COS126 **Written Exam 2** **Fall '21**

Instructions. This exam has nine (9) questions worth a total of seventy (70) points. You have fifty (50) minutes.

This exam is preprocessed by computer. Write neatly and legibly. If you use a pencil, write darkly. Write all answers inside the designated rectangles. Fill in circles completely: ● (not ✔ or ✘). If you change your mind, you must erase completely and fill in another circle!

Resources. The exam is closed book, except that you are allowed to use a one-page reference sheet (8.5-by-11 paper, both sides, in your own handwriting). No electronic devices are permitted.

Discussing this exam. Discussing the contents of this exam before solutions have been posted is a violation of the Honor Code.

This exam. Do not remove this exam paper from this room. Write your name, NetID, precept, and the room in which you are taking the exam in the space below. Also, write and sign the Honor Code pledge. You may enter this information now. Again, please write neatly and legibly.

NAME:

NETID (not email alias):

PRECEPT:

EXAM ROOM: ○ McCosh 50 ○ McDonnell A02 ○ Friend 101 ○ CS 301/302 ○ OTHER ____________

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

____

____

SIGNATURE: ______________________________________

Question 1 — **Recursion** — **6** points

Indicate whether each of the following statements about recursion is true or false by completely filling in the appropriate circle.

	Statement	**True**	**False**
1.1	A computer with more memory than another computer may be able to compute a recursive function that does not have a base case.	◯	◯
1.2	All recursive functions that have base cases are guaranteed to terminate.	◯	◯
1.3	All recursive functions must only have one base case.	◯	◯

Question 2 — **Number Representation** — **6** points

Suppose that you have a 16-bit computer word, using two's-complement representation for integers. In the rectangles to the right, write the **4-digit** hexadecimal representation of each entity described on the left. DO NOT WRITE OUTSIDE THE RECTANGLE.

Note: ^ means XOR

	Entity	**4-digit Hex**
2.1	Decimal number `257`	
2.2	Decimal number `-257`	
2.3	The hex expression: `00FF ^ C0DE ^ 00FF`	

Question 3 **TOY** **6** points

For each TOY program below, indicate which operation (A - I) is performed. Assume each TOY program is independent of one another. Ignore integer overflow. Select the best answer for each program. You may use each operation (A - I) once, more than once, or not at all.

A. No-op (no change) to R[A].	**D.** Divide R[A] by 2.	**G.** Set R[A] to 0.
B. Flip R[A]'s bits (0s to 1s and 1s to 0s).	**E.** Swaps R[A] with R[B].	**H.** XOR R[A] with all 1s.
C. Multiply R[A] by 2.	**F.** Add 1 to R[A].	**I.** Negates R[A].

3.1
```
2BBB   R[B] <- R[B] - R[B]
2ABA   R[A] <- R[B] - R[A]
```
A◯ B◯ C◯ D◯ E◯ F◯ G◯ H◯ I◯

3.2
```
7B01   R[B] <- 0001
2B0B   R[B] <- -R[B]
3AAB   R[A] <- R[A] & R[B]
```
A◯ B◯ C◯ D◯ E◯ F◯ G◯ H◯ I◯

3.3
```
4AAB   R[A] <- R[A] ^ R[B]
4BBA   R[B] <- R[B] ^ R[A]
4AAB   R[A] <- R[A] ^ R[B]
```
A◯ B◯ C◯ D◯ E◯ F◯ G◯ H◯ I◯

```
INSTRUCTION FORMATS                         TRANSFER between registers and memory

           | .... | .... | .... | .... |    7: load address     R[d] <- addr
Format RR: |  op  |  d   |  s   |  t   |    8: load             R[d] <- M[addr]
Format A:  |  op  |  d   |     addr    |    9: store            M[addr] <- R[d]
                                            A: load indirect    R[d] <- M[R[t]]
                                            B: store indirect   M[R[t]] <- R[d]
ARITHMETIC and LOGICAL operations
  1: add           R[d] <- R[s] +  R[t]     CONTROL

  2: subtract      R[d] <- R[s] -  R[t]     0: halt             halt
  3: and           R[d] <- R[s] &  R[t]     C: branch zero      if (R[d] == 0) PC <- addr
  4: xor           R[d] <- R[s] ^  R[t]     D: branch positive  if (R[d] >  0) PC <- addr

  5: shift left    R[d] <- R[s] << R[t]     E: jump register    PC <- R[d]
  6: shift right   R[d] <- R[s] >> R[t]     F: jump and link    R[d] <- PC; PC <- addr
```

Question 4 **Java Terminology** **10** points

Fill in the circle corresponding to the letter that best matches the description (A - M). Use each letter at most once.

4.1	`static`	A◯	B◯	C◯	D◯	E◯	F◯	G◯	H◯	I◯	J◯	K◯	L◯	M◯
4.2	`generic`	A◯	B◯	C◯	D◯	E◯	F◯	G◯	H◯	I◯	J◯	K◯	L◯	M◯
4.3	`public`	A◯	B◯	C◯	D◯	E◯	F◯	G◯	H◯	I◯	J◯	K◯	L◯	M◯
4.4	`private`	A◯	B◯	C◯	D◯	E◯	F◯	G◯	H◯	I◯	J◯	K◯	L◯	M◯
4.5	`this`	A◯	B◯	C◯	D◯	E◯	F◯	G◯	H◯	I◯	J◯	K◯	L◯	M◯
4.6	`new`	A◯	B◯	C◯	D◯	E◯	F◯	G◯	H◯	I◯	J◯	K◯	L◯	M◯
4.7	`void`	A◯	B◯	C◯	D◯	E◯	F◯	G◯	H◯	I◯	J◯	K◯	L◯	M◯
4.8	`class`	A◯	B◯	C◯	D◯	E◯	F◯	G◯	H◯	I◯	J◯	K◯	L◯	M◯
4.9	`null`	A◯	B◯	C◯	D◯	E◯	F◯	G◯	H◯	I◯	J◯	K◯	L◯	M◯
4.10	`main`	A◯	B◯	C◯	D◯	E◯	F◯	G◯	H◯	I◯	J◯	K◯	L◯	M◯

A. Indicates field or method belongs to the entire class rather than an individual instance.
B. Signifies a reference to the invoking object within an instance method.
C. A group of related methods and variables.
D. Automatic cast from primitive type to wrapper type.
E. Signifies that a method can be called directly by another method in a different file.
F. Invokes a constructor.
G. Triggers an exception.
H. Signifies a reference to no object.
I. A parameterized data type.
J. Automatic cast from wrapper type to primitive type.
K. Used to deny clients access to the data type representation.
L. Signifies that a method does not return a value.
M. Identifies the method that is automatically invoked when you run a program.

Question 5 **Performance** **10** points

5.1 To determine the runtime of their algorithm, a Java programmer performs the following tests for various input sizes. Fill in the circle below that describes the best hypothesis for the order of growth of the running time of this program as a function of input size.

Input size	**Runtime**
400	1 second
1600	15 seconds
6400	242 seconds

○ Logarithmic ○ Linear ○ Linearithmic ○ Quadratic ○ Cubic ○ Exponential

For each of the following operations, fill in the circle that corresponds to the order of growth of the running time for each operation.

5.2 Inserting a number into the correct place in a linked list of N sorted numbers:

○ Logarithmic ○ Linear ○ Linearithmic ○ Quadratic ○ Cubic ○ Exponential

5.3. Performing merge sort on an array of size N:

○ Logarithmic ○ Linear ○ Linearithmic ○ Quadratic ○ Cubic ○ Exponential

5.4 Inserting a word in a balanced binary search tree of size N:

○ Logarithmic ○ Linear ○ Linearithmic ○ Quadratic ○ Cubic ○ Exponential

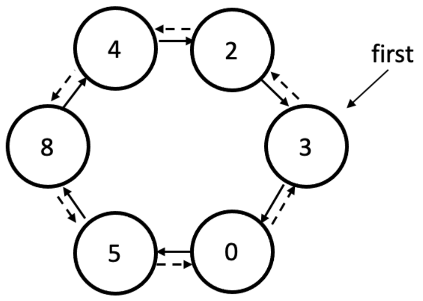
5.5 Performing N binary searches on a sorted array of size N:

○ Logarithmic ○ Linear ○ Linearithmic ○ Quadratic ○ Cubic ○ Exponential

Question 6 **Linked Lists** **9** points

Suppose that the Node data type is defined as

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



and that `first` is a variable of type `Node` that refers to the *first* node in a doubly circularly linked list. Suppose that the doubly circularly linked list is as above, with the *clockwise* solid arrows depicting the next field and the *counterclockwise* dashed arrows depicting the prev field.

Choose the output for each of the following pieces of code from the choices given below.

6.1
```
Node current = first;
do {
   StdOut.println(current.item);
   current=current.next.next;
} while (current != first);
```

- ○ 3 0 5 8 4 2
- ○ 3 2 4 8 5 0
- ○ 3 0 5 8 4
- ○ 3 2 4 8 5
- ○ 0 5 8 4 2
- ○ 2 4 8 5 0
- ○ 0 5 8 4
- ○ 3 5 4
- ○ 3 4 5
- ○ 3 4 5 3
- ○ 0 2 4 8 5

6.2
```
Node current = first;
while (current.prev != first) {
     StdOut.println(current.item);
     current = current.prev;
}
```

- ○ 3 0 5 8 4 2
- ○ 3 2 4 8 5 0
- ○ 3 0 5 8 4
- ○ 3 2 4 8 5
- ○ 0 5 8 4 2
- ○ 2 4 8 5 0
- ○ 0 5 8 4
- ○ 3 5 4
- ○ 3 4 5
- ○ 3 4 5 3
- ○ 0 2 4 8 5

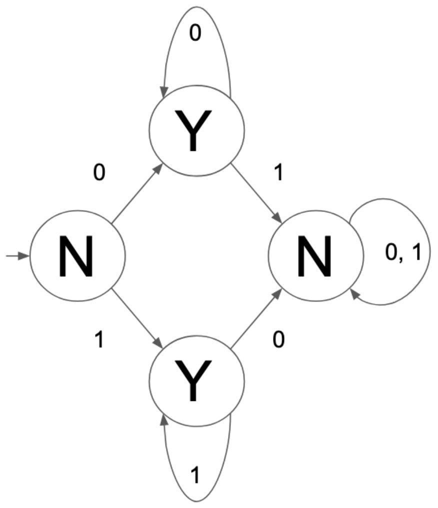
6.3
```
first.prev.next = first.next;
first.next.prev = first.prev;
first = first.next;
Node current = first;
do{
     StdOut.println(current.item);
     current = current.prev;
} while (current != first);
```

- ○ 3 0 5 8 4 2
- ○ 3 2 4 8 5 0
- ○ 3 0 5 8 4
- ○ 3 2 4 8 5
- ○ 0 5 8 4 2
- ○ 2 4 8 5 0
- ○ 0 5 8 4
- ○ 3 5 4
- ○ 3 4 5
- ○ 3 4 5 3
- ○ 0 2 4 8 5

Question 7 **Regular Expressions/DFAs** **6** points

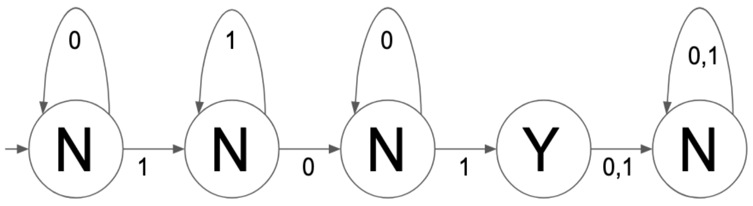
Fill in the circle of the correct regular expression corresponding to each DFA.

7.1



- ◯ `(0|1)*`
- ◯ `0*|1*`
- ◯ `0*1*`
- ◯ `010`
- ◯ `None of the above`

7.2



- ◯ `0*11*00*1(0|1)*`
- ◯ `11*00*1`
- ◯ `(0|1)(0*|1*)`
- ◯ `0*11*00*1`
- ◯ None of the above

Question 8	**Theory of Computation**	**10** points

Suppose *X* is a problem we wish to solve. For each statement, choose the best description from the choices given below. (That is, for each statement, if more than one of A - E is true, choose from F - J.) You may use a description (A - J) once, more than once, or not at all. Fill in ONE circle completely.

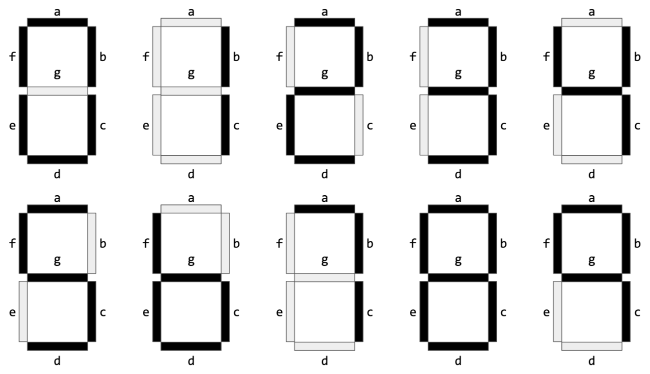
	Statement	**Best Description**
8.1	There is a Turing Machine that solves (i.e., always outputs a correct solution for) Problem *X.*	A◯ B◯ C◯ D◯ E◯ F◯ G◯ H◯ I◯ J◯
8.2	The solution to problem *X* can be checked for correctness in polynomial time in the size of its input.	A◯ B◯ C◯ D◯ E◯ F◯ G◯ H◯ I◯ J◯
8.3	A cubic algorithm exists that solves Problem *X.*	A◯ B◯ C◯ D◯ E◯ F◯ G◯ H◯ I◯ J◯
8.4	*X* is in NP and SAT reduces to Problem *X.*	A◯ B◯ C◯ D◯ E◯ F◯ G◯ H◯ I◯ J◯
8.5	An algorithm that solves Problem *X* implies an algorithm for the Halting Problem.	A◯ B◯ C◯ D◯ E◯ F◯ G◯ H◯ I◯ J◯

A. *X* is undecidable.
B. *X* is decidable.
C. *X* is in NP.
D. *X* is NP-complete.
E. *X* is in P.

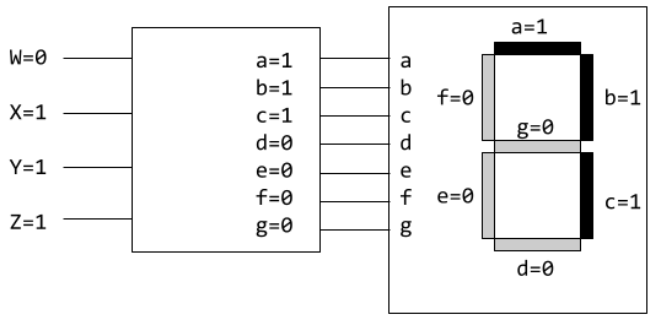
F. Both B and C are true.
G. Both B and D are true.
H. All of B, C, D, and E are true.
I. All of B, C, and E are true.
J. All of B, C, and D are true.

Question 9	**Combinational Circuits**	**7** points

Consider a common 7 LED segment display, found in clocks, timers, thermometers, busses, trains, etc. Here are the seven segment displays for the decimal numbers 0 - 9 (on the right), with each segment labeled with a, b, c, d, e, f, and g. The decimal numbers on the top row are 0, 1, 2, 3, 4, 5 and on the bottom row are 5, 6, 7, 8, 9.



A binary coded display (BCD) to seven segment decoder is a circuit that has four input lines (`W, X, Y, Z`) and seven output lines (`a, b, c, d, e, f, g`). The output lines are connected to a seven segment LED display, which displays the decimal number depending upon inputs. For example, the decimal number **7** in binary is `0111` or `W=0, X=1, Y=1, Z=1` and the output should be `1110000` or `a=1, b=1, c=1, d=0, e=0, f=0, g=0` (on the right), which turns on segments `a, b` and `c`.



9.1 Complete the truth table for turning on the **`e`** segment - fill in ONE circle for a `0` or `1`.

decimal	W	X	Y	Z	e 0	e 1
0	0	0	0	0	○	○
1	0	0	0	1	○	○
2	0	0	1	0	○	○
3	0	0	1	1	○	○
4	0	1	0	0	○	○
5	0	1	0	1	○	○
6	0	1	1	0	○	○
7	0	1	1	1	○	○
8	1	0	0	0	○	○
9	1	0	0	1	○	○

9.2 How many 4-input `AND` gates and 2-input `OR` gates are needed to build a circuit based on the sum of products formula for turning on the **`e`** segment? Do not simplify. Fill in one circle below:

○ 5 AND, 2 OR gates

○ 4 AND, 2 OR gates

○ 4 AND, 3 OR gates

○ 10 AND, 9 OR gates

○ I cannot decide (partial credit)