CIS 110 — Introduction to Computer Programming
Spring 2014 — Midterm

1.) MISCELLANEOUS (9 pts total)

1.1) (1 point) The Easy One:

- Check that your exam has all 9 pages (excluding the cover sheet and scratch paper).
- Write your name, recitation number, and PennKey (username) on the front of the exam.
- Sign the certification that you comply with the Penn Academic Integrity Code.

1.2) (2 points) What would be the result of the following code fragment?

```
ellipseMode(CENTER);
ellipse(width/2, height/4, width, height/2);
```

A. an ellipse that covers the top half of the window

B. an ellipse that covers the bottom half of the window

C. an ellipse that covers the left half of the window

D. an ellipse that covers the entire window

1.3) (2 points) What would the window look like as a result of the following code fragment?

```
stroke(0);
line(0, 0, width, height);
background(255);
```

A. a diagonal black line on a white background

B. a vertical black line on a white background

C. a horizontal black line on a white background

D. a white background

1.4) (2 points) How many times would this code fragment print the text "Hello World"?

```
for (int i = 10; i > 0; i -= 3) {
    System.out.println("Hello World");
}
```

A. Three (3)

B. Four (4)

C. Five (5)

D. None, this is invalid Java code

1.5) (2 points) `!(!(110 > 2014) && !!(true || false) || false)`

A. True B. False

2.) OPERATORS AND EXPRESSIONS (7 points total)

For each code fragment, (a) fill in the most appropriate data type in the 1st column and (b) give the value that `z` contains after the code has been executed in the 2nd column.

If the code would result in an error, write "ERROR" in the 1st column and give the reason for the error in the 2nd column (you do not need to write the exact error message). The first two problems have been completed for you.

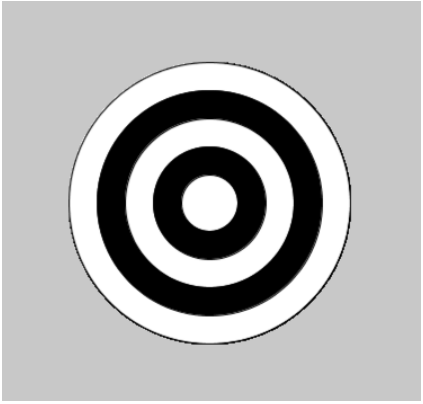
`____ z = 11;` `z++;`	int	12
`int x = 100;` `____ z = x.length;`	ERROR	x doesn't have an x.length field
`____ z = 12.5f;` `z / 2;`	float OR ERROR	12.5 or 12.5f OR Invalid java statement
`int y = 0, z = 1;` `____ z = (y > 0 || z < 0);`	ERROR	z is already defined as an int

`String s = "Life = ";`
`s += 4 + 2;`
`____ z = s;`
	String	"Life = 42"

`int x = 14, y = 4;` `____ z = x % y;`	int	2
`int[] a = {1, 2, 3, 4, 5};` `____ z = a.length / 10;`	int	0
`String s = "Apple";` `____ z = s.charAt(0) + 2;`	char	'C'
`____ z = 9++ + 10;`	ERROR	Cannot increment (++) a constant (9)

3.) CONDITIONALS AND LOOPS (12 points total)

3.1) (6 points) The code fragment below contains four blanks, labeled A, B, C, D. Fill in the blanks to make the bullseye image shown on the right:



```
public void setup() {
  size(640, 640);
  background(200);
  for (int i =   A   ;   B   ;   C   ){
    if (   D   != 0) {
      fill(255);
    } else {
      fill(0);
    }
    ellipse(width / 2, height / 2, 50 * i, 50 * i);
  }
}
```

Write your answers below:

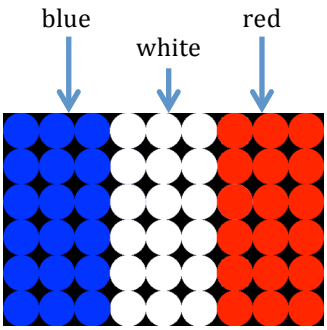
Blank A: 5

Blank B: i > 0

Blank C: i--

Blank D: i % 2

3.2) (6 points) France's flag contains three vertical pales: blue fills the left third, white fills the center, and red fills the right third. This code fragment was written by an artist to form an abstract rendition of France's flag from colored circles, finishing with the image to the right.




```
public void setup() {
  size(450, 300);
  background(0);
  noStroke();

  for (int x = 0; x < width; x++) {
    for (int y = 0; y < height; y++) {
      if (x > 0.67 * width) {
        fill(255, 0, 0);
      }
      if (x > 0.33 * width) {
        fill(255);
      }
      if (x > 0.0){
        fill(0, 0, 255);
      }
      ellipse(x, y, 50, 50);
    }
  }
}
```

Circle the correct answers:

The code fragment will draw the abstract French flag image shown to the right.

A. True **B. False**

The code will draw a flag with _____ colors (ignoring the black background).

A. 1 B. 2 C. 3

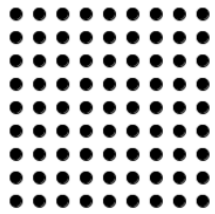
When finished, the image will be formed of multiple distinct circles on a black background.

A. True **B. False**

4.) WHODUNNIT? (12 points total)

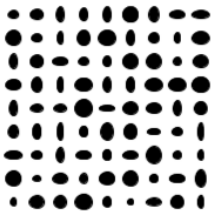
Match each image with the code that was most likely used to draw it. Write the letter of the code fragment on the blank line above the corresponding image. If none of the code fragments could have generated that image, write "None". Each code fragment should be used at most once.

4.1) ____C_____

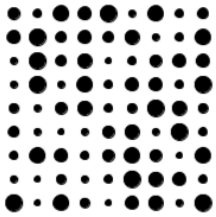


4.2) ____B_____

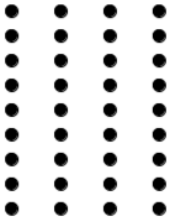
4.3) ____D_____



4.4) ___F_____



4.5) ____E_____



4.6) ___A_____

(A)

```
public void draw() {
  background(255);
  fill(0);
  for (int x = 20; x <= width - 20; x += 20)
    for (int y = 20; y <= height - 20; y += 20) {
      if (x < y)
         ellipse(x, y, 10, 10);
    }
}
```

(B)

```
public void draw() {
  background(255);
  fill(0);
  for (int x = 20; x <= width - 20; x += 20)
    for (int y = height - 20; y >= 20; y -= 20) {
      if (x == y)
        ellipse(x, y, 10, 10);
    }
}
```

(C)

```
public void draw() {
  background(255);
  fill(0);
  for (int x = 20; x <= width - 20; x += 20)
    for (int y = 20; y <= height - 20; y += 20)
      ellipse(x, y, 10, 10);
}
```

(D)

```
public void draw() {
  background(255);
  fill(0);
  for (int x = 20; x <= width - 20; x += 20)
    for (int y = height - 20; y >= 20; y -= 20)
       ellipse(x, y, random(5,15), random(5,15));
  noLoop();
}
```

(E)

```
public void draw() {
  background(255);
  fill(0);
  for (int x = 20; x <= width - 20; x += 20)
   for (int y = 20; y <= height - 20; y += 20) {
      if (x % 40 == 0)
        ellipse(x, y, 10, 10);
      else if (y < 0)
         rect(x, y, 10, 10);
   }
}
```

(F)

```
public void draw() {
  background(255);
  fill(0);
  for (int x = 20; x <= width - 20; x += 20)
    for (int y = height - 20; y >= 20; y -= 20) {
      int i = (int) random(5, 15);
      ellipse(x, y, i, i);
    }
  noLoop();
}
```

5.) TRACERY (10 points total)

Trace through the following code. Assume that the program is executed using the command:

```
java Tracery 3 5
```

Whenever you reach a `println()` statement, write the values of the three variables `a`, `b`, `c` in the table to the right as they would be printed. You may not need to use all of the table's rows.

```
public class Tracery {
  static int a = 0, b = 0;
  static int c = 0;

  static int f1(int a, boolean b) {
    if (b) {
      c = 2 * a;
    } else {
      a *= a; // square a
      System.out.println(a + " " + b + " " + c);
      return -2 * (int) f2(a, c);
    }
    return c;
  }

  static int f2(int b, int c) {
    int a = (b + c) / b;
    return a;
  }

  public static void main(String[] args) {
    a = Integer.parseInt(args[0]);
    b = Integer.parseInt(args[1]);
    System.out.println(a + " " + b + " " + c);

    b = f1(a, true);
    System.out.println(a + " " + b + " " + c);

    b = f1(a / 2, false);
    System.out.println(a + " " + b + " " + c);
  }
}
```

A	B	C
3	5	0
3	6	6
1	false	6
3	-14	6

6.) DEBUGGING (10 points total)

How many random people do you have to ask before you find two who have the same birthday (month and day)? (Would it surprise you to learn that the answer is approximately 25?) The following program tries to answer this problem, but it contains a number of bugs. Find and correct the bugs, filling in the table on the next page.

```
01 /** Simulate multiple experiments of the birthday problem.
02  * Each experiment simulates asking people at random and stops
03  * after it finds two who were born on the same day.
04  * Usage:  java BirthdayProblem n
05  *    n - the number of experiments to run
06  */
07 public class BirthdayProblem {
08
09   static int count = 0
10
11   public static void main(String[] args) {
12       int n = Integer.parseInt(args);
13       double sum = 0;
14       for (i = 0; i < n; i++) {
15          double sum += birthdayProblemExperiment();
16       }
17       System.out.println("Average number of people we need to ask " +
18                          "before finding a duplicate birthday: " +
19                          sum / n);
20   }
21
22   // Simulates choosing random people and asking their birthday.
23   // Birthdays can be on the 0th to 365th day of the year (includes
24   // leap days).  Stops when it finds a duplicate birthday.
25   static int birthdayProblemExperiment() {
26      boolean[] seenDate = boolean[366];  // all entries start as false
27
28      while (true) {
29         // choose a random birthday 0-365
30         int birthday = (Math.random() * 366);
31         count++;
32         if (seenDate[birthday])
33            break;
34         seenDate[birthday] = true;
35      }
36
37      System.out.println("Found a duplicate birthday after " +
38                         count + " tries.");
39
40   }
41 }
```

The first bug has been found for you, and is listed as an example in the table.

Hint: The program contains 6 more syntax bugs that prevent the code from compiling and 1 more logical bug that will prevent the program from giving the correct answer. Each correction should require one line of code or less.

6.1 (6 points) List the six syntax errors in the code (excluding the missing semicolon on line 9).

Line Number	Error	Correction
09	Missing a semicolon	Add semicolon at end of line
12	Missing index into `args`	Change `args` to `args[0]`
14	Missing data type on `i`	Change `i = 0` to `int i = 0`
15	Extra declaration of `sum` variable	Delete `double`
26	Missing `new` keyword	Change `boolean[366]` to `new boolean[366]`
30	Missing cast	Change `(Math.random() * 366)` to `(int)(Math.random() * 366)`
39	Missing return statement	Insert line: `return count`

6.2) (4 points) What is the logical error in the program above that will prevent it from giving the correct answer? How can you fix the logical error?

The logical bug is that the value of `count` persists between subsequent calls to `birthdayProblemExperiment()`, so this function will return the wrong value after the first experiment. To fix it, we can either make `count` a local variable in `birthdayProblemExperiment()`, or can make line 27 to be `count = 0`.

7.) FUNCTIONS! (22 points total)

7.1) (10 points) Write the static function `multiples()` based on the function header below:

```
/** Function Name:  multiples
 *  Parameters:
 *    x - an integer
 *    arr - an array of integers
 *  Returns:
 *    the number of elements in arr that are evenly divisible by x
 */
static int multiple(int x, int[] arr) {
  int count = 0;
  for (int i = 0; i < arr.length; i++) {
    if (arr[i] % x == 0) {
      count++;
    }
  }
  return count;
}
```

7.2) (12 points) Write the static function `interleave()` based on the function header below:

```
/** Function Name:  interleave
 *  Parameters:
 *    arr1 - an array of integers of length n
 *    arr2 - an array of integers of length n
 *  Returns:
 *    an array containing the result of interleaving arr1 and arr2
 *  Error checking:
 *    if the arrays are of different lengths or null, return null
 *  Examples: interleave({1, 3, 5}, {2, 4, 6})  ->  {1, 2, 3, 4, 5, 6}
 *            interleave({1, 3, 5}, {2})        ->  null
 */
static int[] interleave(int[] arr1, int[] arr2) {
  // error checking
  if (arr1 == null || arr2 == null || arr1.length != arr2.length) {
    return null;
  }
  // interleave arrays
  int n = arr1.length;
  int[] result = new int[2 * n];
  for (int i = 0; i < n; i++) {
    result[2 * i]     = arr1[i];
    result[2 * i + 1] = arr2[i];
  }
  return result;
}
```

8.) SORTING AND SEARCHING (8 pts total)

8.1) (2 points) Which algorithm has the slowest runtime (i.e., largest computational complexity) in the best case?

A. linear search in an array

B. binary search in an array

C. finding the maximum value in an array

D. printing the last entry in an array

8.2) (2 points) Which algorithm has the fastest runtime (i.e., smallest computational complexity) in the worst case?

A. linear search in an array

B. binary search in an array

C. finding the maximum value in an array

D. selection sort of an array

Recall that both insertion sort and selection sort separate the array into two portions: the left portion is always in ascending sorted order, and the right portion is not. Each step of the algorithm shifts the boundary between these portions one array position to the right, sorting as it goes.

8.3) (2 points) Assume that we're using insertion sort on the array `{4, 1, 9, 6, 3, 0}`. What would be the state of the array after one step of insertion sort?

A. `{1, 4, 6, 9, 3, 0}`

B. `{0, 4, 1, 9, 6, 3}`

C. `{0, 1, 9, 6, 3, 4}`

D. `{1, 4, 9, 6, 3, 0}`

8.4) (2 points) Assume that we're using selection sort on the array `{8, 9, 0, 6, 1, 4}`. What would be the state of the array after TWO steps of selection sort?

A. `{8, 9, 0, 6, 1, 4}`

B. `{0, 8, 9, 6, 1, 4}`

C. `{0, 1, 8, 9, 6, 4}`

D. `{0, 1, 8, 6, 9, 4}`