CPSC 219 Test 3
Tuesday, November 12th, 2002

Name: ____________________ Student ID: ____________________
Signature: ____________________

Lab Section: ______

Circle which **Lecture Section** you are in: 101 (Patrice) 103 (Donald)

- You have 40 minutes to write the 5 questions on this examination. A total of 36 marks are available.
- **Justify all of your answers.**
- No notes or electronic equipment are allowed. A list of methods from the Java API is provided on the last page.
- Keep your answers short. If you run out of space for a question, you have written too much.
- The number in square brackets to the left of the question number indicates the number of marks allocated for that question. Use these to help you determine how much time you should spend on each question.
- Use the back of the pages for your rough work.
- **Good luck!**

Question	Marks
1	
2	
3	
4	
5	
Total	

– Candidates must not destroy or mutilate any examination material; must hand in all examination papers; and must not take any examination material from the examination room without permission of the invigilator.

[5 marks] 1. At any given point in time, a thread can be in one of several different states. List these states, and describe each of them *briefly*.

[4 marks] 2. Consider the following method header:

```
public synchronized void appendToList(Object foo);
```

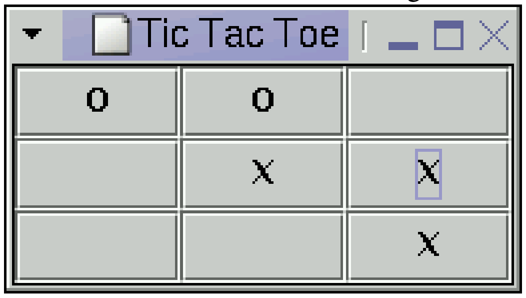
What does the keyword *synchronized* mean, and under what circumstances is it used?

[3 marks] 3. In a computer network, what is the difference between latency and throughput?

[12 marks] 4. You have discovered that there are a number of dictionary servers available on the Internet. To use these servers a TCP/IP connection is made to the server. The client then sends a single newline-terminated string containing the word to lookup. The server responds with the definition and then closes the connection. You may assume that if the word isn't in the dictionary, then some data is still returned (like the string "Not found"). You are to write a class named LexisLookup to simplify access to these servers. Add your code to the appropriate places in the skeleton below. Be sure to add comments were appropriate.

```
import java.io.*;
import java.net.*;
public class LexisLookup {



    public LexisLookup(String host, int port) throws IOException {






    }
    public  void lookup(String word, PrintWriter out) {
      // Open the connection, write word to the server, and copy
      // the response to out.



    }
}
```

[12 marks] 5. During the long week-end, a CPSC 219 student got bored and decided to write a tic-tac-toe game in Java. Recall that tic-tac-toe is played on a 3 by 3 grid. Two players take turns placing a mark in one of the squares (an X for one player, an O for the other), and the player who gets three marks in one horizontal, vertical or diagonal line wins the game.



In the program, the student used a 3 by 3 grid of `JButton` objects, on which the user would click to place their mark. The program then alternated between “X” and “O” from one mouse click to the next. Of course if a player clicks on a cell that already has an “X” or an “O” in it, then nothing happens.

Here is part of the `Board` class that the student wrote (note that the `main` method does nothing except create and display a `Board` object).

```
class Board extends JFrame {
    final static String empty = "   ";
    JButton squares[];
    String  turn;

    public void checkWin() {
        // There was some code here.
    }

    public Board(String s) {
        super(s);
        turn = "X";
        squares = new JButton[9];
        Container mycpane = getContentPane();

        mycpane.setLayout(new GridLayout(3,3));
        for (int i = 0; i < 9; i++)
        {
            squares[i] = new JButton(empty);
            // Write more code here.
        }
    }
}
```

Complete the `Board` constructor by writing the code that would go in place of the comment `// Write more code here.` Do **not** write the `checkWin` method. We have included some useful information about Swing on the last page on the test. Do not write too much! Our solution to this question only contains 11 lines of code, plus a few lines with nothing except a curly bracket on them.

Java API Classes and Method

The Socket class

Socket()	Create an unconnected socket.
Socket(String, int)	Create a socket connecting to the given host on the given port.
void close()	Closes a socket.
InputStream getInputStream()	Returns the input stream for this socket.
OutputStream getOutputStream()	Returns the output stream for this socket.
String toString()	Returns a string representation of the socket.

The ServerSocket class

ServerSocket()	Create an unbound server socket.
ServerSocket(int)	Create a server socket bound to that port.
Socket accept()	Listen for (and accepts) connections made to this socket.
void close()	Closes a server socket.
String toString()	Returns a string representation of the socket.

The JButton class

JButton(String)	Create a JButton instance, initialized with text.
String getText()	Get the text displayed by the button.
void setText(String)	Sets the text displayed by the button.
void setBackground(Color)	Set the background color of the button.
Color getBackground(Color)	Returns the background color of the button.
String setActionCommand(void)	Set the name of the action performed by the button.
void addActionListener(ActionListener)	Add object to listen for action events from the button.

The ActionEvent class

Object getSource()	Returns the object that fired the event.
String getActionCommand()	Returns the string associated with this action.

The ActionListener interface

void actionPerformed(ActionEvent)	Called when the user clicks on a button/menu item.