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

Midterm 1

This test has 8 questions worth a total of 50 points. You have 120 minutes. The exam is closed book, except that you are allowed to use a one page cheatsheet, one side only, handwritten by you. No calculators or other electronic devices are permitted. Give your answers and show your work in the space provided. **Write out and sign the Honor Code pledge before turning in the test.**

"I pledge my honor that I have not violated the Honor Code during this examination."

Signature

Problem	Score	Possible
0		2
1		7
2		6
3		6
4		8
5		9
6		6
7		6
Total		50

Name:

NetID:

Preceptor:	Dan	Mona
	Phil	Benedict
	Zafer	Donna

The TOY instruction cheatsheet is on the last page of the exam.
There is also a list of StdDraw methods on the last page of the exam.
Feel free to tear out the last page in order to use these references more easily.

0. **Miscellaneous. (2 points) (really)**

   (a) Write your name and Princeton NetID in the space provided on the front of the exam, and circle the name of the preceptor who grades your homework assignments.

   (b) *Write* and sign the honor code on the front of the exam.

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

   (a) In the Java programming language, what is the difference between the = operator and the == operator?

   (b) What Java data type has only two possible values?

   (c) True or False. Any `for` loop can be converted into an equivalent `while` loop.

   (d) What is the result of the Java expression $(0.5 * (10/4))$? Circle your answer.

   (e) Convert the decimal (base-10) number 1162 to hexadecimal (base-16). Circle your answer.

   (f) Convert decimal (base-10) number 1162 to binary (base-2). Circle your answer.

   (g) Assume an octal (base-8) number system. It has 8 digits, 0,1,2,3,4,5,6,7. Represent the decimal (base-10) number 1162 as an octal (base-8) number.

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

   The following method is supposed to sort the elements of an array from smallest to largest. If passed the array {3,1,2}, the correct result is the array {1,2,3}. Unfortunately, it has compile-time bugs and run-time bugs. Find the bugs and rewrite the code so that it runs correctly. You do not need to rewrite the comments.

```
public static void sort(int[] a) {

   for (int i = 0; i < N; i++)

      // find ith smallest
      int mini = 0;
      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
      a[i] = a[mini];
      a[mini] = a[i]

}
```

3. **Recursion (6 points)**

```
public class Quest{

    public static int BS(int x, int[] a, int i, int j){
       int m = (i + j) / 2;
       if (a[m] == x) return m;
       if (i >= j) return -1;
       if (a[m] > x) return BS(x, a, i, m);
       else return BS(x, a, m + 1, j);
    }

    public static void main(String[] args){
       int[] array = {2, 3, 5, 7, 11, 13, 17, 19};
       int N = Integer.parseInt(args[0]);
       System.out.println(BS(N, array, 0, 7));
    }
}
```

(a) What is the output when the above program is run with the command

```
java Quest 11
```

(b) What is the output when the above program is run with the command

```
java Quest 12
```

(c) What does the BS method accomplish? (i.e., In general, what is it doing?)

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

   Consider a **3-bit** binary number X represented in 2's complement format.

   (a) Write out the truth table for the following Boolean function of X: the absolute value of X is greater than 2

   (b) Write out the sum-of-products form of this Boolean function.

   (c) Now, using AND, OR, and NOT gates, draw a combinational circuit of the same function. Your AND and OR gates may have any number of inputs.

   (d) For a tiny bit of extra credit, simplify your Boolean function and the circuit.

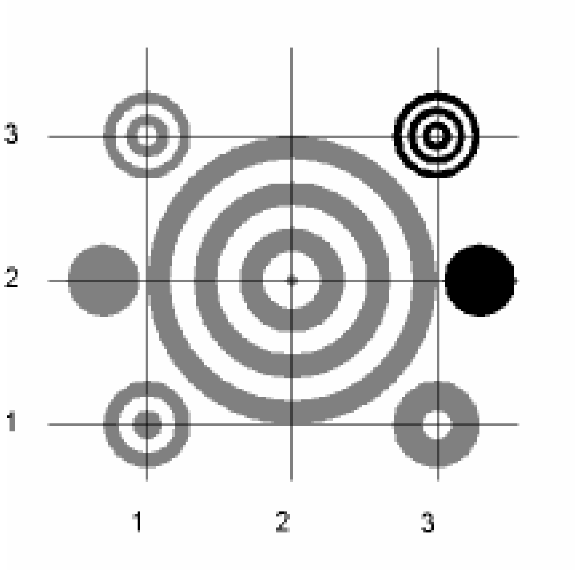
5. **Java Programming and Graphics (9 points)**

You have been using the StdDraw library for creating graphics images. Your task is to create functions to draw a bullseye pattern using StdDraw to perform the most basic drawing tasks.

We provide an example program that calls the drawBullseye method with several different parameters and the image that results. To simplify the drawBullseye function, you will also write a drawFilledCircle function that drawBullseye will call. See the comment with each function for more details about what you should implement.

A list of some of the StdDraw functions is also provided for your reference.

```
public class Graphics {
   public static void main(String[] args) {

      // SEE image example below

      int N = 4;
      StdDraw.create(512, 512);
      StdDraw.setScale(0, 0, N, N);

      // example of drawFilledCircle
      drawFilledCircle(.7, 2, .25, Color.gray);
      drawFilledCircle(3.3, 2, .25, Color.black);

      // example of drawBullseye
      drawBullseye(1, 3, .3, .075, Color.gray);
      drawBullseye(1, 1, .3, .1, Color.gray);
      drawBullseye(2, 2, 1, .16, Color.gray);
      drawBullseye(3, 3, .3, .05, Color.black);
      drawBullseye(3, 1, .3, .2, Color.gray);

      StdDraw.show();
   }
```



Grid lines were added for your reference.

(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) {









}
```

(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) {









}
```

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

The greatest common divisor, or gcd, of two positive integers is the largest integer which evenly divides both of them. The following Java program uses a non-recursive version of Euclid's algorithm to compute the gcd of two positive integers which it reads from standard input. The result is written to standard output.

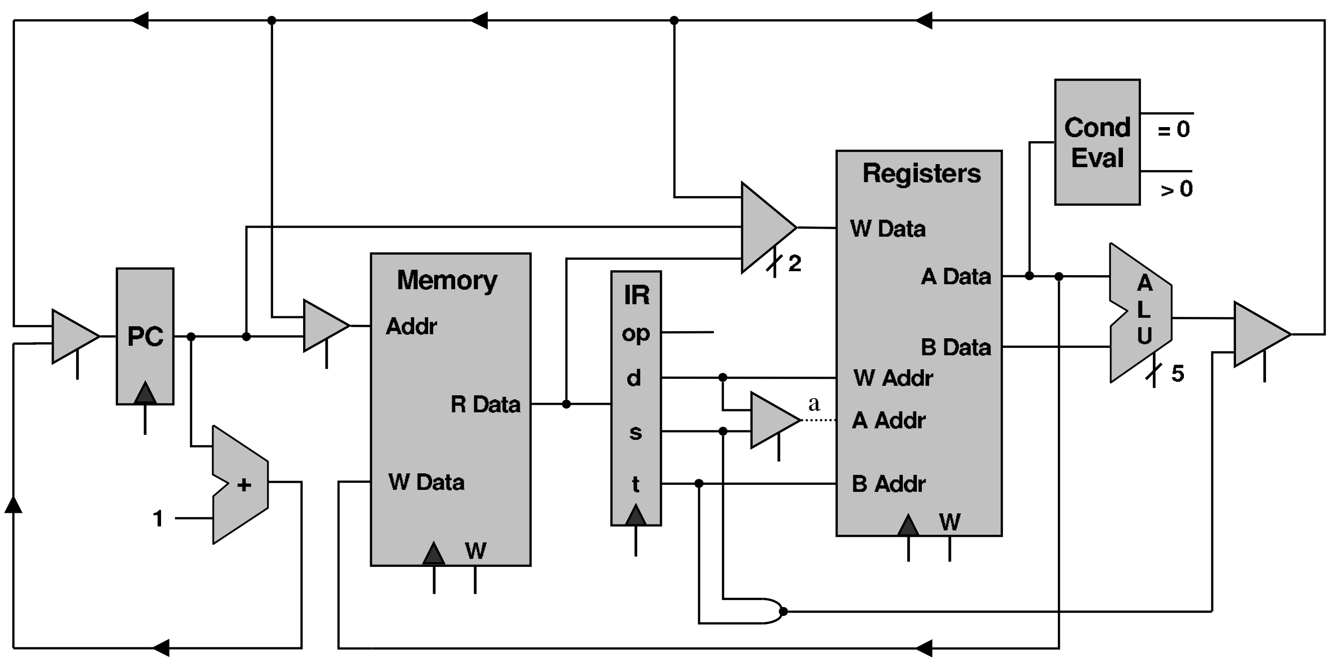
```
public class gcd2 {
 public static void main(String[] args) {
  // read in a and b
  int a = StdIn.readInt();
  int b = StdIn.readInt();

  // subtract the smaller argument
  // from the larger until the smaller
  // argument is 0, then  the larger
  // one will be the gcd
  while (b > 0) {
    // swap a and b if b >= a
    if (b >= a) {
      int temp = b;
      b = a;
      a = temp;
    }

    a = a - b;
  }

  // print out gcd
  System.out.println(a);
 }
}
```

Write a TOY version of this program, including comments. The program should start at memory address 10. Write your final version in the table below. There are more than enough lines for your TOY program in the table. Place your variables in the registers specified in the register map.

```
// 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:		Read b from stdin
12:		if (b==0) go output the gcd
13:		
14:		
15:		
16:		
17:		
18:		
19:		
1A:		
1B:		
1C:		
1D:		

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



Which TOY instructions will still work even if connection a (the dotted line from the mux to A Addr) is severed?

48

TOY REFERENCE CARD

```
  INSTRUCTION FORMATS
           | . . . . | . . . . | . . . . | . . . . |
Format 1:  | opcode  |    d    |    s    |    t    | (0-6, A-B)
Format 2:  | opcode  |    d    |       addr        | (7-9, C-F)
  ARITHMETIC and LOGICAL operations
1:  add          R[d] <- R[s] + R[t]
2:  subtract     R[d] <- R[s] - R[t]
3:  and          R[d] <- R[s] & R[t]
4:  xor          R[d] <- R[s] R̂[t]
5:  shift left   R[d] <- R[s] << R[t]
6:  shift right  R[d] <- R[s] >> R[t]
  TRANSFER between registers and memory
7:  load address    R[d] <- addr
8:  load            R[d] <- mem[addr]
9:  store           mem[addr] <- R[d]
A:  load indirect   R[d] <- mem[R[t]]
B:  store indirect  mem[R[t]] <- R[d]
  CONTROL
0:  halt             halt
C:  branch zero      if (R[d] == 0) pc <- addr
D:  branch positive  if (R[d] > 0) pc <- addr
E:  jump register    pc <- R[d]
F:  jump and link    R[d] <- pc; pc <- addr
  Register 0 always reads 0.
Loads from mem[FF] come from stdin.
Stores to mem[FF] go to stdout.
```

Abbreviated StdDraw Reference

COMMAND	ARGUMENTS	ACTION

`create`	`int w`
	`int h`
		Specifies the dimensions of the drawing canvas in pixels. Typical values for `w` and `h` are 512 and 512.

`penUp`		Lift the pen up.
`penDown`		Put the pen down.
`fillOn`		Set the fill mode to on.
`fillOff`		Set the fill mode to off.

`go`	`double x`
	`double y`
		Make the current location (x, y). If the pen is down, draw the line from the old location to the current location. If the pen is up, don't draw anything.

`spot`		Draws one pixel at the current location in the current foreground color.
`spot`	`double d`	Draws a circular spot of diameter d, centered on the current location, using the current foreground color. Fill in the circle according to the fill mode.
`setColor`	`Color c`	Sets the foreground color to c.

`setColorRGB`	`int r`
	`int g`
	`int b`
		Sets the background color to the RGB color (r, g, b), where r, g, and b are integers between 0 and 255.

`clear`	`Color c`	Sets the background color to c and clears the screen using this color. Typical values are `Color.black` and `Color.blue`. Here is more information on using colors in Java.
`rotate`	`double d`	Rotate the orientation d degrees counterclockwise. All subsequent `goForward` commands will be with respect to this orientation. Any images plotted using `StdDraw.spot` will also be rotated accordingly.
`goForward`	`double s`	Go forward in the current direction s steps, drawing the line from the current location to the new location if the pen is down.
`show`		Display the image on the screen.

`setScale`	`double xmin`
	`double ymin`
	`double xmax`
	`double ymax`
		Rescale the coordinates so that the window goes from (xmin, ymin) to (xmax, ymax).