Final Answers

CS 1063 – Spring 2011
April 30, 2011
100 pts. total

Your Name ______________________

Your Instructor ______________________

I. (10 pts., 1 pt. each) Match each definition on the left to *one* of the terms on the right.

cumulative alg. computes an overall value incrementally

pseudocode English-like description of an algorithm

array holds multiple values of the same type

return type the kind of value a method returns

data type a name for a category of values

identifier a name for a class, method, or variable

statement a single command that can be executed

declaration specifies a variable with a name and a type

counter variable that is incremented when a test is true

keyword a word reserved for a particular use

array
counter
cumulative algorithm
data type
declaration
expression
identifier
keyword
precedence
pseudocode
reference
return type
statement
variable

II. (10 pts., 1 pt. each) Match each description on the left to *one* of the notations or keywords on the right.

{ } contains a class or group of statements

[ ] contains an array index

== equality operator

&& logical AND

|| logical OR

double primitive type for real numbers

% remainder or mod operator

while repeat statements as long as a test is true

, comma separates parameters

\ starts an escape sequence

%
&&
++
, (comma)
/
=
==
[ ]
\
double
main
while
{ }
||

III. (10 pts.) What output will be produced by the following program:

```
public class MyProgram {
  public static void main(String[] args) {
    int a     = 6 / 3 + 2;
    double b  = 0.1 * 2.5;
    boolean c = 4.1 < 3.5;
    String d  = "35" + "3";
    int e     = 5 * (int) 2.9;
    double f  = b * 10;
    boolean g = d.length() < 1 || a < 10;
    String h  = "Hello " + "World";
    int i     = h.length();
    double j  = i * 0.5;

    i = i + 9;

    System.out.println("a = " + a);
    System.out.println("b = " + b);
    System.out.println("c = " + c);
    System.out.println("d = " + d);
    System.out.println("e = " + e);
    System.out.println("f = " + f);
    System.out.println("g = " + g);
    System.out.println("h = " + h);
    System.out.println("i = " + i);
    System.out.println("j = " + j);
  }
}
```

```
a = 4
b = 0.25
c = false
d = 353
e = 10
f = 2.5
g = true
h = Hello World
i = 20
j = 5.5
```

IV. (10 pts.) What is the output of the following program?

```
public class MysteryNumbers {
   public static void main(String[] args) {
      String s1 = "two";
      String s2 = "three";
      String s3 = "four";
      int number = 13;

      sentence(s1, s2, number / 2);
      sentence(s2, s3, number - 1);
      sentence(s3, s3, number + 3);
      sentence(s1, s1, number % 9);
      sentence("eight", s3, 6 + number * 2);
   }

   public static void sentence(String s4, String s5, int number) {
      System.out.println(s5 + " times " + s4 + " = " + number);
   }
}
```

```
three times two = 6
four times three = 12
four times four = 16
two times two = 4
four times eight = 32
```

V. (10 pts.) Write a method named `DistancefromOrigin` with parameters, `x, y` (type double). Consider (`x`,`y`) to represent a point in the XY-Plane. The method must compute and return the distance of the point (`x`,`y`) from the origin (0,0). Note: the distance of (`x`,`y`) from the origin in the plane is given by the formula: d = $\sqrt{\texttt{x}*\texttt{x} + \texttt{y}*\texttt{y}}$.

```
public static double DistancefromOrigin(double x, double y) {
   double result = Math.sqrt(x * x + y * y);
   return result;
}
```

VI. (10 pts., 2 pts. each) Consider the following following program:

```
public class WhileLoop {
  public static void main(String[] args) {
    int x = 1;
    int y = 4;
    while (x < y) {
      /* replace me */
    }
    System.out.println(x + " " + y);
  }
}
```

What output would be produced by a program formed by modifying the program above and replacing the line `/* replace me */` with each of the following statements:

`x++;` 4 4

`y--;` 1 1

`y -= 2;` 1 0

`x = x * 3;` 9 4

`y = -1;` 1 -1

VII. (10 pts., 2 pts. each) Consider the following method, named `mystery`. For each of the five calls to `mystery` below, write what the call evaluates to.

```
public static int mystery(String arg)
{
    String[] ds = {"one", "two", "three", "four", "five",
                   "six", "seven", "eight", "nine", "zero"};
    arg = arg.toLowerCase();
    int result = -1;
    for (int i=0; i<ds.length; i++) {
        if (arg.equals(ds[i])) {
            result = (i + 1) % 10;
        }
    }
    return result;
}
```

-1 `mystery("e")`

-1 `mystery("1")`

1 `mystery("ONE")`

7 `mystery("seven")`

-1 `mystery("Fifty")`

VIII. (10 pts.) Write a method that reverses a String. For example, given "method", the method should return "dohtem". Given "program", the method should return "margorp". Recall that the + operator can be used to combine two Strings into a single String. Methods in the String class are listed on page 10. Start with the method header below.

```
public static String reverse(String phrase)
```

```
public static String reverse(String phrase) {
   String result = "";
   for (int i = 0; i < phrase.length(); i++) {
      char c = phrase.charAt(i);
      result = c + result;
   }
   return result;
}
```

IX. (10 pts.) Write a method named `product` that has a single parameter, named `values`, which is an array of `double`s. The method should return the product of all of the `double`s in the array. As soon as the method detects a `0.0` value in the array, it should terminate early (because the product will already be known to be zero). If the array is empty, the value `1.0` should be returned.

```
public static double product(double[] values) {
   double result = 1;
   for (int i = 0; i < values.length; i++) {
      if (values[i] == 0) {
         return 0;
      }
      result *= values[i];
   }
   return result;
}
```

X. (10 pts.) Write a method named `getDouble` that has parameters: `console` (type `Scanner` class) and `prompt` (type `String` class).

`console` is for reading in tokens from the keyboard, and `prompt` is to tell the user to enter a token. The method should return a valid `double` value read in from the keyboard. A while loop with a lookahead method must be used to handle possible user errors. This will make sure that the next token is a double, otherwise it must be discarded. You can assume that `java.util.*` has been imported.

```
public static double getDouble(Scanner console, String prompt) {
   System.out.print(prompt);
   while (! console.hasNextDouble()) {
      String discard = console.next();
      System.out.println(discard + " is not a double");
      System.out.print(prompt);
   }
   double result = console.nextDouble();
   return result;
}
```