Exam Number:

Points off	1	2	3	4	5			Total off	Net Score

CS 305j – Midterm 2 – Fall 2007

Your Name__________________________________

Your UTEID _________________________________

Circle you TA's name: Priyanka Jacob

Instructions:

1. Please turn off your cell phones
2. There are 5 questions on this test.
3. You have 50 minutes to complete the test.
4. You may not use a calculator.
5. Please make your answers legible.
6. When code is required, write Java code.

1. Simulation. (4 points each, 12 points total.) You are to simulate a method that manipulates an array of integers. Consider the following method:

```
public static void mystery1(int[] data){
     int temp;
     int otherPosition;
     for(int i = 0; i < data.length / 2; i++){
          otherPosition = data.length - 1 - i;
          temp = data[i];
          data[i] = data[otherPosition];
          data[otherPosition] = temp;
     }
}
```

In the left hand column below is an indication of the values initially in array of integers. The left most element is at position 0. In the right hand column indicate what the elements of the array would be after method `mystery1` finished executing when the array in the left hand column is passed as the argument to method `mystery1`.

Initial Array	Resulting Array after method `mystery1`
`{17}`	________________________________
`{12, 1, 15}`	________________________________
`{7, 9, 13, -5, 3, 0}`	________________________________

2. Boolean expressions. (20 points) Write a method to determine the number of points a football team is predicted to score in an upcoming game. For this question, the predicted number of points a team will score is calculated as follows:

- For each 100 yards of rushing the team is expected to score 4 points.
- For each 100 yards of passing the team is expected to score 2 points.
- For each turnover recovered the team is expected to score 2 points.
- If the game is a home game the team is expected to score 3 points.

Points scored from each source can be fractional. thus if a team had 120 yards of rushing they would be expected to score 120 / 100 * 4 = 4.8 points from rushing.

```
// assume rushYards, passYards, and turnoversRecovered will all be >= 0
public static double expectedPoints(int rushYards, int passYards,
                                    int turnoversRecovered, boolean homeGame)
```

Here are some examples of call to `expectedPoints` and what value should be returned:

```
expectedPoints(100, 100, 0, false) -> 6.0
expectedPoints(100, 100, 1, true) -> 11.0
expectedPoints(0, 200, 2, false) -> 8.0
expectedPoints(120, 0, 0, true) -> 7.8
```

Complete the method below:

```
public static double expectedPoints(int rushYards, int passYards,
                                    int turnoversRecovered, boolean homeGame)
```

No material this page.

3. Arrays. (20 points) Complete a method that determines the difference between the minimum value of an array of doubles and the maximum value of an array. You may assume there is at least one element in the array. The values in the array are in no particular order. You may not call any other methods when completing this method.

The method you complete will have the following header:

```
// Assume values.length > 0
public static double diffMinMax(double[] values){
```

Here are some examples of call to `diffMinMax` and what value should be returned:

```
diffMinMax( {1.5} ) -> 0.0
diffMinMax( {1.5, 3.0} ) -> 1.5
diffMinMax( {3.0, -0.5} ) -> 3.5
diffMinMax( {10.5, 1.75, -2.5, 13.8, 7.5, 15.5, 12.8} ) -> 18.0
```

In the previous example the minimum is -2.5 and the maximum is 15.5.

Complete the method below:

```
public static double diffMinMax(double[] values){
```

```
// more room on next page if necessary
// more room for diffMinMax if necessary
```

4. Strings and Arrays. (25 points) Complete a method named `removeChars.` The method accepts a String parameter and an array of `chars`. The method returns a new String that is the same as the old except any occurrence of characters in the array of `chars` have been removed.

The method you complete will have the following header:

```
public static String removeChars(String org, char[] remove)
```

Here are some examples of call to `removeChars` and what value should be returned:

```
removeChars("Hi", {'a'} ) -> "Hi"
removeChars("Hi", {'h', 'i'} ) -> "H"
removeChars("Hi", {} ) -> "Hi"
```
(empty array of chars)

```
removeChars("Hi12Hi", {'H', 'i', '1', '2', 'a', 'b', '3'} ) -> ""
```
Previous example returns an empty String since all chars removed.

```
removeChars("Hi_12*Hi", {'H', 'i', '1', '2', 'a', 'b', '3'} ) -> "_*"
```

You will need to use the `charAt` and `length` methods from the `String` class. Complete the method below:

```
public static String removeChars(String org, char[] remove){
```

```
// more room on next page if necessary
// more room for method removeChars
```

5. Arrays (23 points) Complete a method `removeRange`. This method has 3 parameters, an array of integers, a starting index and an stopping index. The method returns a new array of values. The values are the same as the original array sent as a parameter, except all of the elements in the original array from the starting index inclusive to the ending index exclusive have been removed. You may not call any other methods when completing this method.

The method you complete will have the following header:

```
// Assume 0 <= start, start < data.length, start < stop, stop <= data.length
public static int[] removeRange(int[] data, int start, int stop)
```

Here is an example of an initial array.

```
  0  1  2   3  4  5  6  7   8  array indices
{12, 1, 3, -5, 1, 1, 7, 1, 15} array elements
```

If `start` is equal to 2 and `stop` is equal to 4, then elements 2 and 3 from the original array are removed. The resulting array that is returned would be:

```
{12, 1, 1, 1, 7, 1, 15}
```

Remember, you are to create a new array. The elements of the original array are unchanged.

Here are some more examples of parameters and expected results:

```
removeRange( {1, 2, 3, 4}, 0, 3 ) -> {4}
removeRange( {12, 5, 7, 3, 1}, 0, 3 ) -> {3, 1}
removeRange( {12, 5, 7, 3, 1}, 2, 3 ) -> {12, 5, 3, 1}
removeRange( {12, 5, 7, 3, 1}, 3, 5 ) -> {12, 5, 7}
removeRange( {12, 5, 7, 3, 1}, 1, 3 ) -> {12, 3, 1}
```

Complete the method below:

```
public static int[] removeRange(int[] data, int start, int stop){
```

```
// more room on the next page if needed.
// more room for method removeRange
```

Scratch Paper

Scratch Paper

Scratch Paper

Scratch Paper

Scratch Paper Exam Number: