COS 126 **General Computer Science** **Spring 2005**

Midterm 1 Solutions

1. **Short Answer (7 points)**

   (a) The = operator is used for assignment.
   The == operator is used for testing equality.

   (b) boolean

   (c) True

   (d) 1.0

   (e) 48A

   (f) 10010001010

   (g) 2212

2. **Arrays, Conditionals, Loops, and Bugs (6 points)**

   Corrections are noted in on-line comments.

```
public static void sort(int[] a) {
    int N = a.length;   \\ Add this line to declare and initialize N.
    for (int i = 0; i < N; i++) {  \\ Add this left curly brace.
       // find ith smallest
       int mini = i;               \\ Start mini at i each iteration.
       for (int j = i; j < N; j++)
          if (a[j] < a[mini])
             mini = j;

       // swap entries to put the ith smallest in the ith position
       // now a[0] ... a[i] have the correct entries
       int tmp = a[i];             \\ Add this line.  tmp needed for swap.
       a[i] = a[mini];
       a[mini] = tmp;              \\ Change a[i] to tmp.  Add semi-colon.
    }                              \\ Add this right curly brace.
}
```

3. **Recursion (6 points)**

(a) The output of `java Quest 11` is 4.
The calls are

```
BS(11, a, 0, 7)
   m = 3
   BS(11, a, 4, 7)
      m = 5
      BS(11, a, 4, 5)
         m = 4, since a[4] is 11, return 4
```

(b) The output of `java Quest 12` is -1. The calls are

```
BS(12, a, 0, 7)
   m = 3
   BS(12, a, 4, 7)
      m = 5
      BS(12, a, 4, 5)
         m = 4
         BS(12, a, 5, 5)
            m = 5, since i==j return -1
```

(c) If it finds the number x in the sorted array, BS returns the appropriate index, otherwise it returns -1. BS stands for Binary Search which is a process of splitting a sorted list in half and looking in the lower or upper half.

4. **Combinational Logic (8 points)**

(a) Truth table for the requested function f, as well as the decimal equivalent of X:

abc	decimal	f
000	0	0
001	1	0
010	2	0
011	3	1
100	-4	1
101	-3	1
110	-2	0
111	-1	0

(b) f = a'bc + ab'c' + ab'c

(c) The circuit has three NOT gates, to provide a', b', and c'.
Three 3-input AND gates have inputs matching the terms in the equation, and output going to a 3-input OR gate, whose output is f.

(d) The term ab'c' + ab'c is independent of c, and thus equivalent to ab'.
The simplified equation is f = a'bc + ab'.
The simplified circuit has two NOT gates, two AND gates, one OR gate.

5. **Java Programming and Graphics (9 points)**

(a) Write the **drawFilledCircle** method.

```
/* The drawFilledCircle method is a helper method for drawBullseye.
   The center, radius, and fill color of the circle are specified. */

public static void drawFilledCircle(double x, double y, double radius,
   Color color) {

       StdDraw.penUp();
       StdDraw.go(x, y);
       StdDraw.setColor(color);
       StdDraw.fillOn();
       StdDraw.spot(2*radius);
   }
```

(b) Write the **drawBullseye** method.

```
/* A bullseye consists of alternating bands of dark and white concentric rings.
   The center (x, y) of the bullseye is specified as well as the outermost
   radius (outerR), thickness of each ring (ringThickness) and dark band
   color (color).  The outermost ring is always dark.
   The innermost ring may have a smaller thickness if necessary.
   Use the drawFilledCircle helper function to simplify your code. */

public static void drawBullseye(double x, double y, double outerR,
   double ringThickness, Color color) {
       double radius = outerR;
       while(radius > 0){
           drawFilledCircle(x, y, radius, color);
           radius -= ringThickness;
           if(radius > 0) drawFilledCircle(x, y, radius, Color.white);
           radius -= ringThickness;
       }
   }
```

Another valid solution is to have a loop and a counter keeping track of which color should be drawn. When the counter is even, draw a filled color circle. When the counter is odd, draw a white circle.

6. **TOY Programming (6 points)**

```
// register map:  Add more if needed
// R[A] - a
// R[B] - b
// R[C] - temp
// R[2] - scratch pad register
```

Addr:	Instruction	Comments
10:	8AFF	Read a from stdin
11:	8BFF	Read b from stdin
12:	CB1A	if (b==0) go output the gcd
13:	22AB	R[2] = R[A] - R[B] set up (a-b)
14:	D218	if a greater than b don't swap
15:	1C0B	Swap: temp = b
16:	1B0A	b = a
17:	1A0C	a = temp
18:	2AAB	a = a-b
19:	C012	go to 12
1A:	9AFF	Write a to stdout
1B:	0000	halt
1C:		
1D:		

The swap requires the transfer of data between registers. There are many ways to move data between registers. Add R[0] was used in the solution above. XOR R[0] is another option. Storing the data in memory and then re-loading the registers is another (slightly less efficient) way. There are others as well.

7. **TOY Architecture (6 points)**

```
The following TOY instructions will still work:
   0 Halt
   7 Load address
   8 Load
   A Load Indirect
   F Jump and link

Arithmetic and Logical operations (1, 2, 3, 4, 5, 6) will not work because
they all need s to enter A Addr.

Store instructions (9, B) will not work because both need d to enter A Addr.

Of the Control instructions, the conditional branches (C, D) and jump
register (E) will not work because they need d to enter A Addr.
```