INSTRUCTIONS TO CANDIDATES

1. This examination paper contains NINE (9) questions and comprises FOURTEEN (14) printed pages, including this page.

2. Answer ALL questions within the space provided in this booklet. Use the reverse sides if necessary.

3. This is an OPEN BOOK examination.

4. Write legibly with a pen or pencil.

5. Do not tear off any pages from this booklet.

6. Please write your Matriculation Number below.

MATRICULATION NO: ___________________________________

This portion is for examiner’s use only

<table>
<thead>
<tr>
<th>Question</th>
<th>Marks</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1 to Q5 (25 Marks)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q6 (30 Marks)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q7 (18 Marks)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q8 (13 Marks)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Q9 (14 Marks)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SECTION A (5 questions. Each 5 marks. Total 25 marks)

Q1 What is the output of the following program fragment?

```java
for ( int i = 0; i <= 1; i++ ) {
    try {
        System.out.println( "Is Even?" );
        if ( i % 2 == 0 )
            throw new Exception();
        System.out.println( "Is Odd?" );
        if ( i % 2 == 1 )
            throw new ArithmeticException();
    } catch ( Exception e ) {
        System.out.println( "Even" );
    } catch ( ArithmeticException e ) {
        System.out.println( "Odd" );
    }
}
```

ANS:

Q2 Write a regular expression for a vehicle license plate number. The plate number is made up of 3 uppercase alphabet letters, followed by 1 to 4 digits, and followed by an uppercase alphabet letter. The alphabet letters ‘I’ and ‘O’ cannot be used in the plate number. Your regular expression should not allow leading zeros in the numeric number. For example, XYZ0038W is not allowed.

ANS: ________________________________
Q3 Assume that all the integer values in the `data` array are different and the size of the `data` array is at least 2. What does the following method do? Describe in a sentence.

```java
public static int mystery( int[] data ) {
    int x = data[0];
    for ( int i = 1; i < data.length; i++ )
        if ( x < data[i])
            x = data[i];

    int y = ( data[0] != x ) ? data[0] : data[1];
    for ( int i = 1; i < data.length; i++ )
        if ( y < data[i] && data[i] < x )
            y = data[i];

    return y;
}
```

ANS: 

______________________________

______________________________

Q4 Write a public static method, `hasDuplicate`, that returns `true` if and only if at least two of the integer values in the input array are equal.

ANS:

```java
public static boolean hasDuplicate( int[] x ) {
    // Your implementation here
}
```
Q5 Given the following class definitions of C1 and C2.

```java
class C1 {
    public int x;
    public C1( int i ) {
        x = i;
    }
}

class C2 {
    public void hello( C1 c1 ) {
        c1.x = 789;
        c1 = new C1( 123 );
        System.out.println( c1.x );
    }
}
```

What is the output of the following program fragment?

```java
C1 c1 = new C1( 456 );
C2 c2 = new C2();
System.out.println( c1.x );
c2.hello( c1 );
System.out.println( c1.x );
```

ANS:
SECTION B (4 questions. Total 75 marks)

Q6 (30 Marks)
The University of Noef has adopted a grading system in which every student is given one of the following five grades for each module taken in the university:

<table>
<thead>
<tr>
<th>Grade</th>
<th>Grade Points</th>
<th>Grade</th>
<th>Grade Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>A+</td>
<td>5</td>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>A</td>
<td>4</td>
<td>D</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>3</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Each grade is assigned grade points as shown in the above table. Each module has a module code, such as “CS1101”, and is assigned a number of modular credits to indicate the module’s workload. The number of modular credits can be a non-integer number. The overall performance of a student is indicated by the Cumulative Average Point (CAP) of all the modules he has taken. CAP is computed using the following formula:

\[
\text{CAP} = \frac{\text{Sum (module grade point } \times \text{ modular credits for the module)}}{\text{Sum (modular credits)}}
\]

Write a Java class to store a student’s module grades and to compute the CAP. The class is named StudentGrades, and it should provide the following constructor and public instance methods:

```java
public StudentGrades( int maxNumModules );
Constructor. It assumes that the maximum possible number of modules that can be taken by the student is maxNumModules.
```

```java
public void add( String moduleCode, double modularCredits, String grade );
Adds a module and the student’s grade for the module. The module’s code is moduleCode, its number of modular credits is modularCredits, and the student’s grade for the module is grade. You can assume that all the alphabet letters in the input moduleCode and grade will always be in uppercase.
```

```java
public int getNumModules();
Returns the number of modules that have been added.
```

```java
public String getGrade( String moduleCode );
Returns the student’s grade for the input module moduleCode. If the input module is not found, returns a null.
```

```java
public double getCAP();
Returns the CAP of all the modules that have been added. If no module has been added, returns 0. Your method should not recompute the CAP every time it is called. It should compute or recompute the CAP only when it is called the first time after the StudentGrades object is created or after a module has been added. Subsequent calls should just return the pre-computed CAP, without recomputing the CAP.
```
(a) **(5 Marks)** Write the declarations of the instance data members of the `StudentGrades` class.

(b) **(5 Marks)** Define the constructor.

```java
public StudentGrades( int maxNumModules ) {
```
(c) **(7 Marks)** Define the `add()` method.

```java
public void add( String moduleCode, double modularCredits,
                 String grade ) {
```

(d) **(1 Mark)** Define the `getNumModules()` method.

```java
public int getNumModules() {
```
(e) **(4 Marks)** Define the `getGrade()` method.

```java
public String getGrade( String moduleCode ) {
```

(f) **(8 Marks)** Define the `getCAP()` method.

```java
public double getCAP() {
```
Q7 (18 Marks)
For this question, you will implement two different recursive methods to compute and return the value of $x^k$, where $x$ can be any real number and $k$ can be any non-negative integer.

(a) (5 Marks) Write a recursive public static method, named `intPow1` to compute and return the value of $x^k$, using the following recursive definition of $x^k$:

$$x^k = x \cdot x^{k-1}$$

Do not use the `Math.pow()` method.

public static double intPow1(double x, int k) {
(b) **(8 Marks)** Write a recursive **public static method**, named `intPow2` to compute and return the value of \( x^k \), using the following recursive definition of \( x^k \):

\[
x^k = \begin{cases} 
  x^{k/2} \cdot x^{k/2} & \text{if } k \text{ is even} \\
  x^{[k/2]} \cdot x^{[k/2]} \cdot x & \text{if } k \text{ is odd}
\end{cases}
\]

Do not use the `Math.pow()` or the `Math.floor()` method.

```java
public static double intPow2( double x, int k ) {
```

(c) **(5 Marks)** How many floating-point multiplication operations are performed by `intPow1()` and `intPow2()` to compute \( x^{13} \) and \( x^{128} \)? Write your answers in the table below.

<table>
<thead>
<tr>
<th>Method</th>
<th>Number of FP Multiplications</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>intPow1( x, 13 )</code></td>
<td></td>
</tr>
<tr>
<td><code>intPow1( x, 128 )</code></td>
<td></td>
</tr>
<tr>
<td><code>intPow2( x, 13 )</code></td>
<td></td>
</tr>
<tr>
<td><code>intPow2( x, 128 )</code></td>
<td></td>
</tr>
</tbody>
</table>
Q8 (13 Marks)
Write a public static method to compare two strings. You can assume that the two strings contain only alphabet letters. A string $X$ is “smaller” than a string $Y$ if $X$ comes before $Y$ in alphabetical or “dictionary” order. The comparison must be case-insensitive. You cannot make use of the `compareTo()` or the `compareToIgnoreCase()` method in the `String` class. Use only the `String` instance methods `length()`, `toUpperCase()`, and `charAt()`.

The method header should be

```java
public static int myStrCompare( String s1, String s2 );
```

If $s1$ is smaller than $s2$, the method returns $-1$; if $s1$ and $s2$ are the same (case-insensitively), the method returns $0$; otherwise, it returns $1$. Note that $s1$ and $s2$ can be of different lengths.

```java
public static int myStrCompare( String s1, String s2 ) {
```
Q9 (14 Marks)
A grayscale image is made up of a 2D rectangular array of pixels, where each pixel’s value corresponds to a shade of gray. Very commonly, the pixel value 0 is used to represent black, and the value 255 is used to represent white, and the other values between 0 and 255 represent different shades of gray. This way, a pixel value can be stored in a byte, and an entire 2D grayscale image can be stored in a 2D rectangular array of bytes. For example, the image array declared below can be used to store an image of size 5 rows by 7 columns.

```java
byte[][] image = new byte[5][7];
```
The top-left-most pixel is stored in `image[0][0]`.

(a) (6 Marks) Write a public static method to compute the average pixel value of a square block of pixels of an image. The method header should be

```java
public static byte meanPixel( byte[][] image, int blkRow, int blkCol, int blkWidth );
```
where `image` is the input image, `blkRow` is the row index of the top-left pixel in the square block, `blkCol` is the column index of the top-left pixel in the square block, and `blkWidth` is the number of pixels on each side of the square block. For example, `meanPixel(image, 2, 1, 3)` returns the average pixel value of the dark-shaded block shown in the diagram below. Note that the returned average pixel value must be rounded to the nearest whole number. You can use the `Math.round()` static method. You can assume that the input block is always inside the image.

```java
public static byte meanPixel( byte[][] image, int blkRow, int blkCol, int blkWidth ) {
```
(b) (8 Marks) Write a public static method to shrink an image. The method header should be

```java
public static byte[][] shrinkImage( byte[][] image,
                                    int factor );
```

where `image` is the input original image, and `factor` is the reciprocal of the scaling factor. The method returns the shrunk image as a 2D array of `byte`. For example, if `factor` is 3, then the shrunk image will be 1/3 the width and height of the original image. You can assume that the width and height of the original image are multiples of `factor`.

Each pixel in the shrunk image is the rounded mean pixel value of the corresponding `factor` × `factor` block of pixels in the original image. Therefore, you should make use of the `meanPixel()` method that you wrote in Part (a). The following diagram shows an example in which `factor` is 2.

![Diagram showing an example of shrinking an image with a factor of 2.](image.png)

(continued on next page)
public static byte[][] shrinkImage( byte[][] image, int factor ) {