CMPS 12A—Winter 2006
Prof. Scott A. Brandt
Final Exam, March 21, 2006

Name:_____

Email:_____
This is a closed note, closed book exam. There are II sections worth a total of 200 points. Plan your time accordingly. If you are asked to write code, declare all variables that you use. Unless explicitly permitted, your code may not use any other methods. Write legibly—any answer I cannot read is incorrect.

Section I: [5 points each] For each of the following questions, **write the letter of the best answer to the left of the question** including, where appropriate, "All" for "All of the above" or "None" for "None of the above".

1. The design of a program may not include:
   a. The algorithms used
   b. Code
   c. Pseudocode

2. Which are all Java keywords:
   a. public, static, void, main, float
   b. String, int, double, while, and for
   c. switch, char, continue, this, if

3. Which are all valid identifiers:
   a. main, inti, forloop, String, first_one
   b. f77, 77f, break, String, println
   c. for, five, SIX, se_ven, ei$ght

4. Which are valid literals:
   a. 3.0, 1.2F, 5L, '\n', "123"
   b. 123, '123', 123L, "123", 1.23
   c. -2, seven, 4L, "17", "7"

5. Which of the following need not be specified as part of a variable declaration:
   a. The type of the variable
   b. The identifier of the variable
   c. The value of the variable

6. Which is not a difference between an int and a long:
   a. There is no difference
   b. The number of bits used to store them
   c. The size of the number they can store

7. Which of the following results in a == 1?
   a. int a = 7/4;
   b. int a = 3.0/4 + 0.25; // But it does produce an error without (int)(3.0/4 + 0.25)
   c. int a = (3+1)/4;

8. What does this code return?

```
int a = 5, b = 7;
return(a==5 || (!(a==b-2)&&(b==a+b-7)))
```

   a. true
   b. false
   c. a

9. Which list of operators is listed in order of precedence?
   a. +, *, /, =
   b. ++. *, +, =
   c. &&, *, +, %

10. What is the value of the expression (1 + 2 * (3 – 2) * 3 + 1)?
    a. 0
    b. 8
    c. 2

11. What does this code print out?

```
int a = 1, b = 2;
if(b < a)
     System.out.println("1");
     else if(a < b)
          System.out.println("2");
else
     System.out.println("3");
```

   a. 1
   b. 2
   c. 3

12. What does this code print out?

```
int a = 'h';

switch(a) {
     case 'a': System.out.println("a");
     case 'A': System.out.println("A"); break;
     case 'h': System.out.println("h");
     case 'H': System.out.println("H"); break;
     case 'x': System.out.println("x");
     case 'X': System.out.println("X"); break;
}
```

It prints out: **h**
H

13. What does this code print out?

```
class Foo {
     Public static void main(String[] args) {
          int[] foo = {5, 7, 3};
          if (foo[0] > foo[1]) swap(foo, 1, 2);
          if (foo[1] > foo[2]) swap(foo, 0, 1);
          if (foo[0] > foo[1]) swap(foo, 0, 2);
          System.out.println(foo[0] + "," + foo[1] +
          "," + foo[2]);
     }
     void swap(int[] a, int b, int c) {
          int temp = a[b]; a[b] = a[c]; a[c] = temp;
     }
}
```

a. 3,5,7
b. 7,5, 3
c. 3,7,5

14. In for loops:

a. The initialization expression is executed exactly once
b. The boolean expression is executed one or more times
c. The update expression is executed zero or more times

15. What is the value of j after this code executes?

```
for(int i = 1, j = 1; i < 1000; i = i*(i+i))
     System.out.println(j++ + " ");
```

a. 1
b. 4
c. 8

16. Which is true?
    a. The scope of an object is any code in the same block
    b. Arrays are call-by-reference
    c. Java doesn't allow two return statements in the same method

17. What does this method compute (given a positive integer n)?

```
public int foo(int n) {
     if(n == 1) return 1;
     else return foo(n-1) + foo(n-1);
}
```

    a. $(n-1)^2$
    b. 2^{n-1}
    c. n

18. How many ints can this 2D array store?

```
int[][] = new int[3][5];
```

    a. 8
    b. 15
    c. 125

19. public static void main(String[] args)
    a. public: anyone can call this method
    b. static: there is one copy no matter how many objects of this type are created
    c. void: the method doesn't return anything

20. The keyword `class`
    a. Defines a new object
    b. Defines a new variable
    c. Defines a new type

Section II: [100 points] 9 questions

1. [10 points] Write code to do the following:

a. Multiply two integer variables and store the result in a third.

```
int a, b, c;
c = a*b;
```

b. Print out true or false depending upon whether or not an integer variable is between 0 and 10 (including possible equal to 0 or 10).

```
int i;

if(i >= 0 && i <= 10)
        System.out.println("true");
else
        System.out.println("false");
```

c. Print out "red", "green", "blue", or "error" depending upon whether a char variable c is 'r', 'g', 'b'. or something else, using a switch statement.

```
char c;
switch(c) {
        case 'r': System.out.println("red"); break;
        case 'g': System.out.println("green"); break;
        case 'b': System.out.println("blue"); break;
        default: System.out.println("something else"); break;
}
```

2. [10 points] Write a main() method (*i.e.* a program) that reads in three characters and prints out "yes" if they are all the same and "no" if they are not.

```
public static void main(String[] args) {
        char a, b, c;
        String s;
        Scanner in = new Scanner(System.in);

        s = in.next();
        a = s.charAt(0);

        s = in.next();
        b = s.charAt(0);

        s = in.next();
        c = s.charAt(0);

        if(a == b && b == c)
                System.out.println("yes");
        else
                System.out.println("no");
}
```

3. [10 points] Write a method that calculates the roots of a quadratic equation $ax^2 + bx + c$ given a, b, and c. Recall that the roots of a quadratic equation are $\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$. You may use Math.sqrt() to calculate the square root.

```
void roots(double a, double b, double c) {
        double root1, root2;

        double temp = Math.sqrt(b*b - 4*a*c);

        root1 = (-b + temp)/2*a;
        root2 = (-b - temp)/2*a;

        System.out.println("The roots are " + root1 + " and " + root2);
}
```

+2 points if you noticed that Math.sqrt() may fail if $b^2 < 4ac$.

4. [10 points] Write a method called squares() that takes an integer parameter n and uses a for loop to find all of the integers less than or equal to n that are the square of some smaller integer. Do not use any other methods in your method (such as Math.sqrt()).

```
void squares(int n) {
	for(int i = 1; i <= n; i++) {
		if(i * i < n)
			System.out.println(i*i);
		else
			break;
	}
}
```

OR

```
void squares(int n) {
	for(int i = 1; i <= n; i++) {
		for(int j = 1; j < i; j++)
			if(j*j == i) {
				System.out.println(i);
				break;
			}
		}
	}
}
```

5. [10 points] Write some code that generates random integers between 1 and 10 until their sum is equal to 100, then prints out how many numbers it generated. Recall that Math.random() (which you may use) returns a random double between 0 and 1.

```
int r;
int sum = 0;

while(sum < 100) {
	r = (int)(1+9*Math.random());
	sum += r;
}
```

6. [10 points] Write a method called sort() that takes as a parameter an array of integers, and performs bubble sort on them. Recall that bubble sort works by repeatedly scanning through the array and swapping out-of-order adjacent pairs in the array until no more out-of-order pairs are found (and thus, the array is sorted).

```
void sort(int[] nums) {
	boolean done = true;

	while(!done) {
		done = true;
		for(int i = 0; i < nums.length-1; i++) {
			if(nums[i] > nums[i+1]) {
				int temp = nums[i];
				nums[i] = nums[i+1];
				nums[i+1] = temp;
				done = false;
			}
		}
	}
}
```

7. [10 points] Write a recursive function to calculate $f(n) = 2^n$.

```
// Assumes n > 0
int f(int n) {
	if(n == 1)
		return 2;
	else
		return 2*f(n-1);
}
```

8. [10 points] A queue is a data structure that contains data objects. Data objects may be enqueueud (added to the list) or dequeued (removed from the list). Queues dequeue objects in the same order as they were enqueued. Given a class Queue made to store ints that supports two methods: `void enqueue(int n),` and `int dequeue(),` show how you would declare a queue, enqueue 3, 5, and 7, then dequeue and print each number.

```
Queue q;
int j;

q.enqueue(3);
q.enqueue(5);
q.enqueue(7);

for(int i = 0; i < 3; i++) {
        j = q.dequeue();
        System.out.println(j);
}
```

9. [20 points] Implement the Queue class from problem 8.

```
// This queue can hold 100 ints
class Queue {
	int[] list;
	int head;
	int length;

	Queue() {
		list = new int[100];
		length = 0;
		head = 0;
	}

	void enqueue(int n) {
		list[(head + length)%list.length] = n;
		length++;
	}

	int dequeue() {
		int temp;

		if(length > 0) {
			temp = list[head];
			length--;
			head++;
			return length;
		} else {
			// In really good java code we
			// would throw an exception here
			System.out.println("You can't remove something from
an empty queue");
			exit(-1);
		}
	}
}
```