

**Written Exam 2**

This test has 9 questions, weighted as indicated. The exam is closed book, except that you are allowed to use a one-page (doublesided) cheatsheet handwritten by you. 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
1	/8
2	/6
3	/10
4	/7
Sub 1	

Problem	Score
5	/10
6	/3
7	/5
8	/10
Sub 2	

Total	
-------	--

**Name:**

**NetID:**

**Lecture:** L01 10:00  
L02 11:00

**Precept:** P01 12:30 Christopher Moretti  
(circle your P01A 12:30 Donna Gabai  
lecture P01B 12:30 Jude Nelson  
AND P02 1:30 Chris Miller  
precept) P02A 1:30 Donna Gabai  
P02B 1:30 Maia Ginsburg  
P02C 1:30 Thiago Pereira  
P03 2:30 Chris Miller  
P03A 2:30 Christopher Moretti  
P04 3:30 Maia Ginsburg  
P04A 3:30 Dom Kao  
P05 7:30 Adi Dror  
P06 10:00 Deep Ghosh  
P07 1:30 Ian Davey  
P07A 1:30 Jude Nelson  
P07B 1:30 Dmitry Drutskoy  
P08 12:30 Deep Ghosh  
P08A 12:30 Mark Browning

0. **Cover Page.** Write your name and netid, select your lecture and precept sections, and write out the honor code on the cover (now!) and sign it when you complete the test.
1. **Regular Expressions.** What regular expression will match U.S. street addresses? Assume that street numbers never start with 0, that street names can have multiple capitalized words and may have just one character, that all end with “Street” or “Road” or “Avenue”, and that one or more whitespace characters will separate the parts. Use character range notation: for example, [b-n] will match any lower-case letter alphabetically between b and n, and [2-4] will match 2, 3, or 4. Remember that \s matches any whitespace character.

match	examples	don't match
25 Spring Garden Street		0034 Main Street
6666 N Street		34 MAIN Street
2 Pleasant Avenue		2 loVeLy Road
444 Pheasant Hop Road		44 Happy Hollow Rd.

Circle the letter for each RE below that matches the street address model above. (Circle f if none match.)

For each RE that you do NOT circle, underline one part of the RE which does not match the desired model.

- a. `[0-9]*\s[A-Z][a-z]\s(Street | Road | Avenue)`
- b. `[1-9][0-9]+\s+([A-Z][a-z]+\s+)(Street | Road | Avenue)`
- c. `[1-9][0-9]*\s+([A-Z][a-z]*\s+)(Street | Road | Avenue)`
- d. `[1-9][0-9]*\s+([A-Z][a-z]+\s*)+(Street | Road | Avenue)`
- e. `[1-9]+\s+([A-Z][a-z]*\s+)(Street | Road | Avenue)`
- f. none of the above.

2. **List.** Consider the following buggy linked list code:

```
1: public class List {
2:
3:     private Node start; // pointer to the beginning of the list
4:
5:     private class Node {
6:         int data;
7:         Node next;
8:         Node (int x, Node y) {data=x; next=y;}
9:     }
10:
11:     // create a new linked list of 7 nodes.
12:     public List() {
13:         Node node7 = new Node(7, null);
14:         Node node6 = new Node(6, node7);
15:         Node node5 = new Node(5, node6);
16:         Node node4 = new Node(4, node5);
17:         Node node3 = new Node(3, node4);
18:         Node node2 = new Node(2, node3);
19:         start = new Node(1, node2);
20:     }
21:
22:     // show selected items from the linked list.
23:     public void show() {
24:         Node p = start;
25:         while (p != null) {
26:             p.next = p.next.next;
27:             System.out.print(p.data);
28:             p = p.next;
29:         }
30:     }
31:
32:     public static void main(String[] args) {
33:         List list = new List();
34:         list.show();
35:         System.out.println();
36:         System.out.println("Done!");
37:     }
38: }
```

What is the output when java List is run?

3. **ADT.** Consider the following two files: `AudioPlayer.java` and `TestAudioPlayer.java`. The first is a class that maintains and plays a list of songs. The second is a program that uses these objects.

```
1: public class AudioPlayer {
2:     // table of key->value mappings. a song title -> its popularity count.
3:     private ST<String, Integer> st;
4:
5:     // create empty symbol table
6:     public AudioPlayer() { st = new ST<String, Integer>(); }
7:
8:     // add new song to symbol table or add to count if already there
9:     public void addSong(String title) {
10:         if (!st.contains(title)) st.put(title, 1);
11:         else {
12:             int count = st.get(title) + 1;
13:             st.put(title, count);
14:         }
15:     }
16:
17:     // play a song in symbol table and print its title and popularity count
18:     public void play(String title) {
19:         if (!st.contains(title)) return;
20:         System.out.println("Song: " + title);
21:         System.out.println("Count: " + st.get(title));
22:         addSong(title);
23:         StdAudio.play(title + ".wav"); // assume we can have blanks in filenames
24:     }
25:
26:     // play and print all songs
27:     public void playAll() {
28:         for (String title : st)
29:             play(title);
30:     }
31: }
```

```
1: public class TestAudioPlayer {
2:     public static void main(String[] args) {
3:         AudioPlayer popularlist = new AudioPlayer();
4:         AudioPlayer backgroundlist = new AudioPlayer();
5:         popularlist.addSong("Hard Work");
6:         popularlist.addSong("Too Close");
7:         backgroundlist.addSong("Wild One");
8:         backgroundlist.addSong("Too Close");
9:         backgroundlist.playAll();
10:
11:     }
12: }
```

(A.) Write the printed output when `java TestAudioPlayer` is run. (Assume ST iteration order is alphabetical.)

(B.) Suppose `main` in `TestAudioPlayer` had this code on line 10:

```
int i = backgroundlist.st.get("Hard Work");
```

Why would this cause a compile error?

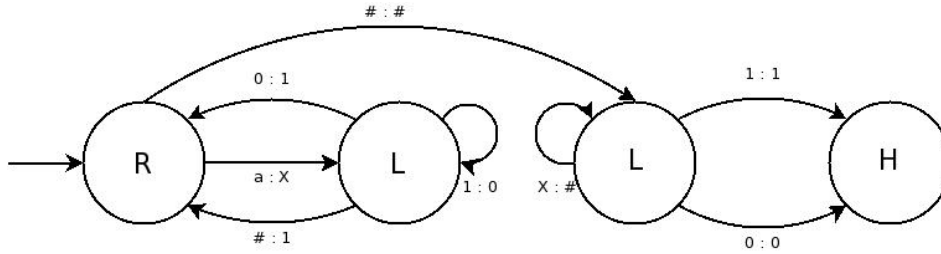
(C.) Suppose `st` were declared as static. What would the answer to (A.) be?

(D.) The class `AudioPlayer` is immutable. TRUE or FALSE (circle your answer)

(E.) Each of the following occurs one or more times in the code above. Give the class name and line number(s) for ONE example of each occurrence (you may abbreviate the class names as AP and TAP).

_____	_____	client program	_____	_____	instance variable declaration
_____	_____	constructor signature	_____	_____	object creation
_____	_____	primitive type declaration	_____	_____	reference type declaration

4. **Turing Machine.** Use this Turing Machine to answer the following questions. You may assume that the tape is initially filled entirely with # symbols except for the a's. Remember that the very first thing the Turing Machine does is read the symbol under the head.



#	#	#	#	<u>a</u>	a	a	a	a	#	#	#	#	#
#	#	#	#	X	a	a	a	a	#	#	#	#	#

--	--	--	--	--	--	--	--	--	--	--	--	--	--

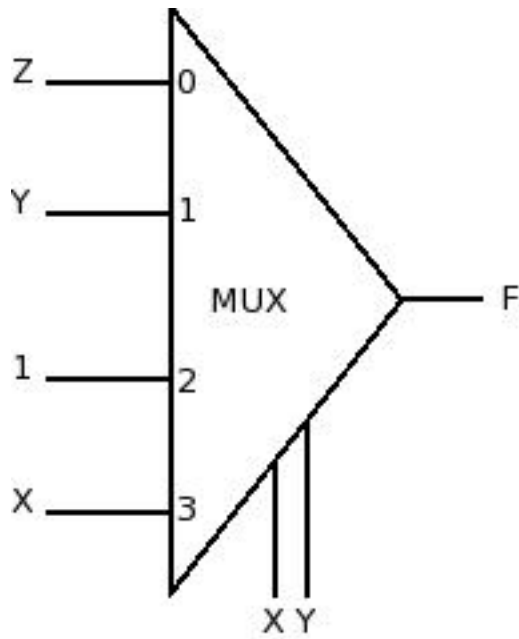
(A.) For the starting tape shown on the first line of the table, with the head (at the shaded box and underlined) starting on the left-most a, what is the ending tape and the location of the head? Use the boxes provided and indicate the location of the head.

We have completed the first step for you as an example, with the result shown on the second line.

(B.) What function is this TM performing?

5. Circuits.

(A.) Using the table on the right, write the truth table for this multiplexer circuit. In the circuit Y is the low-order, or least significant bit, of the select input.



X	Y	Z	F

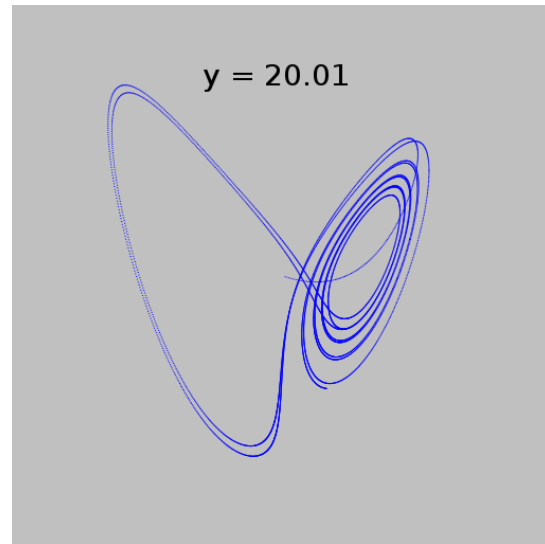
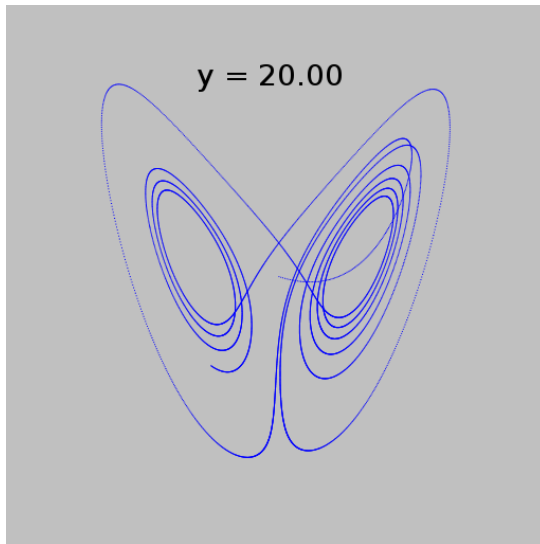
(B.) Write the Boolean algebraic expression for the circuit in simplest form. For partial credit, show the sum of products form.

## 6. Scientific Computation.

The Lorenz Attractor is described by the following system of differential equations:

$$\begin{aligned}\frac{dx}{dt} &= -10(x + y) \\ \frac{dy}{dt} &= -xz + 28x - y \\ \frac{dz}{dt} &= xy - \frac{8}{3}z\end{aligned}$$

The two pictures below were generated by plotting the Lorenz Attractor in Java starting at  $x = 0.0$ ,  $z = 25.0$ , and  $y$  as specified in the images. The same number of iterations were plotted for each image.



Which of the following best explains why the plots are so much different? Circle the best answer.

- (i) Java is not using enough sample points and/or the samples are not taken uniformly.
- (ii) Java uses the IEEE floating-point standard, so the Lorenz Attractor equations are unstable because of catastrophic cancellation.
- (iii) The Lorenz Attractor equations are ill-conditioned, so minor changes to their initial conditions result in significant differences in behavior, regardless of the means used to calculate them.



7. **BST.** Suppose we have int values between 1 and 1000 in a Binary Search Tree and we search for 527. Mark as Y any of the following that could be the sequence of keys examined in a search for 527. Mark as N any sequence that could not result.

- \_\_\_\_(A.) 527
- \_\_\_\_(B.) 1 500 600 700 527
- \_\_\_\_(C.) 605 256 490 300 527
- \_\_\_\_(D.) 10 860 523 602 525 527
- \_\_\_\_(E.) 10 860 523 602 599 610 527

8. **Universality, Computability, and Intractability.** For each statement, identify if it is TRUE (T), FALSE (F), or UNKNOWN (U) in the space provided to the left of the statement.

- \_\_\_\_(A.) If some Turing Machine (TM) matches a certain set of strings, then there is a corresponding Deterministic Finite-State Automaton (DFA) that matches the same set.
- \_\_\_\_(B.) P is the set of problems solvable in Polynomial time on a non-deterministic TM.
- \_\_\_\_(C.) NP is the set of problems solvable in Non-Polynomial time on a non-deterministic TM.
- \_\_\_\_(D.) P equals NP.
- \_\_\_\_(E.) Most computer scientists believe that the Halting Problem will eventually be proved decidable.
- \_\_\_\_(F.) If a polynomial time algorithm for SAT is found, then the Halting Problem is decidable.
- \_\_\_\_(G.) FACTOR (integer factorization) is NP-complete.
- \_\_\_\_(H.) Any problem solvable in polynomial time on your laptop can also be solved in polynomial time on a deterministic TM.
- \_\_\_\_(I.) If the TSP problem is proved to be in P, then so is every other problem in NP.
- \_\_\_\_(J.) If the TSP problem is proved NOT to be in P, then neither is any other problem in NP.