COS 126	General Computer Science	Spring 2009

Exam 1

This test has 6 questions worth a total of 50 points. You have 50 minutes. The exam is closed book, except that you are allowed to use a one page cheatsheet. 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."

Name: _____

Signature

NetID:

Problem	Score
0	
1	
2	
3	
4	
5	
Total	

P01	TTh 1:30	Will
P01A	TTh 1:30	Rob
P01B	TTh 1:30	Aditya
P01C	TTh 1:30	Michael
P02	TTh 2:30	Will
P03	TTh 3:30	Rob
P04	TTh 7:30	Chris
P05	WF 10	JP
P06	WF 1:30	Chris
P06A	WF 1:30	Thomas
P06B	WF 1:30	Donna
P06C	WF 1:30	Michael

0. **Miscellaneous. (1 point)**

   (a) Write your name and Princeton NetID in the space provided on the front of the exam, and circle your precept number.

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

1. **Number systems. (8 points)**

   (a) What is the decimal representation of the *16-bit* two's complement integer 101100_2? Circle your answer.

   (b) What is the decimal representation of the *6-bit* two's complement integer 101100_2? Circle your answer.

   (c) Write the decimal integer -77 in TOY (*16-bit* two's complement integer, in hexadecimal). Circle your answer.

   (d) Let `a` be a Java variable of type `int`. For which of the following values of `a` does the expression `2 ^ (2 ^ a)` not equal `(2 ^ 2) ^ a`? Circle the best answer.

   1 2 5 32 equal for all values

2. **Java basics. (15 points)**

(a) Assume that `a`, `b`, and `c` are of variables of type `int`. Consider the following three conditions.

I. `(a == b) && (a == c) && (b == c)`

II. `(a == b) || (a == c) || (b == c)`

III. `(a - b) * (b - c) * (a - c) == 0`

Which of the conditions above is (are) always true if at least two of `a`, `b`, and `c` are equal?

I only

I and II only

II only

II and III only

III only

(b) Consider the following function (static method).

```
public static int f(int a, int b, int c) {
    if ((a <  b) && (b <  c)) return a;
    if ((a >= b) && (b >= c)) return b;
    if ((a == b) || (b == c) || (a == c)) return c;
}
```

Which of the following best describes why this function does not compile?
Circle the best answer.

i. The `if` statement must have `else` parts when they contain `return` statements.

ii. It is possible to reach the end of the function without returning any value.

iii. The reserved keyword `return` cannot be used in the body of an `if` statement.

iv. Functions cannot have multiple `return` statements.

v. The third `if` statement is not reachable.

(c) Which of the following best describes what is a *data type*. Circle the best answer.

i. A set of values.

ii. A set of operations.

iii. A sequence of 0s and 1s.

iv. A set of values and operations on those values.

v. The type of the arguments, method name, and the type of the return value for a function.

(d) Consider the following desirable features for user input.

I. You can enter data *while* the program is executing.

II. You can execute your program with different input data without having to recompile your program.

III. You can redirect the data to come from a file.

What is (are) the primary reason(s) to use standard input instead of command-line arguments? Circle the best answer.

I only	I, II and III
I and II only	None
I and III only	

(e) Which one or more of the following are features of Java functions (static methods)? Circle all that apply.

i. Can be overloaded.

ii. Can produce side effects.

iii. Can call another function, including itself.

iv. Can return multiple values, and they can be of different types.

v. Can have multiple arguments, and they can be of different types.

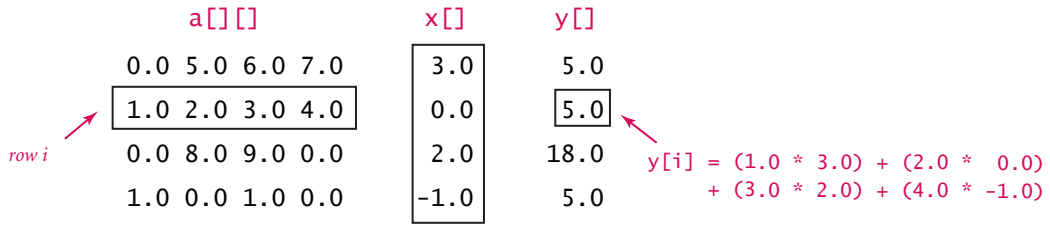
vi. Initializes the argument variable with a *copy* of the corresponding argument value provided by the calling code.

vii. The scope of a variable name declared within a function is limited to that function's body.

3. **Arrays, loops, functions, and debugging. (12 points)**

Given an `N`-by-`N` 2D array `a[][]` and a length `N` array `x[]`, the *matrix-vector product* `y[]` is defined such that `y[i]` is the dot-product of the ith row of `a[][]` with `x[]`:

```
y[i] = (a[i][0]*x[0]) + (a[i][1]*x[1]) + ...  + (a[i][N-1]*x[N-1])
```



Consider the following (buggy) function `times()` for the matrix-vector product.

```
public double[] y times(double[][] a, double[] x) {
   sum = 0.0;
   for (int i = 1; i <= N; i+)
       for (int j = 1; j <= N; j+)
           sum = a[i][j] * x[i];
       y[i] = sum;
}
```

Fix all of the errors and write the corrected function in the box below.

4. **TOY. (8 points)**

Consider each of the following TOY programs (which are identical except for memory addresses `11` and `13`). Suppose the program counter is set to `10`.

(a)
```
10: 7111   R[1] <- 0011
11: 7255   R[2] <- 0055
12: 2221   R[2] <- R[2] - R[1]
13: D212   if (R[2] > 0) goto 12
14: 0000   halt
```

Does the program halt? If so, what is the value of `R[2]` when it halts?

(b)
```
10: 7111   R[1] <- 0011
11: 8210   R[2] <- mem[10]
12: 2221   R[2] <- R[2] - R[1]
13: D211   if (R[2] > 0) goto 11
14: 0000   halt
```

Does the program halt? If so, what is the value of `R[2]` when it halts?

(c)
```
10: 7111   R[1] <- 0011
11: A201   R[2] <- mem[R[1]]
12: 2221   R[2] <- R[2] - R[1]
13: D212   if (R[2] > 0) goto 12
14: 0000   halt
```

Does the program halt? If so, what is the value of `R[2]` when it halts?

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] ^  R[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.

16-bit registers (using two's complement arithmetic)
16-bit memory locations
 8-bit program counter

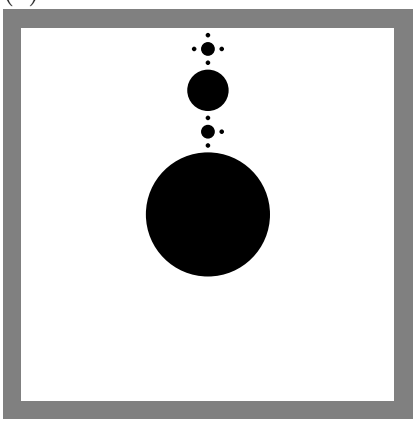
5. **Recursive graphics. (6 points)**

Consider the following recursive Java function.

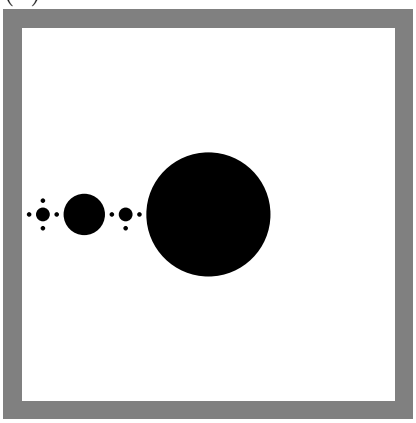
```
public static void mystery(int n, double x, double y, double size) {
   if (n == 0) return;
   StdDraw.filledCircle(x, y, size/6);
   mystery(n-1, x - size/3, y , size/3);
   mystery(n-1, x + size/3, y , size/3);
   mystery(n-1, x,  y - size/3, size/3);
   mystery(n-1, x,  y + size/3, size/3);
}
```

Suppose that you call `mystery(4, .5, .5, 1)`. Select the figure below that results after the 11th call to `StdDraw.filledCircle()`.

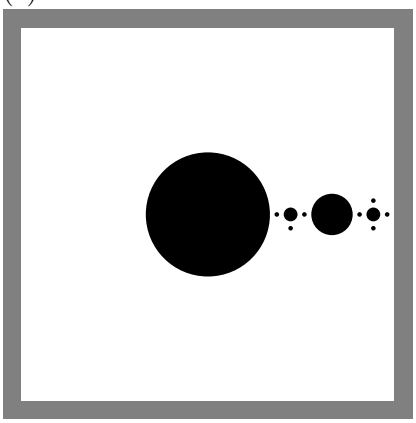
(a)



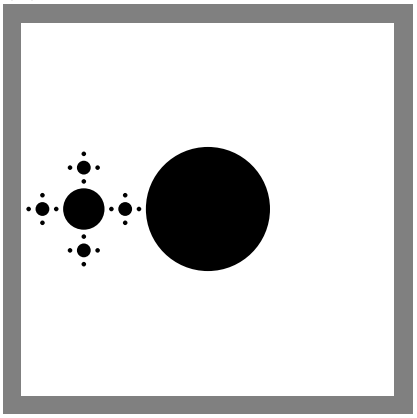
(b)



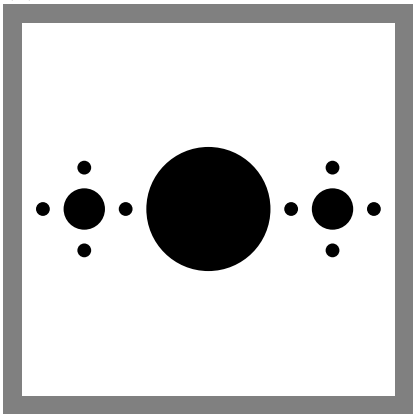
(c)



(d)



(e)



(f)