Midterm Examination SOLUTION KEY

This is a closed-book-closed-notes exam
You may use a crib sheet no more than 2 pages long.

Please check that you have all 7 pages (the last page is blank).

<table>
<thead>
<tr>
<th>Part 1 (20 points)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>Part 2 (20 points)</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Part 3 (15 points)</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Part 4 (25 points)</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Total (80 points)

Read these instructions first:

1. When starting to work on a problem – **read the problem carefully and make sure you understand what you’re asked to do.** You should ask the instructor to clarify anything that is unclear in the problem statement.

2. **Always show your work** – that is the only basis for partial credit.

3. **Do not blindly copy from your crib sheet** – THINK! Valid ideas count for more than blind copying – as they are **your own**!

4. **Always check your answers and test your code on a sample input.**

5. **Minor syntax errors such as, for instance, missing ; in your code will not be counted as an error.**

Good luck!
Part 1. (20 points) The basics. Show your work for partial credit

(a) Consider the following code segment:

```java
int p = 30;
int b;

System.out.print("Please enter an integer");
b = SavitchIn.readLineInt();
if (b > p || b < 0) {
    p = 2*p;
    b = b/25;
} else {
    p = 0;
    b = b%2;
}
```

Assume the user entered 55

What is the value of variable \( p \) after the execution of this program segment?

60

What is the value of variable \( b \) after the execution of this program segment?

2 (the result of integer division of 55 by 25)

(b) What will be printed by the following code segment:

```java
int a = 35;
int b = 100;
boolean flag = false;
if ( !flag || a > b )
    System.out.println ("One");
else
    System.out.println ("Two");
```

One

(c) What will be printed by the following code segment:

```java
int count = 5;
int i;

for (count = 5; count >= 1; count--)
    for (i = 1; i <= count; i++)
        System.out.print(count + " ");
    System.out.println ("*");
```

5 5 5 5 5 *
4 4 4 4 *
3 3 3 *
2 2 *
1 *
(d) What is stored in the array \texttt{fib} after the execution of the following code segment?

```java
double[] fib = new double[10];
int i;

fib[0] = 2.0;
fib[1] = 3.0;

fib[2] = fib[1] + fib[0];
i = 3;
while (i < fib.length) {
    fib[i] = fib[i-1] + fib[i-2];
    i++;
}
```

array \texttt{fib} contains 10 elements:

```
|   2.0  |   3.0  |   5.0  |   8.0  |  13.0  |  21.0  |  34.0  |  55.0  |  89.0  | 144.0  |
```

(e) Explain why the following code segment will generate a compiler error (assume it is a part of an otherwise properly composed main method).

```java
String str = "STOP";
int i = 0;
switch (str) {
    case "START":
        i = 1;
        break;
    case "STOP":
        i = 10;
        break;
    default:
        System.out.println("Error.");
}
```

The syntax rules state that the switch statement cannot be done over an expression of type \texttt{String}, which is the case here since \texttt{str} has type \texttt{String}.

This is the \textit{only} syntax error here – missing break statement is not a problem in terms of the syntax.
Part 2. (20 points) Methods

What is displayed when the following code segment is executed? Show the output in the box on the bottom of the page. Show the values of participating variables and parameters for partial credit.

```java
public class Silly {
    public static void main (String[] args){
        char ch = 'A';
        int offset = 2;
        System.out.println ("ch = " + ch);
        System.out.println ("offset = " + offset);

        ch = FromLast(offset, "Mexico");
        System.out.println ("ch = " + ch);
        System.out.println ("offset = " + offset);
    }

    public static char FromLast (int pos, String str){
        char ch;
        int len = str.length();
        pos = len - pos;
        if (pos > len || pos <= 0){
            ch = '$';
        } else {
            ch = str.charAt(pos);
        }
        System.out.println("Character at position " + pos);
        System.out.println(" is " + ch);
        return ch;
    }
}
```

```
ch = A
offset = 2
Character at position 4
is c
ch = c
offset = 2
```
Part 3. (15 points) In this problem you will develop a static method LetterGrade that returns a character corresponding to a grade value that is passed into the method as a parameter. The following chart describes the correspondence between a numeric grade and its letter equivalent.

<table>
<thead>
<tr>
<th>Grade Range</th>
<th>Letter</th>
</tr>
</thead>
<tbody>
<tr>
<td>90 &lt;= grade</td>
<td>A</td>
</tr>
<tr>
<td>80 &lt;= grade &lt; 90</td>
<td>B</td>
</tr>
<tr>
<td>70 &lt;= grade &lt; 80</td>
<td>C</td>
</tr>
<tr>
<td>60 &lt;= grade &lt; 70</td>
<td>D</td>
</tr>
<tr>
<td>grade &lt; 60</td>
<td>F</td>
</tr>
</tbody>
</table>

For example, when passed value 85, the method must return ‘B’, when passed 99, it must return ‘A’.

The template for the method is provided below.

```java
public static char LetterGrade ( int grade )
{
    char letter;
    if (grade >= 90)
        letter = 'A';
    else if (grade >= 80)
        letter = 'B';
    else if (grade >= 70)
        letter = 'C';
    else if (grade >= 60)
        letter = 'D';
    else
        letter = 'F';
    return letter;
}
```
Part 4. (25 points) Write a complete program that computes the number of A’s on the test given a list of grades and checks if the grades are entered in the non-decreasing order.

The program must at first read the number of grades in the list (n) followed by exactly n numeric grade values. The program must compute and print

1. how many grades were greater or equal to 90,
2. whether or not the grades were entered in non-decreasing order, i.e. if each next grade was equal to or greater than the previous.

For example, given the following user input: 5 80 70 90 95 76 (where 5 is the number of grades, and the grades are 80, 70, 90, 95, and 76) the program must print

There were 2 A-s,
The grades were not entered in non-decreasing order.

since there were only two grades equal or greater than 90, and the second grade (70) was lower than the one before it (80).

Here’s another example. For user input 5 80 82 90 90 99 the output should be

There were 3 A-s,
The grades were entered in non-decreasing order.

To get full credit, your program must use the method LetterGrade from Part 3. A solution that does not use this function will get at most 20 points.

Your program doesn’t have to contain comments, but must otherwise use good programming style.
public class GradeReport{

    public static void main (String[] args) {
        int numGrades; // number of grades in the list
        System.out.println ("Please enter how many grades");
        numGrades = SavitchIn.readLineInt();

        int gr; // grade that was just entered
        int prevgr; // previous grade

        boolean nonDecreasing = true; // flag true iff all
        // grades so far were //entered in non-decreasing order

        int numAs = 0; // number of A's in the list

        gr = -1; // set it to 1 so that the first grade
        // that is entered is greater than prevgr

        // read and analyze exactly numGrades grades
        for (int i = 1; i <= numGrades; i++) {
            // update the previous grade, before reading the next
            prevgr = gr;

            System.out.println ("Please enter next grade");
            gr = SavitchIn.readLineInt();

            if (LetterGrade(gr) == 'A')
                numAs++;

            // check if non-decreasing order is violated
            // by comparing the previous grade to one just entered
            if (prevgr > gr) {
                nonDecreasing = false;
            }
        }

        System.out.println ("There were " + numAs + " A-s");

        if (nonDecreasing == true) // same as if (nonDecreasing)
            System.out.println("The grades were entered in non-
            decreasing order.");
        else
            System.out.println("The grades were not entered in
            non-decreasing order.");

    }

    // definition of method LetterGrade from Part 3 goes here.
    //
}