NAME: login ID:
precept:

COS 126 Written Exam 2, Spring 2008

This test is 8 questions, weighted as indicated. The exam is closed book, except that you are allowed to use a one page cheatsheet. No calculators or other electronic devices are permitted. Give your answers and show your work in the space provided. *Put your name, login ID, and precept number on this page* (*now*), and write out and sign the Honor Code pledge before turning in the test. You have 50 minutes to complete the test.

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

_____

Signature

1	/5
2	/5
3	/6
4	/12
5	/7
6	/14
7	/12
8	/14
TOTAL	/75

1. **Binary Search** (5 points). Circle the statements below that explain why using binary search makes sense (more than one might apply).

   a) It is typically much faster than sequential search.

   b) Everything in a computer is encoded in binary.

   c) When data is sorted, it is easy to cut the amount of data to search in half.

   d) It makes inserting and deleting data items much easier.

   e) It is not a built-in Java library function.

2. **Postfix** (5 points). Give the value of the postfix expression `6 5 4 * 3 2 1 + * + +`

3. **Regular expressions** (6 points). Circle the strings below that are in the language described by the regular expression:

   ```
   a*bb(ab|ba)*
   ```

   a) `abb`

   b) `abba`

   c) `aaba`

   d) `bbbaab`

   e) `cbb`

   f) `bbababbab`

4. **Linked structures** (12 points). Consider the following data type methods which use a helper `Node` data type to find whether a given integer `key` is found in a linked structure. Assume that `start` is an instance variable that refers to a fixed `Node` in the linked structure.

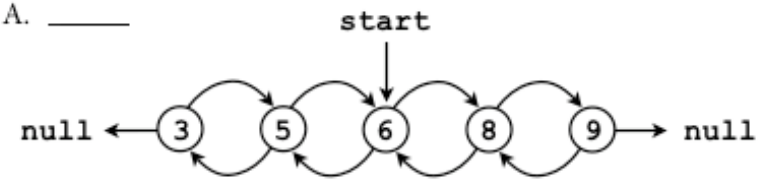
```
public boolean f(int key)
{
   Node x = start;
   while (x != null)
   {
      if (x.key == key)
         return true;
      x = x.right;
   }
   return false;
}
```

```
public boolean g(int key)
{
   Node x = start;
   while (x != null)
   {
      if x.key == key) return true;
      if (x.key > key) x = x.left;
      else             x = x.right;
   }
   return false;
}
```

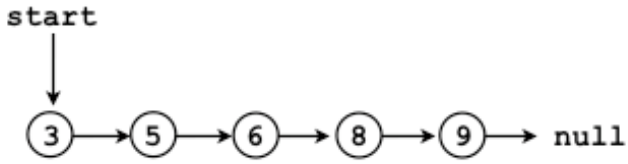
```
public boolean h(int key)
{
   Node x = start;
   while (true)
   {
      x = x.right;
      if (x.key == key) return true;
      if (x == start) break;
   }
   return false;
}
```

Match each of the following linked structures with the corresponding method by writing the method name in the blank provided or leave it blank if there is no match. The nodes in structures A. B. and D. are in ascending order. Structure C. is a binary search tree: each element is bigger than its left child and smaller than its right child.

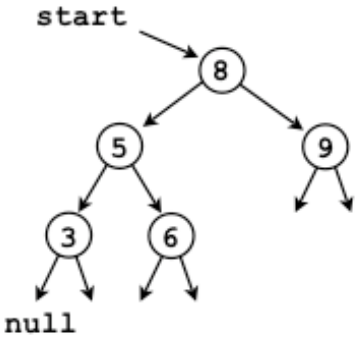
A. _____



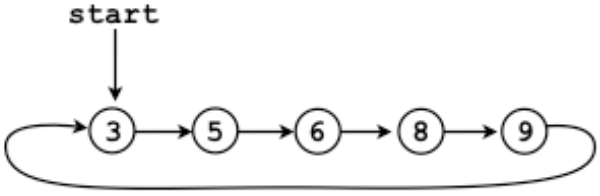
B. _____



C. _____



D. _____



5. **OOP terminology** (7 points). Consider the following code.

```
1     public class Ball
2     {
3         private double rx, ry;

4         public Ball()
5         {  rx = 0.5;  ry = 0.5;  }

6         public void move()
7         {
8            rx = rx + .001;
9            ry = ry + .002;
10        }

11        public void move(double vx, double vy)
12        {
13            rx = rx + vx;
14            ry = ry + vy;
15        }
           ...

16        public static void main(String[] args)
17        {
18            private double vx, vy;
19            Ball b1 = new Ball();
20            Ball b2 = new Ball();
21            b1.move();
              ...
22            b2.move(vx, vy);
              ...
23        }
24    }
```

In the blanks at left, give all line numbers that contain each of the entities described at right.

a) _____ instance variable declaration

b) _____ constructor signature

c) _____ object creation

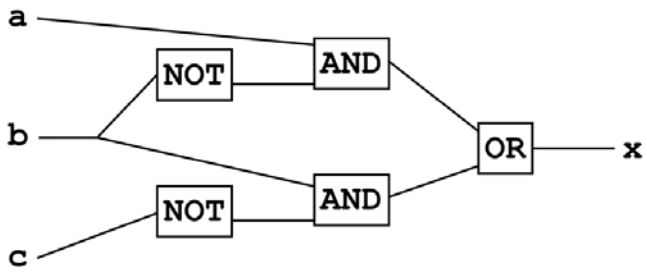
d) _____ overloaded method signature

e) _____ method invocation or call

f) _____ primitive type variable declaration

g) _____ reference type variable declaration

6. **Circuits** (14 points). Consider the following combinational circuit:



a) (8 points) Fill out the truth table.

a	b	c	x

b) (6 points) Circle all boolean expressions in the list below that are logically equivalent to the circuit.

a. `a'b + b'c`

b. `ab' + bc'`

c. `(a + b)c`

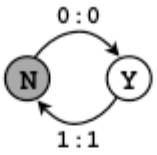
d. `a'bc + ab'c' + abc' + abc`

e. `a'bc' + ab'c' + ab'c + abc'`

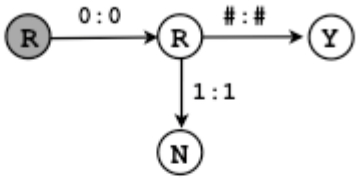
f. `a'bc + ab'c' + ab'c + abc'`

7. **Turing machines** (12 points). Congratulations! You are now a CS graduate student teaching COS126. The exam question asked for a Turing machine that would recognize even (and not odd) binary numbers, assuming that the number is on the input tape, with the head positioned at the most significant bit. Here are the answers from four different students (start state is shaded). In the blank provided, mark each as correct (with a check mark) or wrong (with an X). For those marked wrong, **give a reason** (for example an input for which the machine does not produce the desired result).

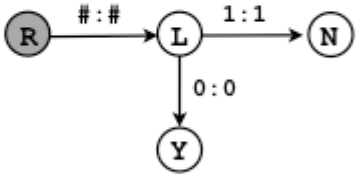
A. _____



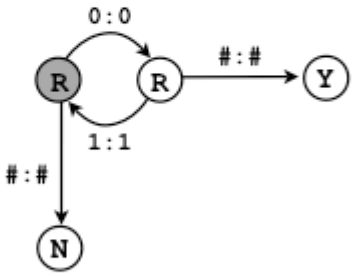
B. _____

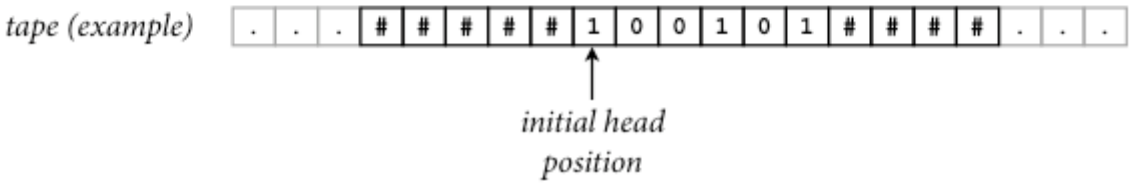


C. _____



D. _____





8. **Theory** (14 points). In the blanks, mark each of the statements below as true (T) or false (F).

A. _____ Any RE with closure (a "*" or a "+") describes infinitely many strings.

B. _____ Any RE without closure describes only finitely many strings.

C. _____ Given unlimited memory, TOY can solve any problem that can be solved by a Turing machine.

D. _____ It is possible to write a program that detects some Java programs that terminate.

E. _____ It is possible to write a program that detects all Java programs that terminate.

F. _____ If P=NP, there is no problem that is NP but is not NP-complete.

G. _____ A Turing machine can decide whether a given string is in the language described by a given regular expression.

H. _____ No Turing machine can decide whether a given DFA halts.

I. _____ The undecidability of the halting problem is a statement about Turing machines: it is *not* applicable to real computers.

J. _____ The Church-Turing thesis is a theory about our universe: it cannot be proven mathematically.

K. _____ If P equals NP, then the Traveling Salesperson Problem can be solved in polynomial time by a deterministic Turing Machine.

L. _____ If P does not equal NP, then there is no case of the Traveling Salesperson Problem for which you can find the optimal tour in polynomial time.

M. _____ Factoring is known to be in NP but has not been proven to be NP-complete, so the discovery of a polynomial-time algorithm for factoring would mean that P equals NP.

N. _____ Factoring is known to be in NP but has not been proven to be NP-complete, so the discovery of a polynomial-time algorithm for TSP would still not give a polynomial time algorithm for factoring.