COS 126	General Computer Science	Spring 2016

Written Exam 2

This exam has 10 questions (including question 0) worth a total of 70 points. You have 50 minutes.

Policies. The exam is closed book, except that you are allowed to use a one page cheatsheet (8.5-by-11 paper, both sides, in your own handwriting). No calculators or other electronic devices are permitted. *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.*

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

This exam. You must turn in this exam. *Print your name, NetID, and precept in the space below. Write and sign the Honor Code pledge.*

Name:

NetID:

Precept:

"I pledge my honor that I have not violated the Honor Code during this examination."

0. **Miscellaneous. (2 point)**

   (a) Write your name and Princeton NetID in the space provided on the front of this exam, and mark your precept number.

   (b) Write and sign the honor code on the front of this exam.

1. **Java keywords. (8 points)**

   For each description on the left, choose the best-matching Java keyword on the right. You may use each letter any number of times.

___ Indicates that there is one variable per class (and not one variable per object of the class)

___ Signifies that a method *can* be called directly by another method in a different file

___ Signifies that an instance variable *cannot* be accessed directly by code in a different file

___ Signifies that a method does not return a value

___ Signifies a reference to no object

___ Signifies a reference to the invoking object, during a method call

___ Invokes a constructor

___ Triggers an exception

A. `class`

B. `final`

C. `new`

D. `null`

E. `private`

F. `public`

G. `return`

H. `static`

I. `this`

J. `throw`

K. `void`

2. **Properties of objects. (8 points)**

   Which of the following statements are true for *Java classes*. Mark all that apply.

   (a) A data type is a set of values and a set of operations on those values.

   (b) A class can define more than one instance method, but each instance method must have a different name.

   (c) A class can define at most one constructor.

   (d) A `.java` file may not include more than one class definition.

   (e) An instance method can refer to the private instance variables of the invoking object, but cannot refer to the private instance variables of any other object.

   (f) A class can contain either static methods or instance methods, but not both.

   (g) It is a compile-time error to define a local variable with the same name as an instance variable.

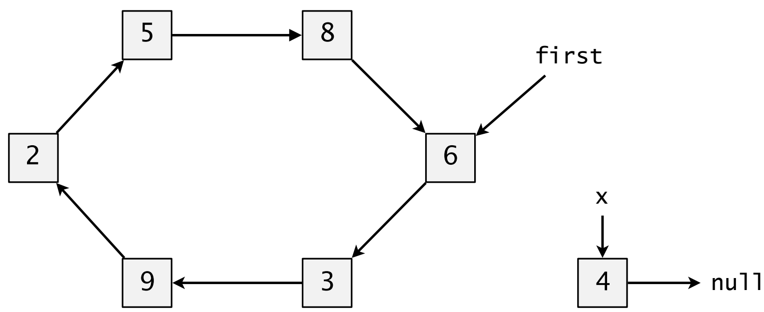
   (h) If you pass an object reference of type `Picture` to a method, the method *cannot* change the caller's object reference (for example, to make it refer to a different `Picture`), but it can change the value of the object (for example, by invoking the `set()` method to change a pixel's color).

3. **Linked structures. (8 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, as illustrated here:



Let `x` be a variable that refers to a newly created node.

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

Independently, for each code fragment on the left, pick the best-matching description on the right. You may use each letter any number of times.

___ `first.next = first.next.next;`

___ `x.next = first.next;`
`first.next = x;`

___ `x.next = first.next.next;`
`first.next = x;`

___ `x.next.next = first.next.next;`
`first.next = x;`

A. no change

B. deletes 6

C. deletes 3

D. deletes 9

E. inserts 4 after 6

F. inserts 4 after 3

G. replaces 3 with 4

H. replaces 6 with 4

I. `first` no longer refers to a circular linked list

J. run-time error

4. **Analysis of algorithms. (6 points)**

   Consider the following two functions. Assume that `Merge.sort()` is a function that uses the mergesort algorithm (the version from the textbook and lecture) to rearrange the n elements of its argument array into ascending order.

```
public static boolean method1(int[] a) {
    int n = a.length;
    for (int i = 0; i < n; i++)
        for (int j = i+1; j < n; j++)
            if (a[i] == a[j]) return true;
    return false;
}

public static boolean method2(int[] a) {
    int n = a.length;
    Merge.sort(a);
    for (int i = 1; i < n; i++)
        if (a[i] == a[i-1]) return true;
    return false;
}
```

   (a) For each term on the left, select the best-matching term from the right. You may use each letter any number of times.

___ Worst-case running time of `method1()`	A. 1
___ Best-case running time of `method1()`	B. $\log n$
___ Worst-case running time of `method2()`	C. n
___ Best-case running time of `method2()`	D. $n \log n$
	E. n^2
	F. $n^2 \log n$

   (b) Is there an input array for which `method1()` and `method2()` return different values? If so, give such an array in the box provided.

5. **Regular expressions and DFAs. (8 points)**

For each formal language on the left, choose the best-matching RE or DFA on the right.

___ All binary strings whose length is odd

___ All binary strings

___ All binary strings that contains three consecutive `b`s

___ All binary strings with an equal number of `a`s and `b`s

A. `(a*b*)*`

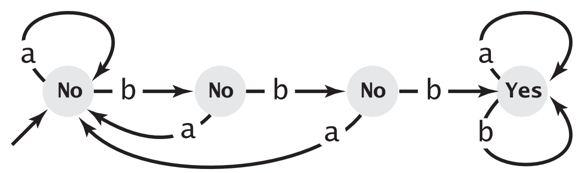
B. `(a|b)(a|b)(a|b)*`

C. `(a|b)((a|b)(a|b))*`

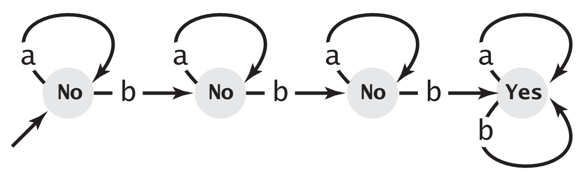
D. `(a|b)(ab|ba)*`

E. `(a*b*)bbb(a*b*)`

F.



G.



H. none of the above

6. **TOY. (8 points)**

Consider the following TOY program:

```
10: 7B00  R[B] <- 0000
11: 8AFF  R[A] from stdin
12:       SEE BELOW
13:       SEE BELOW
14: 1B0A  R[B] <- R[A]
15: C011  PC <- 11
16: 0000  halt
```

Assume the the following integers are available on standard input:

```
1111  3333  5555  2222  1111  0000  AAAA  CCCC  BBBB  BBBB  CCCC  AAAA
```

and that the TOY machine starts at `10`. For each of the following possible initial values of memory locations `12` and `13`, mark the value of `R[A]` upon termination.

(a)
```
12: 0000   halt
13: 1AAA   R[A] <- R[A] + R[A]
```

(b)
```
12: 1AAA
13: DA16
```

(c)
```
12: 2CAB   R[C] <- R[A] - R[B]
13: CC16   if (R[C] == 0) PC <- 16
```

7. **Theory of computing. (8 points)**

For each statement on the left, pick the best-matching description on the right.
You may use each letter any number of times.

___ There exists a formal language that can be decided by a TOY machine but not by a Turing machine.

___ There exists a formal language that can be decided in polynomial time by a Java program but not by a Turing machine.

___ There exists an exponential-time algorithm to solve the halting problem.

___ There exists a physically realizable computing device that can solve the halting problem.

___ There exists a polynomial-time algorithm for TSP.

___ There does not exist a polynomial-time algorithm for TSP.

___ SAT polynomial-time reduces to FACTOR.

___ FACTOR polynomial-time reduces to SAT.

A. known to be true

B. known to be false

C. if true, would falsify the Church–Turing thesis

D. if true, would prove the Church–Turing thesis

E. if true, would imply $P = NP$

F. if true, would imply $P \neq NP$

G. if true, would imply that FACTOR is NP-complete

8. **Circuits. (6 points)**

The 3-bit *minority* function $f(x, y, z)$ is 1 if at most one of its inputs is 1, and 0 otherwise. Which of the following represent the minority function? Check all that apply.

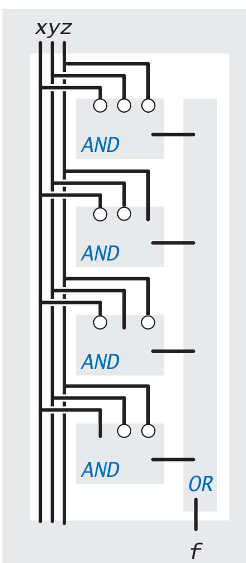
(a)

x	y	z	f
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

(b) $f = x'y'z + x'yz' + xy'z'$

(c) $f = x'y' + y'z' + x'z'$

(d)



(e)
```
public static boolean f(boolean x, boolean y, boolean z) {
    return !(x && y || x && z || y && z);
}
```

(f)
```
public static boolean f(boolean x, boolean y, boolean z) {
    if (x) return !(y || z);
    if (y) return !z;
    return true;
}
```

9. **Powers of 2. (8 points)**

For each description on the left, choose the best-matching power of 2 on the right. You may use each letter any number of times.

___ 1,024	A. 2^0
___ Number of 0s in the 16-bit two's complement representation of the decimal number -16.	B. 2^1
	C. 2^2
___ Number of TOY instruction types	D. 2^3
___ Total number of bits of main memory in TOY (including bits for memory location `FF`)	E. 2^4
___ Number of distinct non-negative values representable in Java's `int` data type	F. 2^5
___ Multiplicative factor by which the running time increases when you quadruple the size of the input of a quadratic algorithm	G. 2^6
	H. 2^8
___ Number of output wires in a decoder that has 8 input wires	I. 2^{10}
___ Number of multiway-OR gates in our 32-bit ripple–carry adder	J. 2^{12}
	K. 2^{15}
	L. 2^{16}
	M. 2^{31}
	N. 2^{32}
	O. 2^{64}

TOY REFERENCE CARD

```
INSTRUCTION FORMATS

             | . . . . | . . . . | . . . . | . . . .|
  Format RR: | opcode  |    d    |    s    |    t    |  (1-6, 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 (using two's complement arithmetic)
16-bit memory locations
 8-bit program counter
```