Cardiff School of Computer Science and Informatics

Coursework Assessment Pro-forma

Module Code:	CM1210
Module Title:	Object Oriented Java Programming
Lecturer:	Matt Morgan
Assessment Title:	Java Implementation Skills
Date Set:	4th March 2021
Submission Date and Time:	26th March 2021 at 09:30am
Return Date:	26th April 2021

This coursework is worth 50% of the total marks available for this module. If coursework is submitted late (and where there are no extenuating circumstances):

1. If the assessment is submitted no later than 24 hours after the deadline, the mark for the assessment will be capped at the minimum pass mark;
2. If the assessment is submitted more than 24 hours after the deadline, a mark of 0 will be given for the assessment.

Your submission must include the official Coursework Submission Cover sheet, which can be found here:

https://docs.cs.cf.ac.uk/downloads/coursework/Coversheet.pdf

Submission Instructions

You should submit your completed coursework under the "Assessment" tab in the CM1210 module on Learning Central. Your submission should include the following files:

Description		Type	Name
Cover sheet	**Compulsory**	One PDF (.pdf) file	[Student number].pdf
Q1a	**Compulsory**	A zip archive containing one or more Java source file(s) (.java) containing your answers to question 1a. Each source code file should contain your name and student number in a comment.	Q1a_[Student number].zip
Q1b	**Compulsory**	A zip archive containing one or more Java source file(s) (.java) containing your answers to question 1b. Each source code file should contain your name and student number in a comment.	Q1b_[Student number].zip
Q2	**Compulsory**	One PDF (.pdf) file containing a report, no more than 4 pages long.	Q2_[Student number].pdf

Any code submitted will be run on a system with Java JDK 15 installed and must be submitted as stipulated in the instructions above.

Any deviation from the submission instructions above (including the number and types of files submitted) will result in a reduction in marks for that question part of 20%.

Staff reserve the right to invite students to a meeting to discuss coursework submissions

Assignment

1. Any $n \times n$ magic square (where n is an odd integer) consists of an $n \times n$ matrix whose elements contain the numbers $1, 2, 3, ..., n^2$ such that the sum of each row, column and diagonal is equal to $\frac{n(n^2+1)}{2}$. For example, the following magic square for $n = 3$, with the sum of each row, column and diagonal being $\frac{3(3^2+1)}{2} = 15$:

6	1	8
7	5	3
2	9	4

   (a) An algorithm for generating an $n \times n$ magic square for odd n is as follows:

   NOTE: Assume the rows and columns wrap around (i.e., moving one column left from the first column gives the last column)

   Create a 2 dimensional array of size n by n and set all values to be 0
   Set $x = 1$, $y = \frac{n+1}{2}$ (row 1 and column $\frac{n+1}{2}$).
   Insert 1 at x, y
   for i = 2 to n^2 **do**
      if element $x - 1$,$y - 1$ is empty (i.e. $= 0$) **then**
         $x = x - 1$, $y = y - 1$
      else
         $x = x + 1$, $y = y$
      end if
      Insert i at x, y
   end for

   Write a **command line application** that prompts the user for an **odd** integer and displays a magic square of that size to standard output (i.e. the command line).

   [*HINTS: Recall that Java arrays start at the element* 0*; it may help to define a class to store a square matrix*].

   (12 Marks)
   [Functionality: 5, Design: 5, Ease of use: 1, Presentation: 1

   (b) Write a **command line** game with the following functionality:

   - The application should prompt the user for an odd integer and create a magic square of that size.
   - The magic square should then be shuffled by repeatedly (for n^2 times) choosing a random element and swapping it with a random neighbour (**not** including diagonals).
   - The shuffled square should be displayed to the user, who must attempt to reconstruct a magic square.
   - The user makes moves by giving input of the form:

   i j direction

   where i and j specify the row and column of an element to be swapped, and *direction* (either U, D ,L ,R representing *up*, *down*, $left$ and *right*) specifies which direction it should be swapped with. For example, the move 2 1 D applied to the square above would give:

6	1	8
2	5	3
7	9	4

- On completion, the game should report the number of moves made.

(18 Marks)

[Functionality: 5, Design: 10, Ease of use: 2, Presentation: 1

HINT: MULTIDIMENSIONAL ARRAYS

The code below gives an example of using a 2-dimensional array to store values:

```
public class MultiArrayTest
{
   public static void main( String[] args )
   {
      int a[][] = {{1,2,3}, {4,5,6}};
      System.out.println( "length of a is " + a.length );
      for (int i = 0; i < a.length; i++)
      {
        for (int j = 0; j < a[i].length; j++)
        {
           System.out.print( a[i][j] + " " );
        }
        System.out.println();
      }
   }
}
```

---

2. Write a **report** that explains what you've done and how you did it. Your report should have **four sections**:

   (a) a brief overview of the assignment and your goals in completing it.

   (b) a description of your solution design, including any assumptions made, algorithms used, etc.

   (c) a discussion of your completed solution to the assignment, including the scope of solution, quality of solution, interesting results, difficulties overcome, enhancements delivered, etc.,

   (d) a discussion of your software test methodology. How did you ensure that your solution does what it's meant to do.

   Your report does not have to be long. On the contrary, you should strive for conciseness and clarity in your writing. However, details of page limits are given in the "Submission Instructions" section above.

(20 Marks)

[Total Marks: 50]

[End of Questions]

Code Reuse

Your solutions may make use of any classes in the Core Java API. You may also reproduce small pieces of code from:

- The CM1210 course handouts, solutions and videos.
- java.oracle.com
- any textbooks

provided:

- The section reproduced does not form the entire solution to a single question
- The source of the code is **clearly referenced** in your source code
- Your code is commented to demonstrate clearly that you understand how the reproduced code works (i.e., explain why types have been selected, why other language features have been used, etc.)

You may **NOT** reproduce code written by any other student or code downloaded from any other website. If you are in any doubt about whether you may include a piece of code that you have not written yourself, please ask before submitting.

Criteria For Assessment

Credit will be awarded against the following criteria:

Functionality

- To what extent does the program perform the task(s) required by the question.

Design

- How well designed is the code, particularly with respect to the ease with which it may be maintained or extended (delivering Object Oriented design will help here).

In particular, consideration will be given to:

- Use of appropriate types, program control structures and classes from the core API.
- Definition of appropriate classes, interfaces and methods.

Ease of use

- Formatting of input/output.
- Interaction with the user.
- How does the code deal with invalid user input? Will the applications crash for certain data?

Documentation and presentation

- Appropriate use of comments.
- Readability of code.

Feedback and Suggestion for Future Learning

Feedback on your coursework will address the above criteria. Feedback and marks will be returned on Monday 26th April 2021 via Learning Central.