COS 126 **General Computer Science** **Spring 2006**

Exam 2

This test has 11 questions worth a total of 50 points. You have 120 minutes. 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. **Write out and sign the Honor Code pledge before turning in the test.**

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

Signature

Problem	Score
0	
1	
2	
3	
4	
5	
Sub 1	

Problem	Score
6	
7	
8	
9	
10	
Sub 2	

Total	

Name:

NetID:

P01	TTh 1:30	Donna
P01A	TTh 1:30	Stephen
P01B	TTh 1:30	Kazu
P02	TTh 2:30	William
P02A	TTh 2:30	Chris
P03	TTh 3:30	Jeff
P04	TTh 7:30	Berk
P05	WF 10	Mona
P06	WF 11	Hakan
P07	WF 1:30	Zhuojuan
P07A	WF 1:30	Melissa

0. **Miscellaneous. (2 points)**

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

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

1. **Deconstructing Hello, World. (4 points)**

   Consider the following Java program.

```
public class HelloWorld {
   public static void main(String[] args) {
      System.out.println("Hello, World");
   }
}
```

For each code fragment on the left, find the best matching description on the right.

___ `class`

___ `public`

___ `static`

___ `void`

___ `main`

___ `String[] args`

A. Identifies the method as not returning any value.

B. Identifies the method as shared by the class, rather than one associated with each object instantiated from the class.

C. Identifies the method as available for use by any other program.

D. Identifies the method as being callable at most once within a program.

E. Identifies the method that is automatically invoked when you run `java HelloWorld`.

F. An array of strings containing the command line arguments.

G. A group of related methods and variables.

H. Has no meaning.

2. **Strings, references and debugging. (4 points)**

   What does the following Java program print out? Circle your answer.

```
public class Team {
   private String name = "Elis";

   public Team(String newName) {
      String name = newName;
   }

   public void setName(String newName) {
      name = newName;
   }

   public String toString() {
      return name;
   }

   public static void main(String[] args) {
      Team team = new Team("Crimson");
      System.out.println(team);

      team.setName("Tigers");
      System.out.println(team);

      String x = team.toString();
      x = "Big Red";
      System.out.println(team);

      x.replaceAll("Red", "Green");
      team.setName(x);
      System.out.println(team);
   }
}
```

3. **Data types. (8 points)**

Here is the API for a `Point` data type.

`Point(double x, double y)`	Create a new point in the plane with the given x and y coordinates.
`double distanceTo(Point b)`	Return the Euclidean distance from the invoking point to `b`.
`String toString()`	Return a string representation of the invoking point.

(a) Implement an ADT for `Point`. Recall that the Euclidean distance between two points (x_1, y_1) and (x_2, y_2) is $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. Your answer will be graded for correctness, clarity, and style.

```
public class Point {









   public String toString() {
      return "(" + x + ", " + y + ")";
   }

}
```

(b) Write a client program that reads in one point `a` from the *command line*, reads in an arbitrarily long sequence of points from *standard input* (using `StdIn`), and prints out the point `furthest` that is furthest from `a`. You may *not* assume you know the number of points ahead of time, but you may assume that there is at least one.

```
public class Furthest {
   public static void main(String[] args) {
      // read in Point a from command line
      double x = Double.parseDouble(args[0]);
      double y = Double.parseDouble(args[1]);
      Point a = new Point(x, y);









      System.out.println(furthest + " is furthest from " + a);
   }
}
```

```
% more points.txt
0.0 0.0
3.0 0.0
3.0 4.0

% java Furthest 5.0 0.0 < points.txt
(0.0, 0.0) is furthest from (5.0, 0.0)
```

4. **Encapsulation. (4 points)**

The Springfield nuclear power plant is interviewing you for a job to monitor their nuclear reactor. You are responsible for modeling (in software) a temperature gauge that records the temperature of the reactor core for each second of a given day (86,400 readings). The readings are measured in Kelvin, so the only meaningful temperature values are *nonnegative*. The following code breaks encapsulation: a sloppy or malicious client can sneak in a negative temperature reading.

```
public class Gauge {
   int N = 86400;
   double[] temp;

   // initialize the gauge to all zeros
   public Gauge() { temp = new double[N]; }

   // set the temperature readings
   public void setTemp(double[] t) {
      temp = t;
      if (!isConsistent()) Reactor.shutDown();
   }

   // is the data type in a consistent state?
   private boolean isConsistent() {
      if (temp == null) return false;
      if (temp.length != 86400) return false;
      for (int i = 0; i < N; i++)
         if (temp[i] < 0) return false;
      return true;
   }

   // return the array of temperature readings
   public double[] getTemp() { return temp; }

   // return the minimum temperature for the day
   public double min() {
      double min = Double.POSITIVE_INFINITY;
      for (int i = 0; i < N; i++)
         if (temp[i] < min)
            min = temp[i];
      return min;
   }
}
```

Give 3 *different* ways that a *client* can get one of the temperature readings to be negative without shutting down the reactor. To do so, provide Java code fragments that would result in a negative value ending up in `temp`. (A bonus point if you identify all 4 ways.) You may not modify `Gauge.java`.

Assume that `gauge` is a reference to a variable of type `Gauge` and `t` is an array of 86,400 zeros.

```
Gauge gauge = new Gauge();
double[] t = new double[86400];
```

1.

2.

3.

*4.

5. **Linked structures. (4 points)**

Consider the following data type methods which use the helper `Node` data type to find a given integer key within a linked structure (assuming that it is present). Assume that `start` is an instance variable (private field) that refers to a fixed `Node` in the linked structure.

```
private static class Node {
   private int key;
   private String name;
   private Node left, right;
}
```

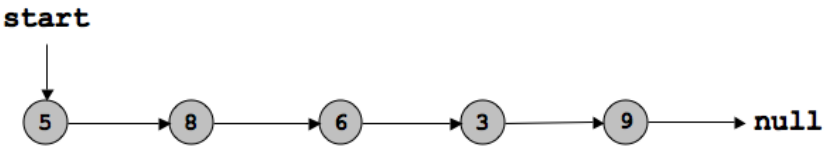
```
public String findA(int key) {
   Node x = start;
   while (x != null) {
      if (x.key == key) return x.name;
      x = x.right;
   }
   return null;
}
```

```
public String findB(int key) {
   Node x = start;
   while (x != null) {
      if (x.key == key) return x.name;
      if (x.key >  key) x = x.left;
      else              x = x.right;
   }
   return null;
}
```

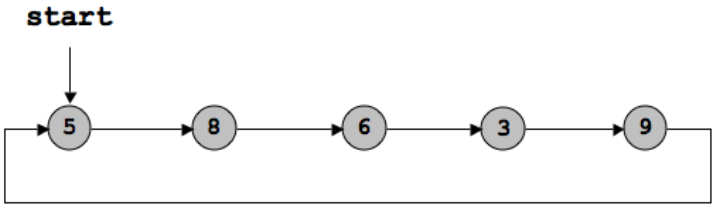
```
public String findC(int key) {
   if (start == null) return null;
   Node x = start;
   do {
      if (x.key == key) return x.name;
      x = x.right;
   } while (x != start);
   return null;
}
```

Match up each linked structure with the corresponding function for finding a given element on the facing page. (Assume that the element is contained in the structure.) The nodes in the first two structures are in arbitrary order. The third linked structure contains the elements in ascending order. The fourth linked structure is a binary search tree: each element is bigger than its left child and smaller than its right child.

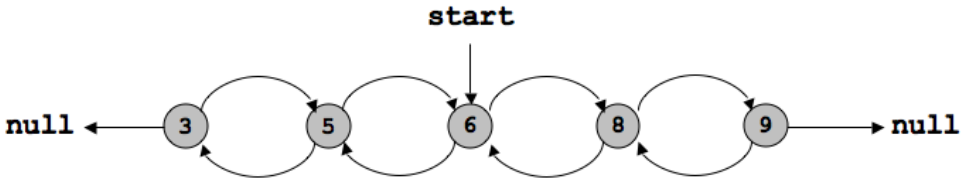
_____



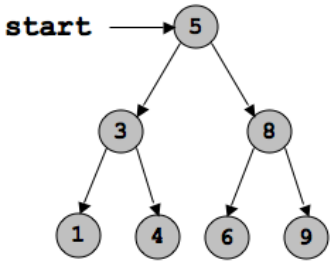
_____



_____

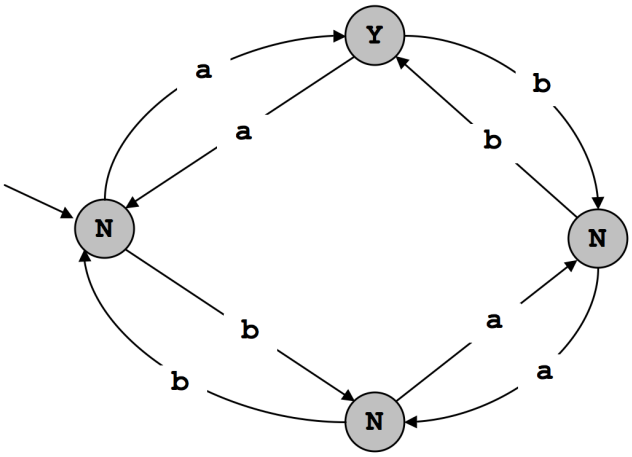


_____



6. **DFAs and regular expressions. (5 points)**

(a) Describe the set of strings the following DFA recognizes.



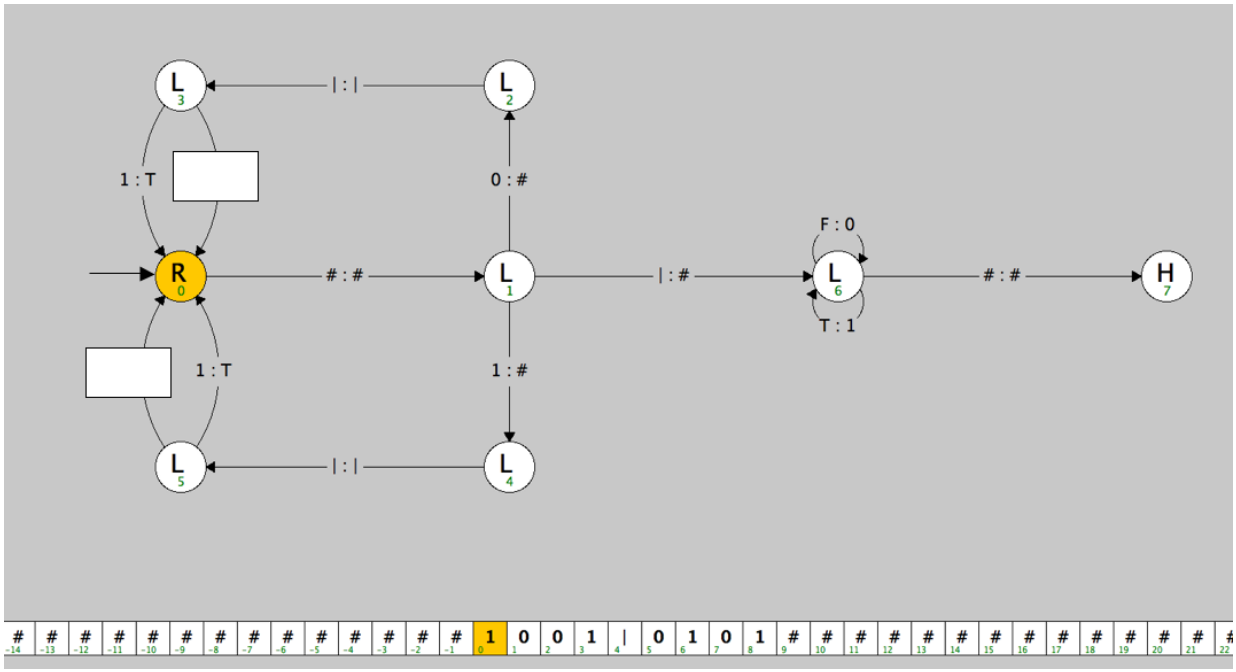
(b) Does there exist a regular expression that matches exactly the same set of strings? Briefly justify your answer.

(c) Give a Java regular expression to match all DNA sequences that start with `GCG`, end with `CTG`, and in between consists of one or more repeats (in any order) of the triplets `CGG` and `AGG`.

yes	no
`GCGCGGCTG`	`TTT`
`GCGCGGAGGCTG`	`GCGCTG`
`GCGCGGAGGCGGCTG`	`GCGCGGCGGTTTCTG`

7. **Turing machines. (3 points)**

(a) The Turing machine below is designed to take two bit-strings (of the same length N), separated by the `|` symbol, and leaves the bitwise OR of the two bit-strings on the tape. For example, when executed with the tape below, the Turing machine should leave `1101` on the tape. Complete the Turing machine (by specifying the label on the two unlabeled transitions) so that it works as advertised.



(b) How many steps (tape head moves) does the Turing machine make to compute the bitwise OR of two bit-strings of length N? Circle the best answer.

$\log N$ $\quad$ N $\quad$ $N \log N$ $\quad$ N^2 $\quad$ 2^N

8. **Data structures. (4 points)**

For each problem on the left, put the letter of the *best* matching data structure on the right.

___ Maintain a database of information about registered vehicles, indexed by VINs (vehicle identification numbers).

___ Simulate the behavior of cars at a traffic light.

___ Evaluate arithmetic expressions.

___ Represent the interconnections between neurons in the human brain.

A. Stack

B. Queue

C. Symbol table

D. Graph

E. Adjacency list

9. **Intractability, Universality, and Computabaility. (6 points)**

Write T for true or F for false next to each of the following statements, according to their veracity.

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

___ The Church-Turing thesis is a theory about our universe: it cannot be proven mathematically, but it can be falsified.

___ The Turing machine is a universal model of computation: with a Turing machine we can solve any decision problem that can be solved with a DFA or with a Pentium M running Linux 2.6.16.

___ P is the class of yes-no problems for which a polynomial-time Java program could, in principle, be written.

___ NP is the class of yes-no problems for which *no* polynomial time Java program could ever be written.

___ NP is the class of yes-no problems for which, in principle, you could write a Java program to check a given proposed solution in polynomial-time.

___ If you discover a polynomial-time algorithm for any problem in NP, then all problems in NP are solvable in polynomial-time.

___ Most Princeton computer science faculty believe that P = NP.

___ Since FACTOR is a problem in NP, any instance of FACTOR can be restated as an instance of 3-SAT (3-satisfiability).

10. **Circuits. (6 points)**

Consider the function $f(x_2, x_1, x_0)$ that is true if and only if $x_2x_1x_0$ is a 3-bit positive number strictly greater than 4. Here x_2 is the most significant (leftmost) bit, x_1 is the middle bit and x_0 is the least significant (rightmost) bit.

(a) Fill in the truth table for the function f.

x_2	x_1	x_0	f
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

(b) Using *sum-of-products*, give a Boolean formula for the function f.

(c) Draw a circuit that computes the function f that uses the *minimum total number* of AND, OR, and NOT gates.