CMPS 12A—Winter 2012
Prof. Scott A. Brandt
Midterm 2, March 2, 2012

Name:_____

Email:_____

Read this first: This is a closed note, closed book exam. There are 8 questions worth a total of 110 points + 1 extra credit question. Plan your time accordingly. If you are asked to write code, declare all variables that you use but do not define the class or (unless asked) the method; just provide the relevant lines of code. Write legibly—any answer I cannot read is incorrect.

1. Given the following javadoc specification:

```
static double mult(double a, double b)
Returns the product of two double values.
```

a. [5 points] Show how to use the `mult()` function to multiply two integer variables x and y.

int z = mult(x, y);

b. [5 points] Show how to use the `mult()` function to multiply three integer variables x, y, and z.

int w = mult(x, mult(y, z));

2. [10 points] Write a method called `cube()` that takes an int parameter (call it `x`) and returns the value x^3.

```
double cube(int x) {
    return x * x * x;
}
```

3. [10 points] Write a method called `issqrt()` that takes two integer parameters (call them `a` and `b`) and returns `true` or `false` depending upon whether or not `a` is the square root of `b`. Do not call any other methods in your method.

```
boolean issqrt(int a, int b) {
        return (b == a * a);
}
```

4. [10 points] Given the following numbered lines of code, write the numbers of the lines that constitute the scope of each variable.

```
10 class Bar {
20
30   public static void main(String[] args) {
40        int i = 1;
50
60        for(int j=0; j<100; j++) {
70             i = i+j;
80        }
90
100       {
110            int k;
120            k = foo(i);
130       }
140       System.out.println(k);
150  }
160
170  public static double foo(int m) {
180       double n = 1/m;
190       return n;
200  }
210 }
```

i: 40-150

j: 60-80

k: 110-130

m: 170-200

n: 180-200

4. [10 points] What does this program print out?

```
class TestProgram{
     public static void main(String[] args){
          int a = 7, b = 5;
          System.out.println(a);
          System.out.println(b);
          a = foo(a, b);
          System.out.println(a);
          System.out.println(b);
     }
     static int foo(int x, int y) {
          x = 3; y = 1;
          System.out.println(x);
          System.out.println(y);
          return y;
     }}
```

7
5
3
1
1
5

5. Implement a method called `n17()` that takes an integer parameter `n` and returns n^{17} = n x n x n x n … x n (17 times).

a) [10 points] Write a *non*-recursive implementation of `n17()` using a for loop.

```
int n17(int n) {
	int result = 1;

	for(int i = 0; i <= 17; i++)
		result = result * n;

	return result;
}
```

b) [10 points] Write a *recursive* implementation of the same function. Recall that $n^x = n*n^{x-1}$ and $n^0 = 1$.

```
int x = 17;

int nx(int n, int x) {
	if(n <= 0)
		return 1;
	else
		return 2 * n17(n, x-1);
}
```

6. Arrays

a) [5 points] Declare and create storage for an array of 12 ints called `monthlytemps.`

```
int monthlytemps = new int[12];
```

b) [10 points] Write a method called `stretch()` that takes an array of ints and finds the average of the numbers in the array, then adds 1 to each array element with a value above the average and subtracts 1 from each array element with a value below the average.

```
void stretch(int[] a) {
	int sum = 0; double average;

	for(int i = 0; i < a.length; i++)
		sum += a[i];

	average = (double)sum/(double)a.length;

	for(int i = 0; i < a.length; i++)
		if(a[i] > average) a[i]++;
		else if(a[i] < average) a[i]--;
}
```

c) [5 points] Show how you would call `stretch()` with the `monthlytemps` array you created as a parameter

```
smooth(monthlytemps);
```

7. [10 points] Write a method called `min()` that takes two integer arrays (of the same length) as parameters and returns a third array, where each element of the third array is the smaller of the corresponding elements of the two input arrays. Call the arrays one, two, and three.

```
int[] min(int[] one, int[] two) {
	int[] three = new int[one.length];

	for(int i = 0; i < a.length; i++)
		if(one[i] < two[i])  three[i] = one[i];
		else three[i] = two[i];

	return three;
}
```

8. [10 points] Write a method called `sumtable()` that creates and returns a 2D array of ints containing the table of sums from 0 to 9 where the (i, j)th element contains i plus j..

```
int[][] sumtable() {
	int[][] st = new int[10][10];

	for(int i = 0; i < st.length; i++)
		for(int j = 0; j < st[i].length; j++)
			st[i][j] = i + j;

	return st;
}
```

Extra credit. [10 points] Write a method called `mean()` that takes as input a 2D array of ints and replaces each element of the array with the average of the 9 pixel square around that point in the array.

```
void mean(int[][] a) {
	double avg;

	int[][] b = a.clone();

	for(int i = 0; i < a.length; i++)
		for(int j  = 0; j < a[i].length; j++)
			a[i][j] = calcavg(b, i, j);


int calcavg(int[][] a, int i, int j) {
	int sum = 0;

	sum += a[i][j];
	if(i > 0 && j > 0) sum += a[i-1][j-1];
	if(i > 0) sum += a[i-1][j];
	if(i > 0 && j < a[i].length-1 sum += a[i-1][j+1];
	if(j > 0) sum += a[i][j-1];
	if(j < a[i].length-1) sum += a[i][j+1];
	if(i < a.length-1 && j > 0) sum += a[i+1][j-1];
	if(i < a.length) sum += a[i+1][j];
	if(i < a.length-1 && j < a[i].length-1) sum += a[i+1][j+1];

	return (int)(sum/9.0);
}
```