CSE 131S Introduction to Computer Science	**Summer SON 2014**
Exam I	
Given: 1 July 2014	Due: End of live session

This exam is closed-book, closed-notes, no electronic devices allowed except for downloading the exam and sending the completed exam. The exception is the "cheat sheet" on which you may have notes to consult during the exam.

Your work must be legible. Work that is difficult to read will receive no credit. Do not dwell over punctuation or exact syntax in code; however, be sure to indent your code to show its structure.

You must sign the pledge below for your exam to count. Any cheating will cause the students involved to receive an F for this course. Other action may be taken.

You must fill in your identifying information correctly.

To submit your exam, mail the scanned pages to `roncytron@gmail.com`.

Print clearly the following information:		
Name (print clearly):		
Student 6-digit ID (print *really* clearly):		
Problem Number	Possible Points	Received Points
1	20	
2	30	
3	25	
4	25	
Total	100	

Pledge: On my honor, I have neither given nor received any unauthorized aid on this exam.

Signed: _____

(Be sure you filled in your information in the box above!)

1. (20 points)

   (a) (10 points) Circle the correct type for each expression in the table below, and state the result of evaluating the expression:

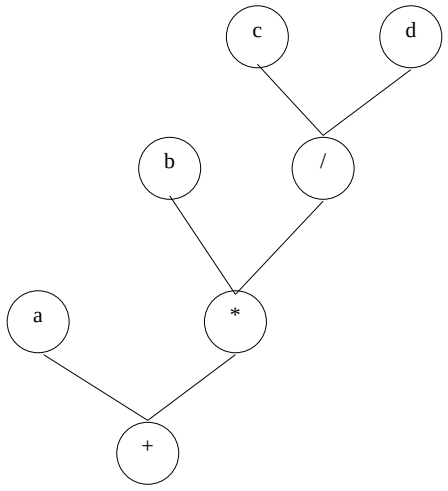
Expression	Type	Result
`1.0 / 2`	double int boolean String	
`true || false`	double int boolean String	
`"100" + (30+1)`	double int boolean String	
`"d" + (7/8) + "od"`	double int boolean String	
`(3/2 <= 1) + "ly"`	double int boolean String	

   (b) (4 points) Below draw the expression tree for the expression

$$1 + (2 - 3) * 4$$

   (c) (1 points) What is the above expression's value?

(d) (5 points)



Complete the blanks below regarding the tree shown above, which uses the arithmethic operators +, ∗, and /:

- The _____ operator is the first operation to execute.
- The _____ operator is the last operation to execute.

2. (30 points)

(a) (10 points) Complete the code below so that it prints `true` if `a` is at least as large as the sum of `b` and `c`. Otherwise it should print `false`.

```
int a = ap.nextInt("Value for a?");
int b = ap.nextInt("Value for b?");
int c = ap.nextInt("Value for c?");
```

Continued on next page...

(b) (10 points) Complete the code below so that it prints the sum of `N` random numbers, with each random number chosen by a call to `Math.random()`. Do not use any arrays!

```
int N = ap.nextInt("How many random numbers?");
```

(c) (10 points) Complete the code below so that it prints the minimum of `N` random numbers, with each random number chosen by a call to `Math.random()`. Do not use any arrays!

```
int N = ap.nextInt("How many random numbers?");
```

3. (25 points) Consider the following code, which is the basis for all parts of this question:

```
int N = ap.nextInt();  // prompt for N
double[] nums = new double[N];
for (int i=0; i < N; ++i) {
   double r = 1.5 * Math.random();
   nums[i] = r;
}
```

(a) (8 points) Consider the following values:

$$-2.0 \qquad -1.0 \qquad -0.5 \qquad 0.0 \qquad 0.5 \qquad 1.0 \qquad 1.5 \qquad 2.0$$

Circle all of the values above that could possibly be assigned to `r` as the above code executes.

Continued on next page...

(b) (17 points) Suppose we are interested in the number of values placed in the array `nums` that have value at or below 0.5. This array has already been computed, as shown on the preceding page. Do not recompute it!

Below, you write code that executes after the code shown on the previous page. Declare and initialize a variable named `count` and write code below the declaration so that when your code is over, `count` is the number of elements in `nums` that have value at or below 0.5. Do not print anything out. For example, if the array had the values 0.6, 0.4, 0.6, and 0.5, then your code should compute `count` as 2.

4. (25 points)

   (a) (12 points) Your task here is to create a `String` array of 1000 elements, where each element is randomly assigned one of the following `String`s:

   - `"heads"`
   - `"tails"`

   Your score will depend on correct instantiation of the array and code that should, on average, cause equal numbers of each of the above `String`s to populate the array.

Continued on next page...

(b) (8 points) Having computed the above array, below you will write code that will randomly change your array by replacing some of the entries with the string `"side"`. Write your code so that, on average, 1 in 100 of the entries will be randomly changed from whatever they were before to `"side"`.

(c) (5 points) Now take your array and write code to iterate over it, to compute `String tosses` as the concatenation of its elements, in the order in which they occur in the array. It is not necessary to put any blank spaces between the entries as you concatentate them.