COS126 **Written Exam 2** **Spring '18**

Instructions. This exam has 7 questions, worth 10 points each. You have 50 minutes.

Resources. You may reference your optional two-sided 8.5-by-11 handwritten "cheat sheet" during this exam. You may not use the textbook, your notes, or any electronic devices. You may not communicate with anyone except the course staff during this exam.

After this exam. Due to travel for extracurriculars and sports, some of your peers will take this exam next week. Do not discuss its contents with anyone who has not taken it yet.

This paper. Do not remove this copy of the exam from the exam room. You may fill in this page now.

NAME: ______________________

NETID: ______________________

PRECEPT: ______________________

EXAM ROOM: ______________________

"I pledge my honor that I will not violate the Honor Code during this examination."

____

____

SIGNATURE: ______________________

Question 1	Number Systems	10 points

Fill in the blanks in the table below by converting the given values between number systems.

decimal	**8-bit two's complement**	**hexadecimal**
126	0111 1110	7E
-126		
	1111 1111	
		1A
		F8
-2		

Question 2 **Regular Expressions** **10** points

Let L = { bbaaa, abbaa, bbbaa, abaaa }. The alphabet for L is {a, b}.

For each of the following regular expressions, choose one of the following:

- NONE -- Matches no strings in L.
- SOME -- Matches some, but not all, strings in L.
- EXACT -- Matches all strings in L and no other strings.
- MORE -- Matches all strings in L and some other strings.

		NONE	SOME	EXACT	MORE
1.	`.....`	○	○	○	○
2.	`.(ab)*`	○	○	○	○
3.	`(a*|b*)(aa)*`	○	○	○	○
4.	`ab*a*b*a*a*bab*`	○	○	○	○
5.	`(ab|bb)(baa|aaa)`	○	○	○	○
6.	`(a|ab|bbb)(bba|abb|a)`	○	○	○	○
7.	`(a|b)(ba|bb)(a|b)a`	○	○	○	○
8.	`(bbb|bba|abb|aba)..`	○	○	○	○
9.	`(b*|((ab)|(ba)))*`	○	○	○	○
10.	`((a|b)*|(aba*|..a))*`	○	○	○	○

Question 3	**TOY**	**10** points

Fill in the blanks in the following TOY program that reads two inputs from `StdIn`, multiplies them using repeated addition, and prints the result to `StdOut`.

```
10: 8AFF    reads from StdIn to R[A]

11: 8BFF    reads from StdIn to R[B]

12: 7C00    R[C] <- 0000

13: 7101    R[1] <- 0001

14: CA__    HINT: if R[A] == 0, we have our answer!

15: 1C__    HINT: add something to R[C]

16: 2A__    HINT: subtract something from R[A]

17: ____    HINT: we're not done yet!

18: 9CFF    prints R[C] to StdOut

19: 0000    halt
```

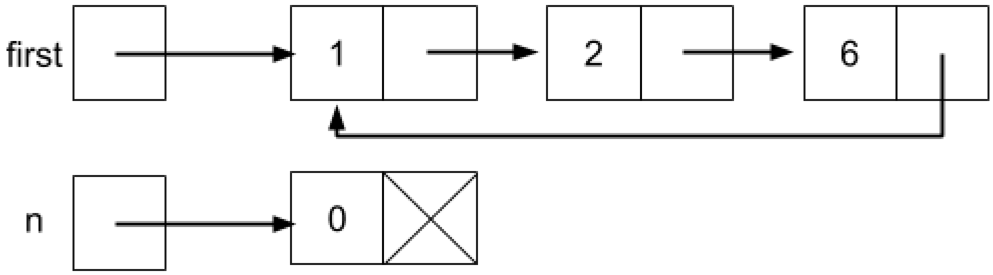
Question 4	**Theory**	**10** points

For each cell in the table below, select the response that best represents our current understanding. Assume that, as we suspect, the Church-Turing thesis holds.

	solvable in polynomial time	reduces to SAT in polynomial time	SAT reduces to it in polynomial time
SORT	○ true ○ false ○ unknown	○ true ○ false ○ unknown	○ true ○ false ○ unknown
FACTOR	○ true ○ false ○ unknown	○ true ○ false ○ unknown	○ true ○ false ○ unknown
PRIME	○ true ○ false ○ unknown	○ true ○ false ○ unknown	○ true ○ false ○ unknown
TSP	○ true ○ false ○ unknown	○ true ○ false ○ unknown	○ true ○ false ○ unknown
SAT	○ true ○ false ○ unknown	○ true ○ false ○ unknown	○ true ○ false ○ unknown
HALTING PROBLEM	○ true ○ false ○ unknown	OUTSIDE THE SCOPE OF THIS COURSE	OUTSIDE THE SCOPE OF THIS COURSE

Question 5 **Linked Lists** **10** points

Choose the correct sequence of Java instructions to insert a node into a circularly linked list. You have access to two node variables, `first` and `n`, as diagrammed below. `n.next` is initialized to `null`. Your answers must produce a circularly linked list. Treat each of the four parts below as independent.



Choose one letter per box below, in the correct order, to perform the following operations.
You may use each letter once, more than once, or not at all.

A. `first = n;`

B. `first.next = n;`

C. `first.next.next = n;`

D. `first.next.next.next = n;`

E. `first.next.next.next.next = n;`

F. `n.next = first;`

G. `n.next = first.next;`

H. `n.next = first.next.next;`

I. `n.next = first.next.next.next;`

J. `n.next = first.next.next.next.next;`

1. Insert n between the first node and the second node. □ □

2. Insert n after the last node. □ □

3. Insert n between the second node and the third node. □ □

4. Insert n before the first node. □ □ □

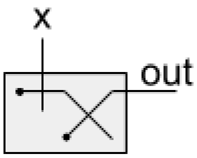
Question 6	**Runtime Analysis**	**10** points

Choose the order of growth of each of the following operations.

		1	logN	N	NlogN	N^2	N^3
1.	Best-case scenario for insertion sort.	○	○	○	○	○	○
2.	Best-case scenario for mergesort.	○	○	○	○	○	○
3.	Worst-case scenario for insertion sort.	○	○	○	○	○	○
4.	Worst-case scenario for mergesort.	○	○	○	○	○	○
5.	A program whose running time doubles (exactly) when the input size doubles.	○	○	○	○	○	○
6.	A program that performs an operation on all possible pairings of N elements.	○	○	○	○	○	○
7.	Finding the largest element in a linked list.	○	○	○	○	○	○
8.	Finding the median element in a sorted array.	○	○	○	○	○	○
9.	Finding the average of all the elements in a binary search tree.	○	○	○	○	○	○
10.	Finding the largest element in a balanced binary search tree.	○	○	○	○	○	○

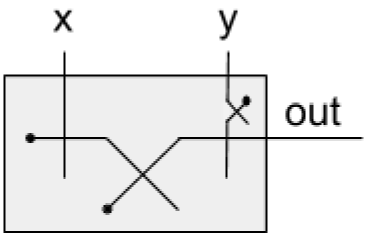
Question 7 **Circuits** **10** points

For each circuit below, complete the corresponding truth table and choose the best descriptor.



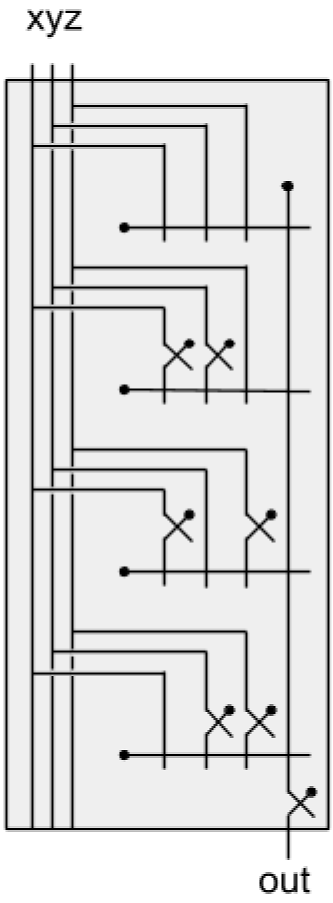
x	out
0	
1	

○ not ○ maj
○ xor ○ odd
○ and ○ even
○ none of the above



x	y	out
0	0	
0	1	
1	0	
1	1	

○ not ○ maj
○ xor ○ odd
○ and ○ even
○ none of the above



x	y	z	out
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

○ not ○ maj
○ xor ○ odd
○ and ○ even
○ none of the above

TOY REFERENCE CARD

```
INSTRUCTION FORMATS
             | . . . . | . . . . | . . . . | . . . .|
  Format RR: | opcode  |    d    |    s    |    t   |  (0-6, A-B)
  Format A:  | opcode  |    d    |       addr       |  (7-9, C-F)


ARITHMETIC and LOGICAL operations
    1: add              R[d] <- R[s] +  R[t]
    2: subtract         R[d] <- R[s] -  R[t]
    3: and              R[d] <- R[s] &  R[t]
    4: xor              R[d] <- R[s] ^  R[t]
    5: shift left       R[d] <- R[s] << R[t]
    6: shift right      R[d] <- R[s] >> R[t]

TRANSFER between registers and memory
    7: load address     R[d] <- addr
    8: load             R[d] <- M[addr]
    9: store            M[addr] <- R[d]
    A: load indirect    R[d] <- M[R[t]]
    B: store indirect   M[R[t]] <- R[d]

CONTROL
    0: halt             halt
    C: branch zero      if (R[d] == 0) PC <- addr
    D: branch positive  if (R[d] >  0) PC <- addr
    E: jump register    PC <- R[d]
    F: jump and link    R[d] <- PC; PC <- addr


Register 0 always reads 0.
Loads from M[FF] come from stdin.
Stores to M[FF] go to stdout.

16-bit registers (two's complement)
16-bit memory locations
 8-bit program counter
```

You may tear this page out and use it as scratch paper.

If you do, fill in the blanks below and return it *inside* your exam.

NAME: ______________________________

NETID: ______________________________

PRECEPT: ______________________________

EXAM ROOM: ______________________________