QUIZ 2

CMPS 12a - Spring 02

Thomas Raffill

Name:_____

Student ID:_____

This exam is closed book, closed notes, no electronic devices. Show all work. Partial credit given for partial solutions. Presentation counts! Be legible and coherent for full credit.

Question 1:_____(out of 10)
Question 2:_____(out of 10)
Question 3:_____(out of 10)
Question 4:_____(out of 10)
Question 5:_____(out of 10)
Question 6:_____(out of 10)
Question 7:_____(out of 10)
Question 8:_____(out of 10)
Question 9:_____(out of 10)
Question 10:_____(out of 10)

Total:_____(out of 100)

Name:_____

1. (10 points)
   Assume you already have a function

   ```
   boolean divisibleBy(int a, int b)
   ```

   which returns true if a is divisible by b, otherwise returns false.

   Write a code fragment to check whether a number is divisible by 2, 5 or 11 by calling this function.

   The code fragment should print out three lines:
   A line saying the number is or is not divisible by 2.
   A line saying the number is or is not divisible by 5.
   A line saying the number is or is not divisible by 11.

   You don't have to write the function, just show how to call it.

   Solution:

```
if (divisibleBy(x, 2))
  {
  System.out.println(x + " is divisible by 2.");
  }
else
  {
  System.out.println(x + " is not divisible by 2.");
  }
if (divisibleBy(x, 5))
  {
  System.out.println(x + " is divisible by 5.");
  }
else
  {
  System.out.println(x + " is not divisible by 5.");
  }
if (divisibleBy(x, 11)
  {
  System.out.println(x + " is divisible by 11.");
  }
else
  {
  System.out.println(x + " is not divisible by 11.");
  }
```

Name:_____

2. (10 points)
What does this program print out? **Hint:** The function name is meaningful.

```
public class PrintRows
  {
  public static void main(String[] argv)
    {
    printRow(5);
    printRow(2);
    }

  public static void printRow(int n)
    {
    for (int i = 1; i <= n; i++)
      {
      System.out.print("X");
      }
    System.out.println();
    }
  }
```

Solution: It prints a row of 5 X's followed by a row of 2 X's:

```
XXXXX
XX
```

Name:_____

3. (10 points)
Identify the compiler error in the following code fragment. Explain the error and how to fix it.

```
public boolean isEven(int k)
  {
  if (k % 2 == 0)
    {
    boolean n = true;
    }
  else
    {
    boolean n = false;
    }
  return n;
}
```

Solution: The return statement accesses the variable n outside of its scope. There are various ways to fix it. The simplest would be to declare n in the beginning of the function before the conditional branch:

```
public boolean isEven(int k)
{
boolean n;
if (k % 2 == 0)
  {
  n = false;
  }
else
  {
  n = true;
  }
return n;
}
```

Name:_____

4. (10 points)
What does the following program print out?

```
class TrickyScope
  {
  public static void main(String[] argv)
    {
    int i = 10, n = 20;
    printNumber(n);
    }

  public static void printNumber(int i)
    {
    System.out.println(i);
    }
  }
```

Solution: 20. The trick is that the variable i in the subroutine refers to the value 20 that was passed to it, and not the i in the main function.

Name:_____

5. (10 points)
Which of the following two boolean expressions will correctly avoid a zero division error when $x = 0$?

(a) `(x == 0 || 3/x > 5)`

and

(b) `(3/x > 5 || x == 0)`

Solution: (a). The expressions are evaluated left-to-right in a lazy fashion. The expression (a) will avoid a zero divison error, because if $x = 0$, then the left side of the disjunction is satisfied and the entire expression will evaluate to true without evaluating the right side of the disjunction.

Name:_____

6. (10 points)

What does the following program print out?

```
class TrickyParameterPassing
  {
  public static void main(String[] argv)
    {
    int i = 3;
    int j = f(i) + g(i);
    System.out.println("The answer is " + j);
    }

  public static int f(int x)
    {
    return x*x;
    }

  public static int g(int x)
    {
    return x + f(x);
    }
  }
```

Solution: It will print out: The answer is 21
This code computes and assigns to j the expression $f(3) + g(3) = 3 \cdot 3 + 3 + 3 \cdot 3 = 9 + 12 = 21$.

Name:_____

7. (10 points)
What does the following code fragment print out?

```
class TrickyFunction
  {
  public static void main(String[] argv)
    {
    int a = 10;
    triple(a);
    System.out.println("The value of a is " + a);
    }

  public static void triple(int a)
    {
    a = 3*a;
    }
 }
```

Solution: It prints out: The value of a is 10
The trick is that the expression triple(a) does evaluate to 30, but it is not assigned to a and the value of a is unchanged by the function call.

Name:_____

8. (10 points)
Write a function called isSquare which has two int parameters a and b and returns a boolean which is true only if one of them is the square of the other (in other words, either $a = b^2$ or $b = a^2$) and otherwise is false.

Solution: Here is one possible solution for this.

```
public static boolean isSquare(int a, int b)
  {
  return (a == b*b || b == a*a);
  }
```

Name:_____

9. (10 points)
What does the following program print out?

```
class TrickyFunctionCall
  {
  public static void main(String[] argv)
    {
    int a = 2;
    cube(a);
    System.out.println("The value of a is " + a);
    }

  public static int cube(int x)
    {
    return x*x*x;
    }
  }
```

Solution: It prints out: The value of a is 2
The trick is that the expression cube(a) does evaluate to 8, but it is not assigned to a and the value of a is unchanged by the function call.

Name:_____

10. (10 points)
What is the value of quizBoolean after the following code fragment executes?

```
boolean quizBoolean;
int a = 100, b = 3, c = 5, d = 9;
quizBoolean = !(b > a || c % d == 0);

{\bf Solution:} The value of quizBoolean is true.\\

The entire expression is a negation of a disjunction.
The expression is true if the disjunction is false. The
first part of the disjunction is false because it is
not true that $3 > 100$. The second part of the disjunction
is false because the remainder of 5 divided by 9 is 5 and
not 0. Since both parts of the disjunction are false, the
disjunction is false. Thus, its negation is true.
```