First Name: _____ **Last Name**: _____

McGill ID: _____ **Section**: _____

Faculty of Science
COMP-202A - Introduction to Computing I (Fall 2008)
Midterm Examination

Tuesday, November 4, 2008
18:30–20:30

Examiners: Mathieu Petitpas [Section 1]
Prof. Paul Kry [Section 2]
Gregory B. Prokopski [Section 3]

Instructions:

- **DO NOT TURN THIS PAGE UNTIL INSTRUCTED**
- This is a **closed book** midterm; notes, slides, textbooks, and other forms of documentation are **not** allowed.
- **Non-programmable calculators** are allowed (though you should not need one).
- **Computers, PDAs, cellphones, and other electronic devices** are **not** allowed.
- Answer all questions on the midterm paper; if you need additional space, use the last page or the booklets supplied and clearly indicate where each question is continued. **In order to receive full marks, you must show all work.**
- This midterm has examination **16** pages including this cover page.

1	2	3	4	Subtotal
/4	/6	/6	/4	/20

5	6	Subtotal
/15	/15	/30

7	8	Subtotal
/30	/20	/50

Total
/100

Section 1 - Short Questions

[4] 1. List 4 Java **reserved words**.

[6] 2. What will be the value stored in variable `result` after each of the following code fragments is executed? Describe with a few words and a value what happens at all intermediate steps. Here is an example:

```
int a = 2, b = 1, c = 5;
double result = a * (b + c);
```

Answer:

- Addition produces produces `6`
- Multiplication produces `12`
- Conversion on assignment to `result` gives `12.0`

(a)
```
int a = 23;
int b = 46;
double c = 3.0;
double d = 4.5;
double result = (int)d * c + a / b;
```

(b)
```
int a = 4;
int b = 1;
double c = a * 2;
double d = c / a;
String result = a + c + " == " + b + d;
```

[6] 3. For each of the following conditions, write a **boolean expression** that evaluates to `true` if and only if the condition is true. You are not allowed to use any methods from the Java Platform API (that is, the Java standard library) to answer this question.

(a) The number stored in variable `y` (of type `int`) is the number of a leap year; a leap year is a year whose number is divisible by 4, but not by 100, unless it is divisible by 400.

(b) **Exactly two** of the values stored in variables `a`, `b`, and `c` (all of type `int`) are equal.

(c) The value stored in variable `c` (of type `char`) is an upper-case **consonant**. In the following set (which contains all the letters of the English alphabet), the letters in **bold** are consonants, while the letters in *italics* are not: {*A*, **B**, **C**, **D**, *E*, **F**, **G**, **H**, *I*, **J**, **K**, **L**, **M**, **N**, *O*, **P**, **Q**, **R**, **S**, **T**, *U*, **V**, **W**, **X**, **Y**, **Z**}.

[4] 4. The following program prints a shape to the screen using the character *. What shape is it and what are its dimensions?

```
public class Shape {
  public static void main(String[] args) {
    final int SIZE = 6;

    for (int i = 0; i <= SIZE; i++) {
      for (int j = 0; j <= SIZE; j++) {
        if ((j == i) || (j == SIZE-i) || (j == SIZE/2)) {
          System.out.print("*");
        } else {
          System.out.print(" ");
        }
      }
      System.out.println();
    }
  }
}
```

Section 2 - Long Questions

[15] 5. Consider the following program:

```
    public class TraceMe {
      public static void main(String[] args) {
        String text, chunk;
        int i;
        char c;

 1      text = "C u";
 2      chunk = "";
 3      i = 0;
 4      while (i < text.length()) {
 5        c = text.charAt(i);
 6        if ((c == ' ') || (c == '\n')) {
 7          System.out.println("Chunk: '" + chunk + "'");
 8          chunk = "";
          } else {
 9          chunk += c;
          }
10        i++;
        }
11      System.out.println("Last chunk: '" + chunk + "'");
      }
    }
```

Trace this program by writing down the list of the statements which will be executed, in the order in which they will be executed in. Each statement must be included on your list exactly as many times as it is executed. Use the numbers to the left of each statement to indicate which statement is executed.

To do this correctly, write your answer using a 2-column format. For each row, the first column contains the statement number of the statement being executed, and the second column contains the following information, according to the type of the statement:

- If the statement is an assignment statement: the variable whose value is being changed, and its new value
- If the statement is a control flow construct (`if` or `while`): the value to which the boolean expression controlling it evaluates
- If the statement is a call to the `println()` method: what is displayed on the screen

As an example, here are the first five lines of the answer:

```
1 text ← "C u"
2 chunk ← ""
3 i ← 0
4 true
5 c ← 'C'
```

Continue the trace until the program terminates:

[15] 6. Consider the following program, saved in a file called `EqualStringChecker.java`:

```
1  import java.util.Scanner;
2
3  public class EqualStringChecker {
4    Public static void main(String[] args) {
5      Scanner keyboard = null;
6      boolean equal;
7      String s1, s2;
8      int s1Length;
9
10     System.out.print("Enter the first String: ");
11     s1 = keyboard.nextLine();
12     System.out.print("Enter the second String: ");
13     s2 = keyboard.nextLine();
14
15     s1Length = s1.length();
16     if (s1Length != s2.length()) {
17       equal = false;
18     } else {
19       equal = true;
20       i = 0;
21       while (i < s1Length || equal) {
22         if (s1.charAt(i) != s2.charAt(i)) {
23           equal = false;
24         } else {
25           i = i + 1;
26         }
27       }
28
29     if (equal) {
30       System.out.println("The Strings are equal.");
31     } else {
32       System.out.println("The Strings are not equal.");
33     }
34   }
35 }
```

The above program is designed to check whether two `String`s entered by the user are equal (that is, whether they contain the same characters in the same order). However, there are **5** errors in the above program. Find all the errors and list them. For each error you list, you must include the number of the line where the error occurs, the type of error (syntactic or semantic) and a description of the error. Do not list more than 5 errors, as you will be penalized for every "error" in excess of 5 that you list.

Note that the line numbers to the left of the above program are included solely to help you make it easier for you to list the line numbers where errors occur; they are not part of the actual program.

List the errors you find in the program here:

Total marks for Section 2: **30**

Section 3 - Programming Questions

7. This question has two parts. In the first part, you will write a class; an object which belongs to this class represents a 2 by 2 rectangular table of elements called a matrix. In the second part, you will write a short program which creates and uses objects which belong to the class you wrote in the first part.

[20] **Part 1**:

Write a class called `Matrix`; each object which belongs to this class represents a 2×2 matrix whose elements are real numbers. You should use the labels a, b, c, and d, to represent the 4 entries of the matrix, where

- a is the value in row 1, column 1
- b is the value in row 1, column 2
- c is the value in row 2, column 1
- d is the value in row 2, column 2

Thus, a 2×2 matrix M with values a, b, c, and d looks like this:

$$M = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

Your `Matrix` class should provide the following `public` methods:

- A constructor, which takes as parameters four `doubles`; these `doubles` represent the values for a, b, c, and d of the `Matrix` to be created, in this order.
- A method called `hasInverse()`, which takes no parameters and returns a value of type `boolean`. This value is `true` if the `Matrix` object has an inverse, `false` otherwise; the general 2×2 matrix M defined above has an inverse if and only if $ad - bc$ is not equal to 0.
- A method called `computeInverse()`, which takes no parameters and returns a `Matrix`. This method should compute the inverse of the `Matrix` object on which it is invoked, and return a new `Matrix` object representing this inverse matrix. If the `Matrix` object has no inverse, this method should return `null`. In all cases, the state of the `Matrix` object on which this method is invoked should not change as a result of calling this method. The inverse of the general 2×2 matrix M defined above is denoted M^{-1} and has the following form:

$$M^{-1} = \begin{pmatrix} d/(ad-bc) & -b/(ad-bc) \\ -c/(ad-bc) & a/(ad-bc) \end{pmatrix}$$

- A method called `toString()`, which takes no parameters and returns a `String` object. This `String` object is a textual representation of the `Matrix` object. This text representation should have the following format: `[a, b \ c, d]`, where a, b, c, and d are replaced by the actual values of the `Matrix` object.

WRITE YOUR `Matrix` CLASS IN THE SPACE BELOW:

[10] **Part 2**:

Write a class called `ComputeInverse`, which has only a `main()` method that does the following:

- Ask the user to enter values `a`, `b`, `c`, and `d` to be stored in a 2×2 matrix, and read these values from the keyboard.
- Display the matrix in the following format: `[a, b \ c, d]`
- Compute the inverse of the matrix entered by the user if it exists, and display this inverse matrix in the same format as the original matrix. If the matrix created using the values entered by the user has no inverse, your program should display: `M has no inverse`

Sample session:

```
Enter the values of the matrix M:
    Row 1, column 1 (a):  -1
    Row 1, column 2 (b):  3
    Row 2, column 1 (c):  2
    Row 2, column 2 (d):  -4

M == [-1.0, 3.0 \ 2.0, -4.0]
M^-1 == [2.0, 1.5 \ 1.0, 0.5]
```

You should use the `Matrix` class written in the first part of this question as much as possible to write your program for this question. You may write `print()` instead of `System.out.print()` and `println()` instead of `System.out.println()` in order to save time.

YOUR `ComputeInverse` CLASS CONTINUED:

[20] 8. Write a method called `add()` that takes two `String`s as parameters, each representing a non negative integer. Your `add()` method should construct a new `String` representing the sum. You may only use the `String` methods `length()` and `charAt()` in writing your method; you cannot call any other method defined in the `String` class or any other class. You may assume that neither of the `String` parameters represents the empty `String`, that is, `""`, and that all characters in the two parameters will only consist of digits. Note that your method should work for `String`s representing numbers which consist of an arbitray number of digits. Here follows a program with some test cases you may want to consider when writing your method.

```
public class AddTwoStrings {

  public static String add(String s1, String s2) {
    // Your code here
  }

  public static void main( String args[] ) {
    System.out.println( "7 + 5 = " + add("7", "5") );
    System.out.println( "18 + 6 = " + add("18", "6") );
    System.out.println( "6 + 18 = " + add("6", "18") );
    System.out.println( "999 + 1 = " + add("999", "1") );

    // A silly example
    String coffee = "12648430";  // 0xC0FFEE
    String mersennePrime61 =
        "2305843009213693951"; // Pervushin 1883
    String usNationalDebt =
        "10474000336886";
    System.out.print(
        "coffee + mersennePrime61 + usNationalDebt = ");
    System.out.println(
          add( add( coffee, mersennePrime61 ), usNationalDebt) );
  }
}
```

This program generates the following output:

```
7 + 5 = 12
18 + 6 = 24
6 + 18 = 24
999 + 1 = 1000
coffee + mersennePrime61 + usNationalDebt = 2305853483226679267
```

Write your answer in the space provided on the following page.

Hint: Think of how you add two numbers by hand.

WRITE YOUR `add()` METHOD IN THE SPACE BELOW:

```
public static String add(String s1, String s2) {
  // your code here
```

Total marks for Section 3: **50**

Total marks: **100**

USE THIS PAGE IF YOU NEED ADDITIONAL SPACE. CLEARLY INDICATE WHICH QUESTION(S) YOU ARE ANSWERING HERE.