EXAMINATION INSTRUCTIONS

This examination has 6 pages, 5 numbered pages and this cover page.
Check that you have a complete paper.
Each candidate should be prepared to produce, upon request, his or her SUNY/UB card.
This examination has 5 questions. Answer all questions.
You have 60 minutes to complete this examination. Use your time accordingly.

READ AND OBSERVE THE FOLLOWING RULES:

- Names are pre-printed on the exam booklets. Ensure that you have YOUR exam.
- Sign, using your usual signature, in the space provided on the back cover.
- All of your writing must be handed in. This booklet must not be torn or mutilated in any way, and must not be taken from the examination room.
- Show all of your work in arriving at an answer, unless instructed otherwise. Partial credit will be awarded as appropriate.
- Candidates are not permitted to ask questions of the invigilators, except in cases of supposed errors or ambiguities in examination questions.
- CAUTION – Candidates guilty of any of the following, or similar, dishonest practices shall be immediately dismissed from the examination and shall be liable to disciplinary action.
  - Making use of any books, papers or memoranda, calculators or computers, audio or visual cassette players, or other memory aid devices, other than those explicitly authorised by the examiners.
  - Speaking or communicating with other candidates.
  - Purposely exposing written papers to the view of other candidates. The plea of accident or forgetfulness shall not be received.

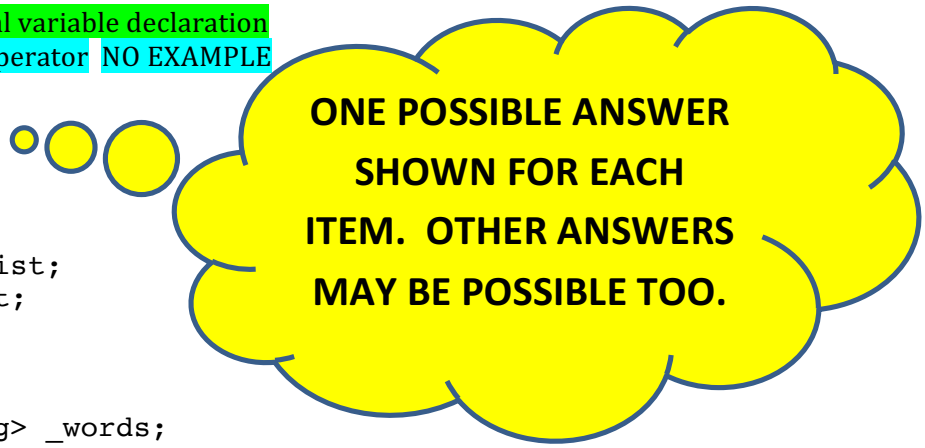
---------- DO NOT WRITE BELOW THIS LINE! ----------

Q1	Q2	Q3	Q4	Q5	TOTAL	%
/10	/10	/10	/10	/10	/50	**/100**

Question 1 [10 points, 2 points each]

The code given below is correct: it compiles without errors. Clearly circle and identify by number one (and only one) example of each of the following. Be sure to identify each item precisely (circle no more and no less than necessary). If you believe no example exists in the given code, write "*no example*" next to that item.

1. int literal
2. a (non-parameter) local variable declaration
3. String concatenation operator NO EXAMPLE
4. relational operator
5. boolean operator




```
package code;

import java.util.ArrayList;
import java.util.HashSet;

public class Model {

  private HashSet<String> _words;
  private ArrayList<String> _longWords;

  public Model(HashSet<String> words, ArrayList<String> longWords) {
    _words = words;
    _longWords = longWords;
  }

  public Model(String filename) {
    _words = new HashSet<String>();
    _longWords = new ArrayList<String>();
    readDictionaryFromFile(filename);
  }

  public void readDictionaryFromFile(String filename) {
    CSE115FileReader scan = new CSE115FileReader(filename);
    while (scan.hasNext()) {
      String word = scan.next();
      if (word.length() == 7 && ! _longWords.contains(word)) {
        _longWords.add(word);
      }
      if (word.length() < 7) {
        _words.add(word);
      }
    }
  }

  // THE REST OF THE CLASS DEFINITION IS OMITTED

}
```

Question 2 [10 points total]

[GRADING: 10 total marks – graded as 10 (perfect), 7 (small mistakes but on the right track), 3 (clearly wrong but a decent attempt), or 0 (otherwise)]

Study the following code, then answer the question below.

```
public int question2(int x, int y) {
      System.out.println("x is "+x);
      System.out.println("y is "+y);
      int answer = 0;
      if (2*x < y) {
            System.out.println("2*x is "+(2*x));
            answer = -1;
      }
      else if (2*y < x) {
            System.out.println("2*y is "+(2*y));
            answer = 1;
      }
      if (x == y) {
            System.out.println("YES");
      }
      else {
            System.out.println("NO");
      }
      return answer;
}
```

What is printed when `question2(12, 4)` is executed?

```
X is 12
y is 4
2*y is 8
NO
```

What value is returned when `question2(12, 4)` is executed?

```
1
```

Question 3 [10 points, 2 points each]

[GRADING: 10 total marks – graded as 10 (perfect), 7 (small mistakes but on the right track), 3 (clearly wrong but a decent attempt), or 0 (otherwise)]

The U.S National Library of Medicine publishes guidelines for blood sodium levels. These guidelines are summarized in this table:

Blood sodium level	Category
135 to 145 mEq/L	normal
above 145 mEq/L	hypernatremia
below 135 mEq/L	hyponatremia

Define a method named `sodiumLevel` so it has one parameter, of type `int`. Define the method so it returns, as a `String`, the category associated with the sodium value (an integer representing the blood sodium level, expressed in units mEq/L) passed in as an argument.

For example, the result of running this code,

```
System.out.println(sodiumLevel(134));
System.out.println(sodiumLevel(142));
System.out.println(sodiumLevel(153));
```

must be:

```
hyponatremia
normal
hypernatremia
```

```
public String sodiumLevel(int x) {
    if (x < 135) {
        return "hyponatremia";
    }
    if (x > 145) {
        return "hypernatremia";
    }
    return "normal";
}
```

Question 4 [10 points total]

[GRADING: 10 total marks – graded as 10 (perfect), 7 (small mistakes but on the right track), 3 (clearly wrong but a decent attempt), or 0 (otherwise)]

Define a class LipoProteinPanel so that a LipoProteinPanel object stores the results of a cholesterol test: total cholesterol, LDL cholesterol and HDL cholesterol. Each of these individual results is given by an int value. Don't forget to define a constructor that lets these three values be provided as arguments, as in:

```
LipoProteinPanel p = new LipoProteinPanel(190, 90,74);
```

The order or arguments must be total cholesterol, LDL cholesterol and HDL cholesterol. Finally, define a no-argument method named 'normal' that returns true if total cholesterol is less than 200 and LDL cholesterol is less than 130 and HDH cholesterol is greater than or equal to 40. The method must return false otherwise.

```
package exam2;

public class LipoProteinPanel {

	private int _total;
	private int _ldl;
	private int _hdl;

	public LipoProteinPanel(int tot, int ldl, int hdl) {
		_total = tot;
		_ldl = ldl;
		_hdl = hdl;
	}

	public boolean normal() {
		return _total < 200 && _ldl < 130 && _hdl >= 40;
	}

}
```

Question 5 [10 points]

[GRADING: 10 total marks – graded as 10 (perfect), 7 (small mistakes but on the right track), 3 (clearly wrong but a decent attempt), or 0 (otherwise)]

Assume that the LipoProteinPanel class discussed in question 4 has been correctly defined. Define a method named flag that takes an ArrayList<LipoProteinPanel> and returns a new ArrayList<LipoProteinPanel>. The returned ArrayList must contain all the LipoProteinPanel objects in the argument list for which the normal method returns false.

For example:

```
ArrayList<LipoProteinPanel> input;
input = new ArrayList<LipoProteinPanel>();

LipoProteinPanel a = new LipoProteinPanel(190,90,74);
LipoProteinPanel b = new LipoProteinPanel(205,90,74));
LipoProteinPanel c = new LipoProteinPanel(190,141,74));

input.add(a);
input.add(b);
input.add(c);

ArrayList<LipoProteinPanel> actual = flag(input);
```

Actual must contain exactly b and c. In particular, actual must not contain a.

```
public ArrayList<LipoProteinPanel> flag(ArrayList<LipoProteinPanel> results) {

	ArrayList<LipoProteinPanel> flaggedResults = new ArrayList<>();

	for (LipoProteinPanel p : results) {

		if (!p.normal()) {
			flaggedResults.add(p);
		}

	}

	return flaggedResults;
}
```