CSE 142 SPRING 2008 MIDTERM SOLUTIONS

1. Expressions (10 points)

For each expression in the left-hand column, indicate its value in the right-hand column.
Be sure to list a constant of appropriate type (e.g., `7.0` rather than `7` for a `double`, `String`s in "quotes"). If the expression is illegal, then write "error".

Expression	Value
`1 * 2 * 3 + (4 - 5)`	`5`
`28 % 4 + 18 % 5 % 5 + 9`	`12`
`1000 * 2 + 18 / 2 / 2 * 2`	`2008`
`1 / 10.0 + "1" + 17 * 2`	`"0.1134"`
`0.25 * 2 - 0.5 + 1 / 2`	`0.0`

2. Parameter Mystery (20 points)

At the bottom of the page, write the output produced by the following program, as it would appear on the console.

```
public class ParameterMystery {
    public static void main(String[] args) {
        String farm = "here";
        String old = "macdonald";
        String macdonald = "there";
        String everywhere = "farm";
        String here = "everywhere";
        String there = "old";
        String quack = "duck";

        mystery(macdonald, there, "everywhere");
        mystery(old, macdonald, farm);
        mystery("quack", here, "there");
        mystery(quack, "here", "farm");
        mystery(old, everywhere, there);
    }

    public static void mystery(String macdonald, String farm, String old) {
        System.out.println(old + " " + macdonald + " had a " + farm);
    }
}
```

```
everywhere there had a old
here macdonald had a there
there quack had a everywhere
farm duck had a here
old macdonald had a farm
```

3. While Loop Simulation (15 points)

For each call below to the following method, write the output that is printed, as it would appear on the console:

```
public static void mystery(int a, int b) {
    while (b < a) {
        b = b + 3;
        a = a - b;

        System.out.print(a + " ");
    }

    System.out.println(b);
}
```

Method Call	Output
`mystery(20, 2);`	`15 7 8`
`mystery(16, -2);`	`15 11 4 7`
`mystery(10, 25);`	`25`
`mystery(3, -1);`	`1 2`
`mystery(15, 3);`	`9 0 9`

4. Assertions (15 points)

For each of the five points labeled by comments, identify each of the following assertions as being either always true, never true or sometimes true / sometimes false.

```
public static void mystery(Scanner input) {
    System.out.print("Enter a positive number: ");
    int num = input.nextInt();
    int z = num;
    int a = 0;

    // Point A
    while (z > 0) {
        // Point B
        if (a < z) {
            // Point C
            a++;
        } else {
            z--;
        }

        num = z + 1;
        // Point D
    }

    // Point E
}
```

	`z > 0`	`num > 0`	`a < z`
Point A	SOMETIMES	SOMETIMES	SOMETIMES
Point B	ALWAYS	ALWAYS	SOMETIMES
Point C	ALWAYS	ALWAYS	ALWAYS
Point D	SOMETIMES	ALWAYS	SOMETIMES
Point E	NEVER	SOMETIMES	NEVER

5. Programming (20 points)

Write a static method named `isBalanced` that accepts a String of parentheses and returns whether the parentheses in the String are balanced or not. To be balanced:

- Every opening parenthesis must have a matching closing parenthesis after ("to the right of") it.
- Every closing parenthesis must have a matching opening parenthesis before ("to the left of") it.

You may assume that the String parameter only has opening and closing parentheses.

Here are some example calls to the method:

Call	Returns
`isBalanced("")`	`true`
`isBalanced("()")`	`true`
`isBalanced("(")`	`false`
`isBalanced(")")`	`false`
`isBalanced("()(())")`	`true`
`isBalanced("(())(())()")`	`true`
`isBalanced(")()(")`	`false`
`isBalanced(")(")`	`false`
`isBalanced("())")`	`false`
`isBalanced("((()))")`	`false`
`isBalanced("(())")`	`true`

Hint: You will want to do some counting.

```
public static boolean isBalanced(String line) {
    int numOpen = 0;
    int numClosed = 0;
    for (int i = 0; i < line.length(); i++) {
        if (line.charAt(i) == '(') {
            numOpen++;
        } else {
            numClosed++;
        }

        if (numClosed > numOpen) {
            return false;
        }
    }

    return (numOpen == numClosed);
}
```

```
public static boolean isBalanced(String line) {
    int numOpen = 0;
    for (int i = 0; i < line.length(); i++) {
        if (line.charAt(i) == '(') {
            numOpen++;
        } else {
            numOpen--;
        }

        if (numOpen < 0) {
            return false;
        }
    }

    return (numOpen == 0);
}
```

6. Programming (20 points)

Write a static method named `adder` that takes a Scanner for the console as input and repeatedly asks the user for two numbers to sum. The method stops asking for numbers when two consecutive sums are the same and prints out how many sums were computed. You may assume that the user will always type exactly two integers per prompt.

The following sample logs of execution show the output produced by the method:

Sample log #1

```
Enter two numbers: 1 2
The sum is: 3
Enter two numbers: 3 4
The sum is: 7
Enter two numbers: 5 6
The sum is: 11
Enter two numbers: 1 2
The sum is: 3
Enter two numbers: 0 3
The sum is: 3
We computed 5 sums.
```

Sample log #2

```
Enter two numbers: 0 0
The sum is: 0
Enter two numbers: -1 1
The sum is: 0
We computed 2 sums.
```

```
public static void adder(Scanner input) {
    System.out.print("Enter two numbers: ");
    int sum1 = input.nextInt() + input.nextInt();
    System.out.println("The sum is: " + sum1);

    System.out.print("Enter two numbers: ");
    int sum2 = input.nextInt() + input.nextInt();
    System.out.println("The sum is: " + sum2);

    int count = 2;
    while (sum1 != sum2) {
        sum1 = sum2;

        System.out.print("Enter two numbers: ");
        sum2 = input.nextInt() + input.nextInt();
        System.out.println("The sum is: " + sum2);

        count++;
    }

    System.out.println("We computed " + count + " sums.");
}
```

(More possible solutions on the next page.)

```
public static void adder(Scanner input) {
    System.out.print("Enter two numbers: ");
    int sum1 = input.nextInt() + input.nextInt();
    System.out.println("The sum is: " + sum1);

    int sum2 = sum1 + 1; // sum1 and sum2 have to be different to enter the loop

    int count = 1;
    while (sum1 != sum2) {
        sum2 = sum1;

        System.out.print("Enter two numbers: ");
        sum1 = input.nextInt() + input.nextInt();
        System.out.println("The sum is: " + sum1);

        count++;
    }

    System.out.println("We computed " + count + " sums.");
}
```

```
public static void adder(Scanner input) {
    System.out.print("Enter two numbers: ");
    int sum1 = input.nextInt() + input.nextInt();
    System.out.println("The sum is: " + sum1);

    int sum2;
    int count = 1;
    do {
        sum2 = sum1;

        System.out.print("Enter two numbers: ");
        sum1 = input.nextInt() + input.nextInt();
        System.out.println("The sum is: " + sum1);

        count++;
    } while (sum1 != sum2);

    System.out.println("We computed " + count + " sums.");
}
```