Computer Sciences 302

Midterm Exam 1

100 points (15% of final grade)

Instructors: Deb Deppeler and Laura Hobbes LeGault

(Family) Last Name: _____ **(Given) First Name:** _____

CS Login Name: _____ **Net ID:** _____ @wisc.edu

Circle your Lecture: Lec 001 8:00 TR Lec 002 1:00 TR Lec 003 1:20 MWF

Fill in these fields (left to right) on the red scantron form (use #2 pencil):

1. LAST NAME (family name) and FIRST NAME (given name) and fill in bubbles
2. IDENTIFICATION NUMBER is your Campus ID number, fill in bubbles
3. Under ABC of SPECIAL CODES, write your lecture number as a 3 digit value 001, 002, or 003
4. Write the letter P for primary under the letter J and fill in the number (1) bubble

The following exam has 25 questions and is worth a total of 100 points. You will have 120 minutes to complete the exam. **Be sure to read through every question completely.**

I certify that I will keep my answers covered so that they may not be viewed by another student during the exam or prior to completion of their exam. I also certify that I will not view or in any way use another's work or any unauthorized devices. I understand that I may not make any type of copy of any portion of this exam without express permission from my instructor. I understand that being caught allowing another to view my work or being caught viewing another's work are both violations of this agreement and either will result in automatic failure of the course. Any penalty will also be reported to the Deans Office for all involved.

Signature: _____

---

1. Be sure to review the reference pages as needed during the exam.
2. **Turn off and put away** your cell phone, calculator, Inspector Gadget (watches glasses pencils, etc.) now and wait for the proctor to signal the start of the exam.

Disclaimer: the following are provided for your reference only, and the inclusion of information here does not guarantee it will be used on the exam.

Operator Precedence Table:

level	operator	description
higher	`( <expression> )`	grouping with parentheses
	`[ ] ( ) .`	array index, method call, member access (dot operator)
	`++ --`	post-increment, post-decrement
	`++ -- + - !`	pre-increment, unary plus/minus, logical negation
	`(type) new`	casting and creating object
↕	`* / %`	multiplication, division, modulus
↕	`+ - +`	addition, subtraction, concatenation
↕	`< <= > >=`	relational
↕	`== !=`	equality
↕	`&&`	conditional AND (short-circuits)
↕	`||`	conditional OR (short-circuits)
↕	`? :`	ternary conditional
lower	`= += -= *= /= %=`	assignment, arithmetic (compound) assignment

Methods from the java.lang.Object class: (all reference variables have these methods)

`String toString()`	Returns a `String` representation of the object. This is the hashcode of the instance unless `toString()` has been overridden.
`boolean equals(Object o)`	Returns `true` if the object referenced as `o` is the same as `this`. It is often overridden (redefined) by the class.

Constant and Methods from the java.lang.Math class:

`Math.PI`	Field that represents the constant π
`double random()`	Returns a random value between 0 (inclusive) and 1 (exclusive)
`double pow(double x, double n)`	Returns x^n
`double sqrt(double n)`	Returns $\sqrt{n}$
`double abs(double n)`	Returns the absolute value of n
`double ceil(double n)`	Returns the value of n rounded up to the nearest whole number.

Methods from the java.util.Random class:

`Random()`	Creates a new random number generator.
`Random(int s)`	Creates a new random number generator seeded with `s`.
`int nextInt()`	Returns the next pseudo-random integer value.
`int nextInt(int n)`	Returns the next pseudo-random integer value between 0 (inclusive) and n (exclusive).
`double nextDouble()`	Returns the next pseudo-random double value between 0.0d (inclusive) and 1.0d (exclusive)

Methods from the java.lang.String class: (*REMEMBER 0-based indexing is used)

`int length()`	Returns number of characters in the `String`
`char charAt(int index)`	Returns the character at the specified `index`
`boolean contains(String s)`	Returns `true` iff string s is in this string, otherwise `false`
`String toLowerCase()`	Returns a*new* string that is the lowercase version of this string.
`String toUpperCase()`	Returns a *new* string that is the UPPERCASE version of this strin
`int indexOf(String s)`	Returns the index within this string of the first character of the first occurrence of the specified string s.
`boolean equals(String s)`	Returns `true` if the contents of this String is the same as the contents of String s.
`boolean equalsIgnoreCase(String s)`	Returns `true` iff the contents of the `this` string is the same as that of the string `s`, ignoring differences in case.
`String substring(int begin)`	Returns a *new* string that is a substring of this string starting at begin to the end of the this string.
`String substring(int begin, int end)`	Returns a *new* string that is a substring of this string starting at index `begin` up to but not including `end`.
`boolean startsWith(String prefix)`	Returns `true` if this string starts with the specified prefix `prefix`, `false` otherwise.
`boolean startsWith(String pre, int off)`	Returns `true` if `this` string starts at the specified offset `off` with the specified prefix `pre`, `false` otherwise.

Methods from the java.util.Scanner class:

`Scanner(String s)`	Creates a `Scanner` to read the `String s`
`Scanner(System.in)`	Creates a `Scanner` that reads from the keyboard.
`Scanner(File fn) throws FileNotFoundException`	Creates a `Scanner` to read from file
`void close() throws IOException`	Closes the stream and any associated file
`boolean hasNext()`	Returns `true` if there's another token of input.
`boolean hasNextInt()`	Returns `true` if the next input is an int value.
`boolean hasNextDouble()`	Returns `true` if the next input is a double value.
`boolean hasNextLine()`	Returns `true` if there's another line of input.
`String next()`	Returns the next word only, as a String.
`int nextInt()`	Returns the next word only, as an integer.
`double nextDouble()`	Returns the next word only, as a double.
`String nextLine()`	Returns the next line as a String.

Method from the java.util.Arrays class:

`static String toString(T[] array)`	Returns a `String` representation of any type (`T[]`) array.