CIS 200: Fundamentals of Software Design
Exam 2 (50 points)

Practice Exam

You may use one 8.5”x11” page of handwritten notes, but no books, computers, calculators, or cell phones on the exam. Please do your own work.

Name:_____

Lab section (day and time):_____

EXAM SCORE: _____

1. (6 pts) Consider the following questions about arrays.
   a) (2 pts) Declare a two-dimensional array of character values with 3 rows and 2 columns. You may call the array anything you like.

```
char [] [] grid = new char[3][2];
```

   b) (4 pts) Initialize the appropriate spots in the array to make the array look like this:

‘a’	‘A’
‘b’	‘B’
‘c’	‘C’

```
grid[0][0]=’a’;
grid[0][1]=’A’;

grid[1][0]=’b’;
grid[1][1]=’B’;
etc.
```

2. (10 pts) Write a method called **`sumDoubleDigits`**. This method should take an integer array as a parameter. It should then return the sum of all double-digit numbers in the array (for example, both 99 and -10 count as double-digit numbers).

```
public static int sumDoubleDigits (int [ ] arr) {
int sum = 0;
for (int i = 0; i  < arr.length; i++) {
  if (arr[i] >= 10 && arr[i] <=99) sum+= arr[i];
 else if (arr[i] > -100 && arr[i] < -9) sum+= arr[i];
}

return sum;
}
```

3. (4 pts) Suppose you are in the same class as the **`sumDoubleDigits`** method. Further suppose that an integer array called **`values`** has been declared and filled with values. Finally, suppose an integer variable called **`answer`** has been declared. Write ONE line of code that stores the sum of the double digit elements in **`values`** in the **`answer`** variable.

```
answer = sumDoubleDigits (values);
```

4. (10 pts) Suppose **numbers.txt** is an input file that contains several numbers with decimals, one per line. It might look like this:

   3.42
   17.0
   47.151

   Write a code fragment that reads every line in the file, and prints the decimal portion of each number to an output file called **afterDecimal.txt**. For the example above, the resulting **afterDecimal.txt** file should look like this:

   42
   0
   151

   Do not assume that there are only three lines in the file – your code should work no matter how many lines there are.

```
Scanner inFile = new Scanner (new File("numbers.txt");

PrintWriter pw = new PrintWriter (new FileWriter
("afterDecimal.txt");

while (inFile.hasNext()) {
    String line = inFile.nextLine();
    String [ ] pieces = line.split(".");
    pw.println(pieces[1]);
}
inFile.close();
pw.close();
```

5. (8 pts) Consider the following full Java program. When this program runs, what is printed?

```
public class Exam {
     public static void main(String[] args) {
          System.out.println(“main”);
          int[] arr = {1,2,3,4};
          int val = 10;
          int result = second(arr, val);     // result = 11
          System.out.println(val);
          System.out.println(result);
          System.out.println(arr[0]);
     }

     public static void first(int[] list) {
          System.out.println(“first”);
          list[0] = 4;   // arr[0] replaced with 4
     }

     public static int second(int[] nums, int x) {
          System.out.println(“second”);
          first(nums);
          System.out.println(“middle of second”);
          x = 7;
          return nums[0]+x;        // 4 + 7 = 11
     }
}
```

arr = 1,2,3,4…arr[0] replaced w/4 in method “first”
val = 10

Output

main
second
first
middle of second
10
11
4

6. (4 pts) Consider the following statements. What is printed?

```
String example = "problem3onexam";
System.out.println(example.substring(9));
System.out.println(4 + 5 + example.substring(7,8) + 9); // 9+"3"+9
System.out.println(example.charAt(3)+example.substring(5,8));
```

example[0] = p
example[3]= b
example[9]= n

OUTPUT
nexam
939
bem3

7. (8 pts) We would like to be able to call a **`multiply`** method like this:

```
double times = multiply(4.6, 3.2);
```

and have `times` hold 14.72 (which is 4.6 times 3.2). In general, we'd like to be able to pass two numbers and get back their product. **Write the method `multiply` that satisfies those criteria**.

```
public static double multiply (double first, double second)
{ double product = first * second;
    return product;
}

or { return first * second;
}
```