CSE 142, Autumn 2008
Midterm Exam Key

1. Expressions

Expression	Value
`-(6 + 3 - 2 * 3)`	`-3`
`15 % 6 + 5 % 5 + 12 % 7 % 3`	`5`
`9 / 2 / 2.0 + 9 / 2.0 / 2`	`4.25`
`5 + 6 + "7" + 8 + 9 + 2 * 3`	`"117896"`
`!(3 < 7 && -1 != 8)`	`false`

2. Parameter Mystery

```
-1 + 2 + j
2 + 7 + 5
4 + 2 + y
-1 + 0 + 51
```

3. If/Else Simulation

Method Call	Output
`ifElseMystery(10, 3);`	`0 3`
`ifElseMystery(6, 6);`	`6 7`
`ifElseMystery(3, 4);`	`-4 4`
`ifElseMystery(4, 20);`	`8 21`

4. While Loop Simulation

Method Call	Output
`whileMystery(0, 5, 10);`	`5 10 15 20`
`whileMystery(4, 2, 12);`	`6 4 10 8 18 16`
`whileMystery(10, 1, 14);`	`11 2 13 4 17 8 25 16`

5. Assertions

	`steps > 0`	`x > 0`	`r > 0`
Point A	SOMETIMES	NEVER	SOMETIMES
Point B	SOMETIMES	SOMETIMES	SOMETIMES
Point C	SOMETIMES	ALWAYS	NEVER
Point D	SOMETIMES	NEVER	ALWAYS
Point E	NEVER	ALWAYS	SOMETIMES

6. Programming

There are many ways to solve any programming problem. Here are some common correct solutions we saw:

```
public static void hopscotch(int hops) {
    System.out.println("    1");
    for (int i = 1; i <= hops; i++) {
        System.out.println((3 * i) + "      " + (3 * i + 1));
        System.out.println("    " + (3 * i + 2));
    }
}

public static void hopscotch(int hops) {
    for (int i = 0; i < hops; i++) {
        System.out.println("    " + (3 * i + 1));
        System.out.println((3 * i + 2) + "      " + (3 * i + 3));
    }
    System.out.println("    " + (3 * hops + 1));
}

public static void hopscotch(int hops) {
    for (int i = 1; i <= hops * 3 + 1; i++) {
        if (i % 3 == 1) {
            System.out.println("    " + i);
        } else if (i % 3 == 2) {
            System.out.print(i);
        } else {
            System.out.println("      " + i);
        }
    }
}

public static void hopscotch(int hops) {
    System.out.println("    1");
    int num = 2;
    for (int i = 1; i <= hops; i++) {
        System.out.println(num + "      " + (num + 1));
        System.out.println("    " + (num + 2));
        num = num + 3;
    }
}
public static void hopscotch(int hops) {
    System.out.println("    1");
    int num = 2;
    while (num <= 3 * hops + 1) {
        System.out.print(num);
        num++;
        System.out.println("      " + num);
        num++;
        System.out.println("    " + num);
        num++;
    }
}
public static void hopscotch(int hops) {
    int count = 1;
    for (int i = 1; i <= 2 * hops + 1; i++) {
        if (i % 2 == 1) {
            System.out.println("    " + count);
            count++;
        } else {
            System.out.println(count + "      " + (count + 1));
            count = count + 2;
        }
    }
}
public static void hopscotch(int hops) {
    int num = 1;
    System.out.println("    " + num++);
    for (int i = 1; i <= hops; i++)
        System.out.println(num++ + "      " + num++ + "\n    " + num++);
}
```

7. Programming

```
public static boolean containsBothDigits(int a, int b, int c) {
    int bCount = 0;
    int cCount = 0;
    while (a != 0) {
        int digit = a % 10;
        a = a / 10;
        if (digit == b) {
            bCount++;
        }
        if (digit == c) {
            cCount++;
        }
    }

    if (bCount > 0 && cCount > 0) {
        return true;
    } else {
        return false;
    }
}

public static boolean containsBothDigits(int a, int b, int c) {
    boolean foundB = false;
    boolean foundC = false;
    while (a != 0) {
        int digit = a % 10;
        a = a / 10;
        if (digit == b) {
            foundB = true;
        }
        if (digit == c) {
            foundC = true;
        }
    }

    return foundB && foundC;
}

public static boolean containsBothDigits(int a, int b, int c) {
    int copyA = a;
    int count = 0;

    while (a != 0) {            // look for b
        if (a % 10 == b) {
            count++;
        }
        a = a / 10;
    }

    a = copyA;                    // restore a
    while (a != 0) {            // look for c
        if (a % 10 == c && count > 0) {
            return true;
        }
        a = a / 10;
    }

    return false;
}
```

```
public static boolean containsBothDigits(int a, int b, int c) {
    int count = 0;
    while (a != 0) {
        int digit = a % 10;
        a = a / 10;
        if (digit == b) {
            count++;
            b = -1;    // so that it won't be counted twice
        }
        if (digit == c) {
            count++;
            c = -1;    // so that it won't be counted twice
        }
    }

    if (count == 2) {
        return true;
    } else {
        return false;
    }
}

public static boolean containsBothDigits(int a, int b, int c) {
    boolean foundB = false, foundC = false;
    while (a != 0) {
        foundB = foundB || a % 10 == b;
        foundC = foundC || a % 10 == c;
        a /= 10;
    }
    return foundB && foundC;
}

public static boolean containsBothDigits(int a, int b, int c) {
    return containsDigit(a, b) && containsDigit(a, c);
}
public static boolean containsDigit(int a, int b) {
    while (a != 0) {
        if (a % 10 == b) {
            return true;
        }
        a = a / 10;
    }
    return false;
}
```