CSE 142
Sample Final 1

Name of Student____

Section (e.g., AA)___________________ TA_____________________________________

This exam is divided into eleven questions with the following points:

#	Problem Area	Points	Score
1	Expressions	5	_____
2	Array Simulation	10	_____
3	Inheritance	6	_____
4	Token-Based File processing	10	_____
5	Line-Based File Processing	9	_____
6	Arrays	10	_____
7	ArrayList	10	_____
8	Critters	15	_____
9	Arrays	15	_____
10	Programming	10	_____
11	Art (bonus)	1	_____
	Total	100	_____

This is a closed-book/closed-note exam. Space is provided for your answers. There is a "cheat sheet" at the end that you can use as scratch paper or to write answers. You are not allowed to access any other paper during the exam (not even scratch paper). Use the backs of the pages of this test if necessary. Anyone caught with extra paper will lose at least 10 points.

In general the exam is not graded on style and you do not need to include comments, although the Critter class has special requirements you must follow. You do not have to include any import statements. Do not abbreviate any code that you write (e.g., S.o.p versus System.out.print) and do not abbreviate any answer asking for sample output (show the complete output).

You are NOT to use any electronic devices while taking the test, including calculators. Anyone caught using an electronic device will receive a 10 point penalty.

Do not begin work on this exam until instructed to do so. Any student who starts early or who continues to work after time is called will receive a 10 point penalty.

If you finish the exam early, please hand your exam to the instructor and exit quietly through the front door.

1. Expressions, 5 points. For each expression in the left-hand column, indicate its value in the right-hand column. Be sure to list a constant of appropriate type (e.g., 7.0 rather than 7 for a double, Strings in quotes).

Expression	Value
17 / 3 + 55 % 20 / 2	__________
19.0 / 2 + 2 * 2.25	__________
2 + 2 + "." + 2 + 2 + 4 * 5	__________
73 / 10 * 4 / 3 / 2.0 * 3	__________
3 / 2 * (1.5 + 2.5) + 98 % 5 / 2	__________

2. Array Simulation, 10 points. You are to simulate the execution of a method that manipulates an array of integers. Consider the following method:

```
public static void mystery(int[] list) {
    for (int i = 2; i < list.length; i++) {
        if (i % 2 == 0) {
            list[i] = list[i] + list[i - 2];
        }
    }
}
```

In the left-hand column below are specific lists of integers. You are to indicate in the right-hand column what values would be stored in the list after method mystery executes if the integer list in the left-hand column is passed as a parameter to mystery.

Original List	Final List
{2, 5, 8}	____________________________
{3, 4, 7, 5}	____________________________
{2, 4, 6, 2, 6}	____________________________
{4, 5, 8, 9, 3}	____________________________
{1, 2, 8, 5, 10, 5, 4}	____________________________

3. Inheritance, 6 points. Assume the following classes have been defined:

```
public class D extends C {
    public void method1() {
        System.out.println("d 1");
    }
}

public class C {
    public void method1() {
        System.out.println("c 1");
    }

    public void method2() {
        System.out.println("c 2");
    }

    public String toString() {
        return "c";
    }
}

public class A extends C {
    public void method1() {
        System.out.println("a 1");
    }

    public String toString() {
        return "a";
    }
}

public class B extends A {
    public void method2() {
        System.out.println("b 2");
    }
}
```

Consider the following code fragment:

```
    C[] elements = {new B(), new C(), new A(), new D()};
    for (int i = 0; i < elements.length; i++) {
        System.out.println(elements[i]);
        elements[i].method1();
        elements[i].method2();
        System.out.println();
    }
```

What output is produced by this code? Write the output as a series of 3-line columns in order from left to right (do not label columns or rows).

4. Token-Based File Processing, 10 points. Write a static method called printStrings that takes as a parameter a Scanner holding a sequence of integer/String pairs and that prints to System.out one line of output for each pair with the given String repeated the given number of times. For example if the Scanner contains the following data:

```
    6 fun. 3 hello 10 <> 4 wow!
```

your method should produce the following output:

```
    fun.fun.fun.fun.fun.fun.
    hellohellohello
    <><><><><><><><><><>
    wow!wow!wow!wow!
```

Notice that there is one line of output for each integer/String pair. The first line has 6 occurrences of "fun.", the second line has 3 occurrences of "hello", the third line has 10 occurrences of "<>" and the fourth line has 4 occurrences of "wow!". Notice that there are no extra spaces included in the output. You are to exactly reproduce the format of this sample output. You may assume that the input values always come in pairs with an integer followed by a String. If the Scanner is empty (no integer/String pairs), your method should produce no output.

5. Line-Based File Processing, 9 points. Write a static method called flipLines that takes a Scanner containing an input file as a parameter and that writes to System.out the same file with successive pairs of lines reversed in order. For example, if the input file contains the following:

```
    Twas brillig and the slithy toves
    did gyre and gimble in the wabe.
    All mimsey were the borogroves,
    and the mome raths outgrabe.

    "Beware the Jabberwock, my son,
    the jaws that bite, the claws that catch,
    Beware the JubJub bird and shun
    the frumious bandersnatch."
```

The program should print the first pair of lines in reverse order, then the second pair in reverse order, then the third pair in reverse order, and so on. Thus, your method should produce the following output.

```
     did gyre and gimble in the wabe.
     Twas brillig and the slithy toves
     and the mome raths outgrabe.
     All mimsey were the borogroves,
     "Beware the Jabberwock, my son,

     Beware the JubJub bird and shun
     the jaws that bite, the claws that catch,
     the frumious bandersnatch."
```

Notice that a line can be blank, as in the third pair. Also notice that an input file can have an odd number of lines, as in the one above, in which case the last line is printed in its original position. You may not make any assumptions about how many lines are in the Scanner.

6. Arrays, 10 points. Write a static method called minToFront that takes an array of integers as a parameter and that moves the minimum value in the list to the front by swapping its position with whatever is currently at the front of the list. For example, if a variable called list stores the following values:

```
     [3, 8, 92, 4, 2, 17, 9]
```

and you make the following call:

```
     minToFront(list);
```

The value 2 is the minimum, so the list should store the following values after the call:

```
     [2, 8, 92, 4, 3, 17, 9]
```

Notice that the value 3 which used to be at the front of the list is now at index 4 where the value 2 was before. If there is more than one occurrence of the minimum value, your method should move the first occurrence to the front of the list. If the minimum value is already at the front of the array or if the array is empty, then the array should be unchanged after the method executes.

You may not construct any extra data structures to solve this problem (not even a string).

7. ArrayList, 10 points. Write a static method called reverse3 that takes an ArrayList of integer values as a parameter and that reverses each successive sequence of three values in the list. For example, suppose that a variable called list stores the following sequence of values:

```
     [3, 8, 19, 42, 7, 26, 19, -8, 193, 204, 6, -4]
```

and we make the following call:

```
     reverse3(list);
```

Afterwards the list should store the following sequence of values:

```
     [19, 8, 3, 26, 7, 42, 193, -8, 19, -4, 6, 204]
```

The first sequence of three values (3, 8, 19) has been reversed to be (19, 8, 3). The second sequence of three values (42, 7, 26) has been reversed to be (26, 7, 42). And so on. If the list has extra values that are not part of a sequence of three, those values are unchanged. For example, if the list had instead stored:

```
[3, 8, 19, 42, 7, 26, 19, -8, 193, 204, 6, -4, 99, 2]
```

The result would have been:

```
[19, 8, 3, 26, 7, 42, 193, -8, 19, -4, 6, 204, 99, 2]
```

Notice that the values (99, 2) are unchanged in position because they were not part of a sequence of three values. You may not construct any extra data structures to solve this problem. You must solve it by manipulating the ArrayList you are passed as a parameter.

8. Critters, 15 points. Write a class called Raccoon that extends the Critter class. The instances of the Raccoon class always infect when an enemy is in front of them and otherwise randomly choose between turning left and turning right, with each choice being equally likely. Their appearance changes over time. Each Raccoon initially displays as a less-than followed by a dash followed by a greater-than ("<->"). Then as each Raccoon chooses a move, it changes its appearance to match that move. If its most recent move was an infect, it displays as "<I>". If its most recent move was to turn left, it displays as "<L>". And if its most recent move was to turn right, it displays as "<R>". Its color should alternate between blue and red. It should be blue initially when it is displayed as "<->" and then it should alternate between the two colors (red after the first move, blue after the second move, etc.).

   As in assignment 8, all fields must be declared private and fields that need to be initialized to a non-default value must be set in a constructor. If you want to use a Random object, you should make it a field so that you only construct it once for each Raccoon.

9. Arrays, 15 points. Write a static method called collapse that takes an array of integers as a parameter and that returns a new array that contains the result of collapsing the original list by replacing each successive pair of integers with the sum of the pair. For example, if a variable called list stores this sequence of values:

```
[7, 2, 8, 9, 4, 13, 7, 1, 9, 10]
```

   Then the following call:

```
collapse(list)
```

   Should return a new array containing the following values:

```
[9, 17, 17, 8, 19]
```

   The first pair from the original list is collapsed into 9 (7 + 2), the second pair is collapsed into 17 (8 + 9), the third pair is collapsed into 17 (4 + 13) and so on.

   If the list stores an odd number of elements, the final element is not collapsed. For example, if the list had been:

```
[1, 2, 3, 4, 5]
```

   Then the call on collapse would produce the following list:

```
[3, 7, 5]
```

   with the 5 at the end of the list unchanged. The method should not construct any extra data structures other than the array to be returned and it should not alter its parameter.

10. Programming, 10 points. Write a static method called acronym that takes as a parameter a string containing a phrase and that returns an acronym for the phrase. For example, the following call:

```
acronym("self-contained underwater breathing apparatus")
```

should return "SCUBA". The acronym is formed by combining the capitalized first letters of each word in the phrase. Words in the phrase will be separated by some combination of dashes and spaces. There might be extra spaces or dashes at the beginning or end of the phrase. The String will not contain any characters other than dashes, spaces, and letters, and is guaranteed to contain at least one word. Below are several sample calls.

Method Call	Value Returned
`acronym("   automatic   teller   machine  ")`	"ATM"
`acronym("personal identification number")`	"PIN"
`acronym("computer science")`	"CS"
`acronym("merry-go-round")`	"MGR"
`acronym("All my Children")`	"AMC"
`acronym("Troubled Assets Relief Program")`	"TARP"
`acronym("--quite-- confusing - punctuation-")`	"QCP"
`acronym("  loner  ")`	"L"

Key to CSE142 Sample Final 1

1.

Expression	Value
17 / 3 + 55 % 20 / 2	12
19.0 / 2 + 2 * 2.25	14.0
2 + 2 + "." + 2 + 2 + 4 * 5	"4.2220"
73 / 10 * 4 / 3 / 2.0 * 3	13.5
3 / 2 * (1.5 + 2.5) + 98 % 5 / 2	5.0

2.

Original List	Final List
{2, 5, 8}	{2, 5, 10}
{3, 4, 7, 5}	{3, 4, 10, 5}
{2, 4, 6, 2, 6}	{2, 4, 8, 2, 14}
{4, 5, 8, 9, 3}	{4, 5, 12, 9, 15}
{1, 2, 8, 5, 10, 5, 4}	{1, 2, 9, 5, 19, 5, 23}

3. Inheritance. The output produced is as follows.

```
a          c          a          c
a 1        c 1        a 1        d 1
b 2        c 2        c 2        c 2
```

4. Token-Based File Processing. One possible solution appears below.

```
public static void printStrings(Scanner input) {
    while (input.hasNextInt()) {
        int times = input.nextInt();
        String word = input.next();
        for (int i = 0; i < times; i++) {
            System.out.print(word);
        }
        System.out.println();
    }
}
```

5. Line-Based File Processing. One possible solution appears below.

```
        public static void flipLines(Scanner input) {
            while (input.hasNextLine()) {
                String first = input.nextLine();
                if (input.hasNextLine()) {
                    String second = input.nextLine();
                    System.out.println(second);
                }
                System.out.println(first);
            }
        }
```

6. Arrays. One possible solution appears below.

```
        public static void minToFront(int[] list) {
            if (list.length > 0) {
                int min = 0;
                for (int i = 0; i < list.length; i++) {
                    if (list[i] < list[min]) {
                        min = i;
                    }
                }
                int temp = list[0];
                list[0] = list[min];
                list[min] = temp;
            }
        }
```

7. ArrayLists. Two possible solutions appear below.

```
        public static void reverse3(ArrayList<Integer> list) {
            for (int i = 0; i < list.size() - 2; i += 3) {
                int n1 = list.get(i);
                int n3 = list.get(i + 2);
                list.set(i, n3);
                list.set(i + 2, n1);
            }
        }

        public static void reverse3(ArrayList<Integer> list) {
            for (int i = 0; i < list.size() - 2; i += 3) {
                list.add(i, list.remove(i + 2));
                list.add(i + 2, list.remove(i + 1));
            }
        }
```

8. Critters. One possible solution appears below.

```
        public class Raccoon extends Critter {
            private Random r;
            private String display;
            private int count;

            public Racoon() {
                r = new Random();
                display = "<->";
            }

            public Action getMove(CritterInfo info) {
                count++;
                if (info.getFront() == Neighbor.OTHER) {
                    display = "<I>";
                    return Action.INFECT;
                } else {
                    int flip = r.nextInt(2);
                    if (flip == 0) {
                        display = "<L>";
                        return Action.LEFT;
                    } else {
                        display = "<R>";
                        return Action.RIGHT;
                    }
                }
            }

            public Color getColor() {
                if (count % 2 == 0) {
                    return Color.BLUE;
                } else {
                    return Color.RED;
                }
            }

            public String toString() {
                return display;
            }
        }
```

9. Arrays. One possible solution appears below.

```
public static int[] collapse(int[] list) {
    int[] result = new int[list.length / 2 + list.length % 2];
    for (int i = 0; i < list.length; i++)
        result[i / 2] += list[i];
    return result;
}
```

10. Programming. Two possible solutions appear below.

```
public static String acronym(String s) {
    boolean inWord = false;
    s = s.toUpperCase();
    String result = "";
    for (int i = 0; i < s.length(); i++) {
        char ch = s.charAt(i);
        if (ch == ' ' || ch == '-') {
            inWord = false;
        } else if (!inWord) {
            inWord = true;
            result += ch;
        }
    }
    return result;
}
```