IV. The `final` keyword when added to the declaration of an instance variable of a class helps enforce which of the following principles?

a) data encapsulation
b) immutability
c) avoiding repeated code
d) divide-and-conquer

V. All the following are true statements about recursion ***EXCEPT***:

a) A simple recursive algorithm (e.g. computing Fibonacci numbers) can take the age of the universe to run (i.e. it may take exponential time).
b) Recursion is closely related to mathematical induction.
c) A recursive method must always have a base case that does not call itself recursively.
d) Divide-and-conquer algorithms such as binary search can only be implemented using recursion.

VI. All the following are true statements about methods ***EXCEPT***:

a) A parameter of type `String` can be changed inside a method but that change will have ***no*** effect outside the method.
b) The elements of a passed in array parameter can be changed and that change will have an effect outside the method.
c) Two parameters of a given method may have the same name as long as they have different types.
d) Methods can return nothing, a single primitive type, or a single reference type.

6. Objects (14 points). You are writing a class for an airline that represents a checked bag. A Bag object knows things such as its length, width, height (in inches) and its weight (in pounds). A Bag can do things such as:

- Calculate its volume in cubic inches
- Check if its volume is bigger than another Bag
- Check if it is allowed on the aircraft or not. A bag is allow on an aircraft if the ***sum*** of its length, width and height is ***less than or equal to 62 inches*** and its weight is ***less than or equal to 50 pounds***

Part A. Complete the missing body of the instance methods:

```
public class Bag
{
    private double w;    // width in inches
    private double l;    // length in inches
    private double h;    // height in inches
    private double wgt;  // weight in pounds

    // Construct a new bag of width w, length l, height h, and weight wgt
    public Bag(double w, double l, double h, double wgt)
    {









    }

    // Calculate the volume of this bag in inches^3
    public double volume()
    {


    }

    // Returns whether this bag's volume is >= the other bag's volume
    public boolean isBigger(Bag other)
    {




    }

    // Returns whether this bag meets the airline size and weight limits
    public boolean isAllowed()
    {









    }
}
```

(continued on next page)

Part B. Show how to declare and instantiate two objects of the Bag class. Give the variables descriptive names in appropriate style.

- The first bag is 10" wide, 20" long, 30" high and weighs 40 pounds.
- The second bag is 15" wide, 25" long, 35" high and weighs 45 pounds.

Part C. Show how to create an array that can hold 10 Bag objects. Fill the array with objects where each object has randomly assigned sizes and weights. The width, length, height and weight are each assigned randomly to a value between 0.0 (inclusive) and 100.0 (exclusive), i.e. numbers in [0.0, 100.0).