COS 126	Intro to CS	Fall 2015

Midterm 2 Written Exam

There are ten questions on this exam, weighted as indicated at the top of each page. There is one question per lecture, numbered corresponding to the lectures, *not in order of difficulty.* If a question seems difficult to you, skip it and come back to it.

Policies. The exam is closed book, though you are allowed to use a single-page two-sided hand-written cheatsheet. No calculators or other electronic devices are permitted. Give your answers and show your work in the space provided. You have 50 minutes to complete the test. **This exam is preprocessed by computer.** If you use pencil (and eraser), write darkly. Write all answers inside the designated rectangles. Do not write on corner marks.

This page. *Print your name, NetID; fill in your precept on this page*; write and sign the Honor Code pledge.

Discussing this exam. As you know, discussing the contents of this exam before solutions have been posted is a serious violation of the Honor Code.

"*I pledge my honor that I have not violated the Honor Code during this examination.*" NONE

Name:

NetID:

Room:

	P01	P01A	P01B	P02	P02A	P02B	P03	P04	P05	P05A	P06	P11	P12	P13	P14
Precept:	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

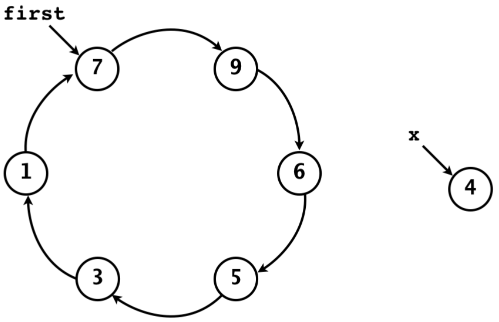
P01	T/Th 12:30pm	Dan Leyzberg
P01A	T/Th 12:30pm	Ted Brundage
P01B	T/Th 12:30pm	Doug Clark
P02	T/Th 1:30pm	Jérémie Lumbroso
P02A	T/Th 1:30pm	Doug Clark
P02B	T/Th 1:30pm	Andrea LaPaugh
P03	T/Th 2:30pm	Jérémie Lumbroso
P04	T/Th 7:30pm	Ming-Yee Tsang
P05	W/F 1:30pm	Stephen Cook
P05A	W/F 1:30pm	Lawrence Lin
P06	W/F 2:30pm	Alan Kaplan
P11	T/Th 1:30pm	Dan Leyzberg
P12	W/F 1:30pm	Donna Gabai
P13	T/Th 3:00pm	Alan Kaplan
P14	W/F 3:00pm	Donna Gabai

Q11. Searching and Sorting (7 points). To the right of each option, fill in the circle corresponding to the one-word characterization that best describes the order of growth of the ***worst-case*** running time. *Correct answers are worth 1 point; incorrect answers are worth -1 point; blank answers are worth 0 points.*

	logarithmic ($\log n$)	linear (n)	linearithmic ($n \log n$)	quadratic (n^2)
A. mergesort	○	○	○	○
B. merge	○	○	○	○
C. binary search	○	○	○	○
D. BST search	○	○	○	○
E. insertion sort	○	○	○	○
F. sequential search	○	○	○	○
G. bubble sort	○	○	○	○

Q12. Linked Structures (7 points). Suppose that the `Node` data type is defined as:

```
private class Node {
    private int item;
    private Node next;
}
```



and that `first` is a variable of type `Node` that refers to one node in a circular linked list. Let `x` be a variable that refers to a newly created `Node`.

```
Node x = new Node();
x.item = 4;
```

For each of the **independent** code fragments below, fill in the circle that describes its effect. Fill in "neither" if the code invalidates the property that `first` refers to a node in a circular linked list. *Correct answers are worth 1 point; incorrect answers are worth -1 point; blank answers are worth 0 points.*

	inserts `x` after `first`	inserts `x` before `first`	neither

`x.next = first.next;`
`first.next = x;`
	○	○	○

`Node z = first;`
`while(z.next != first) z = z.next;`
`z.next = x;`
`x.next = first;`
	○	○	○

`first.next = x;`
`x.next = first.next;`
	○	○	○

`x = first;`
`x.next = first;`
	○	○	○

`first.next = x;`	○	○	○

`Node z = first;`
`x.next = first.next;`
`z.next = x;`
	○	○	○

`x.item = first.item;`
`first.item = x.item;`
`first.next = x;`
	○	○	○

Q13. Binary Search Trees (7 points). *Correct answers are worth 1 point; incorrect answers are worth -1 point; blank answers are worth 0 points.*

A. Suppose that we create a BST by inserting integers into an initially empty tree. For each of the following insertion orders, fill in the bubble corresponding to the height of the binary tree that is produced (max distance from the root to any node) when keys are inserted in that order into an initially empty tree. The first answer is provided for you.

	1	2	3	4	5	6	7
`10, 11, 20, 30, 32, 48, 60`	○	○	○	○	○	○	●
`10, 20, 32, 11, 30, 48, 60`	○	○	○	○	○	○	○
`30, 10, 20, 11, 60, 48, 32`	○	○	○	○	○	○	○
`20, 11, 10, 30, 32, 60, 48`	○	○	○	○	○	○	○

B. Suppose that you are searching for the key `70` in a binary search tree. In the following list, indicate for each sequence whether it could be the sequence of keys examined, by filling in the corresponding bubble.

	Possible	Not possible
`77, 41, 99, 20, 85, 70`	○	○
`99, 10, 80, 20, 60, 70`	○	○
`5, 10, 80, 40, 32, 50, 70`	○	○
`22, 58, 81, 70`	○	○

Q14. REs (7 points). Let $L = \{aaaba, aabaa, abbba, ababa, aaaaa\}$. For each of the regular expressions below fill in the only answer that applies. *Correct answers are worth 1 point; incorrect answers are worth -1 point; blank answers are worth 0 points.*

The possible options (and their shortnames) are:

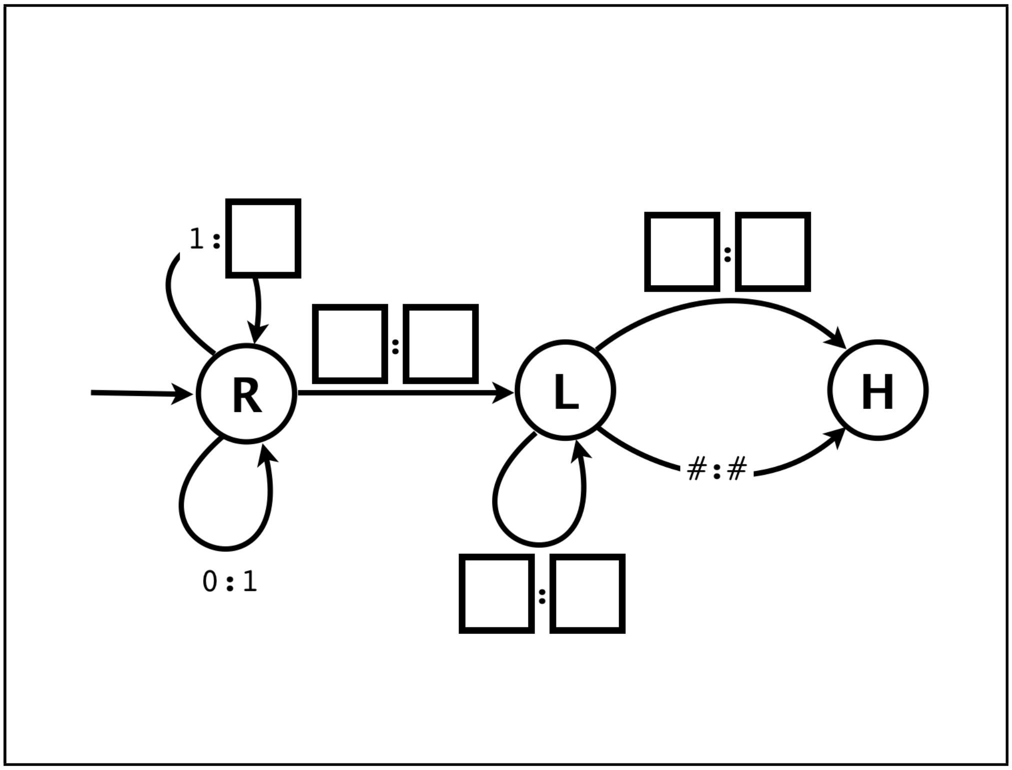
- [NONE] Matches no strings in L.
- [SOME] Matches only some strings in L and some other strings.
- [MORE] Matches all strings in L and some other strings.
- [EXACT] Matches all strings in L and no other strings.

<table>
<tr><th></th><th>NONE</th><th>SOME</th><th>MORE</th><th>EXACT</th></tr>
<tr><td><b>A.</b> a(a|b)*abb(a|b)*</td><td>○</td><td>○</td><td>○</td><td>○</td></tr>
<tr><td><b>B.</b> a(a|b)*a</td><td>○</td><td>○</td><td>○</td><td>○</td></tr>
<tr><td><b>C.</b> a*b*aba</td><td>○</td><td>○</td><td>○</td><td>○</td></tr>
<tr><td><b>D.</b> a((a*|b*)|(b*aba*))a</td><td>○</td><td>○</td><td>○</td><td>○</td></tr>
<tr><td><b>E.</b> a*b*aa*b*ba*a*a*b*b*a*a*b*</td><td>○</td><td>○</td><td>○</td><td>○</td></tr>
<tr><td><b>F.</b> (a|b)(a|b)(a|b)(a|b)a</td><td>○</td><td>○</td><td>○</td><td>○</td></tr>
<tr><td><b>G.</b> (a|aa|aaa)(ba|aa|bbb)a</td><td>○</td><td>○</td><td>○</td><td>○</td></tr>
</table>

Q15. Turing Machines (7 points). Below is an incomplete diagram of a Turing Machine. Complete the diagram so that it satisfies the following specification:

- Assume the initial tape consists of a binary string with infinitely many `#` symbols on both sides.
- Assume the initial head location is the leftmost bit of the binary string.
- Assume the initial state is the state labelled R on the left, and recall that the Turing Machine's first step is reading/writing, not moving the head.
- Interpret the input as a two's-complement binary integer. After the Turing machine halts, the binary integer that remains on the tape should be the negative of the original input. (Recall that computing the negative of a number involves flipping all of the bits and then adding one.)

Fill in exactly one symbol in each of the seven empty square boxes below so as to satisfy this specification. Do not add new states or new transitions, and do not use any tape symbols other than `#`, `0`, or `1`.



Q16. Intractability (7 points). For each of the computational problems below, indicate its difficulty by filling in the most appropriate choice among *True*, *False* or *Nobody Knows*. *Correct answers are worth 1 point; incorrect answers are worth -1 point; blank answers are worth 0 points.*

	True	False	Nobody Knows
A. If P is not equal to NP there is no polynomial-time algorithm for integer linear programming (ILP).	○	○	○
B. If P = NP, every problem in P is NP-complete.	○	○	○
C. If P = NP there is a polynomial-time algorithm for factoring.	○	○	○
D. No problem is in both P and NP.	○	○	○
E. There exists a deterministic Turing machine that can solve every problem in NP.	○	○	○
F. A Universal Turing Machine can simulate the operation of any Turing machine, including itself, in polynomial time.	○	○	○
G. No polynomial-time algorithm can solve the Halting Problem.	○	○	○

Q17. TOY (7 points). Fill in the blanks in the following table.

Hint: $7 \times 16^3 = 28,672$.

Note: The notation here and in the text and lectures is slightly different than the notation used in precepts and online. For example, `R[2] = R[3] + R[4]` means precisely the same thing as `R[2] <- R[3] + R[4]`.

hex	decimal integer	16-bit two's complement	TOY instruction pseudo-code
FFFE	-2	1111111111111110	R[F] = PC; PC = FE
1234	4,660		R[2] = R[3] + R[4]
1101	4,353	0001000100000001	
77FF	30,719	0111011111111111	R[7] = 00FF
FF01			
7A00		0111101000000000	

Q18. TOY Programming (7 points). Consider the following TOY program:

```
20: 81FF  R[1] = stdin
21:       SEE BELOW
22: 1211  R[2] = R[1] + R[1]
23: 0000  halt
```

To the right of each of the possibilities below for the instruction at `M[21]`, write the contents of `R[1]` when the machine halts after being started at `20` with `1111` on standard input. Your answers must each be four hex digits.

Note: The notation used here and in the text and lectures is slightly different than the notation used in precepts and online. For example, `R[1] = M[21]` means precisely the same thing as `R[1] <- mem[21]`.

Possibilities for `M[21]`	**Contents of `R[1]` when halting**
`21:  1111 R[1] = R[1] + R[1]`	
`21:  0000 halt`	
`21:  1211 R[2] = R[1] + R[1]`	
`21:  7100 R[1] = 0000`	
`21:  C023 PC = 23`	
`21:  8121 R[1] = M[21]`	
`21:  9122 M[22] = R[1]`	

Q19. Combinational Circuits (7 points). The *IS-XOR?* function of 3 boolean variables is **x**, **y** and **z** is 1 if and only if **x** XOR **y** == **z**.

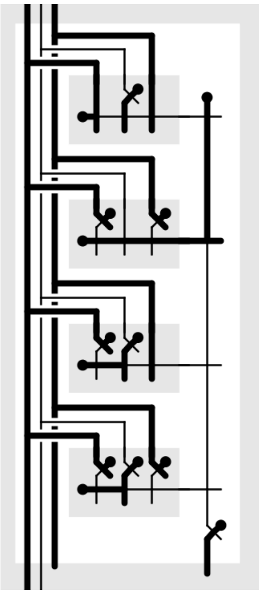
A. (2 points) Fill in the missing entries in this truth table for the 3-variable *IS-XOR?* function.

x	y	z	*IS-XOR?*
0	0	0	1
0	0	1	0
0	1	0	
0	1	1	
1	0	0	0
1	0	1	
1	1	0	
1	1	1	0

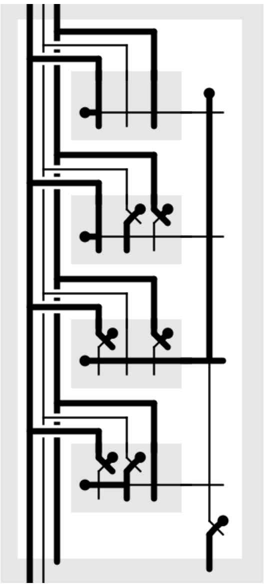
B. (3 points) In the box, write out the sum-of-products form of the 3-variable *IS-XOR?* function.

C. (2 points) Which of the circuits below is computing *IS-XOR?* for 3 variables with the inputs `1 0 1`? In each circuit, assume that the inputs `x y z` are provided in that order to the three lines at the upper left and the output is the line at the bottom right. Fill in the circle above the correct circuit. *Correct answers are worth 1 point; incorrect answers are worth -1 point; blank answers are worth 0 points.*

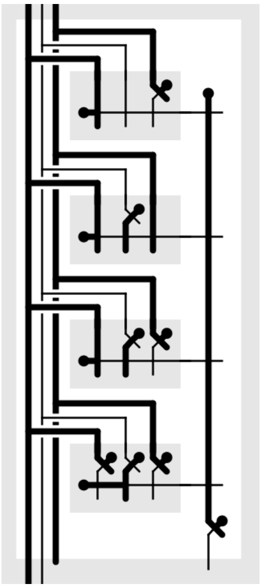
○



○



○



Q20. Central Processing Unit (7 points). Consider the following list of CPU components. In the box to the right of each option, write the letter corresponding to the description at right that best matches. Each letter should be used **at most** once. *Correct answers are worth 1 point.*

Bus connection □

ALU □

MUX □

IR □

Control line □

PC □

Clock □

A. Sequential circuit that carries information.

B. Combinational circuit that holds data.

C. Component input that determines behavior.

D. Holds address of current instruction.

E. Allows switching among component inputs.

F. Allows switching among component outputs.

G. Computes boolean function values.

H. Holds instruction being executed.

I. Path carrying data between components.

J. Sequential circuit that produces periodic pulses.

K. Memory unit extension.