***Exam Coversheet***

**To be completed by the student:**

Studentnumber:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name:

\*2412mm132A\*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Group/Class:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |
| --- | --- |
| **To be completed by the lecturer** Exam Name | Software Engineering |
| Exam Code | Mock Exam |
|  |  |
| Exam Date | Mock Exam |
| Time | Mock Exam |
| Classroom | Mock Exam |
| Length (in minutes) | Mock Exam |
|  |  |
| Academic Year | 2 |
| Education Period | Mock Exam |
| Opportunity/Chance | Yes |
|  |  |
| Examiners | H. Drillenburg Lelyveld |
|  |  |
|  |  |
| Faculty | Engineering, Design and Computing |
| Cluster | ICT |
| Programme + full time/part time | Mathematical Engineering |
| Location | Diemen |
|  |  |
| Number of pages (incl. coversheet) | 6 |
| Number of questions | 7 |
| Passmark | 9 /1/2 out of 17points |

|  |  |
| --- | --- |
| Allowed tools/aids |  |
| Answer sheet |  |
| Additional comments/details |  |

(\* delete if not applicable)

**In te vullen door de toetsorganisatie:**

Surveillanten:

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

Bijzondere voorzieningen:

|  |  |  |
| --- | --- | --- |
| **Voorziening:** | **Aantal:** | **Opmerking:** |
| Dyslexie |  |  |
| A3 |  |  |
| Overig |  |  |

# Exam Software Engineering, 2015-2016, term 2

**Guidelines**

With several of the following questions, you are asked to refer to the following case study. This case study has been used extensively during class, and will be well-known to you by now. You can use the ideas developed during the course, you do not need to stick to the exact wording of the description here. When you have to make new assumptions, you are allowed to, as long as you mention them explicitly.

**Case study - MMORPG**

The startup company Shadowdancer intends to release a new Massive Multiplayer Online Roleplaying Game (MMORPG ) based on a series of novels the founder is writing. As a small startup, Shadowdancer does not have the resources to develop the software itself; instead, you are hired to do this development.

The release date of the first version of the software should be the same as the release of the first book of the series in order to gain maximum exposure. It's not a problem that this first release does not have all the features and options available. For example, magic will be a part of the world, but as in the storyline magic all but disappeared after the Last War and has to be reinvented, its inclusion in the first release is not necessary. However, the architecture of the application should allow for its later inclusion.

**Short description of the novels and game envisioned**

About a century ago (in the MMORPG timeline), a tremendous war has taken place. On the one side: the Gods of the world, who were for the most part feared, not loved. On the other side: mankind and the so-called Old Ones. The war has banished the Gods from this world, destroyed an area the size of the Netherlands and killed nearly all the Old Ones, as well as a whole lot of humans. The location of that battlefield is now a desert where nothing can live. (This is an area which is, at least at the start of the game, off-limits for all player characters).

In the far, far future (again, in the MMORPG's timeline) two schools of magic, the Black Pole and the White, are on the verge of dying out. The last mage of the Black Pole, Finsternis (the main character in the books) has to find out why this is happening, and must do what he can to restore the magic to his word. Meanwhile, the banished Gods are still trying to return to this world, to reclaim what they feel they own. Since all of this is in the game's future, these characters are – initially, at least – not relevant in the game.

I have added a few loosely sketched maps for you. Though the world extends beyond the map, all the actions in the novels play out in the area depicted there. Verre is situated in the north, between the river just under the mountains to the woods in the upper half of the map. A close-up of that area is depicted in the more detailed map. Bear in mind that I created the second map to help myself visualise the area in the novels – most villages and cities either do not exist yet, or might be located in different places. This gives you a lot of leeway to play with.

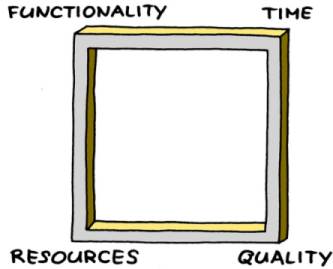
The white and black triangles at the top and bottom of the larger map are the future locations of magic schools: White Pole in the north, Black Pole in the south. Player characters will be able to choose to become mages, and help build these schools. However, this is not for the first version of the game. Gradually, in later versions, more hints to magic will have to be added.

The people in this world are humans. There are some other beings still in the world, called 'The Old ones' or 'The Ancients', but certainly in the first release of the game, none of these will be encountered by the players. There are no dwarves, elves, gnomes, orcs, etc. Their true enemies are the Gods, banished about a hundred years ago and seeking a way back in. However, this is the overall – enormous – storyline, so in the first releases of the game, this is not relevant. Initially, enemies can be people like highwaymen, people who terrorise a village, etc.

Initially, a player can level up and can specialise into different classes. They can discover magic – which, again, in the early releases will be hinted at –, build magic schools and study magic, they can specialise into other classes such as fighters, bards, rangers and rogues. No priests, for obvious reasons. Their long-term goals will be finding out what happened over a hundred years ago, and seeking to prevent the Gods' return.

**Question 1 (2 points)**

The following picture depicts the so-called square from hell:



Explain why in software development you have to choose two or three, instead of being able to give the customer all four.

**Question 2 (1 point)**

Explain in less than 40 words the concept of *refactoring* your code. Support your explanation by referring to the case study.

**Question 3 (1 point)**

When developing an online game like in the case study, the software development process used is often the Rational Unified Process. Explain the basic principles of this process.

**Question 4 (1 point)**

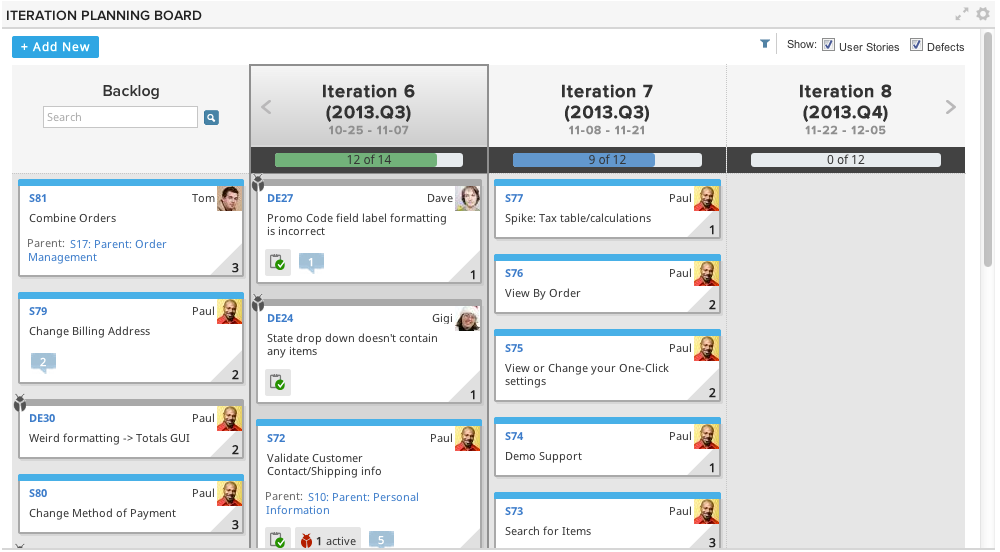
With Test Driven Development, you define and code tests before you even code the 'real' classes. Explain how you can test something that is not even available yet, and how that can dramatically improve the quality of your code.

**Question 5 (4 points)**

1. Take a general overview of the case study. In designing the overall structure of the application, one developer comes up with the idea of using a Command Design Pattern to increase the flexibility of the application, especially in the client side part. He mutters something about actions on different characters (personae) on the screen. Do you think this is a good idea? Explain your answer including a specific situation being helped or harmed by a Command pattern. (3 points)
2. Suppose the client side of the application from the case study is being developed using a MVC Design Pattern. Sketch the structure of the resulting architecture using an UML Class Diagram. Indicate the patterns which form the building blocks of the compound pattern. (1 point)

**Question 6 (5 points)**

1. Assume you just started the development of the MMORPG game as described in the case study. Make a list of at least five (5) functional requirements suitable for development. Then assume you are the customer wanting this. Prioritize the requirement using the most suitable priorization method. (1 point)
2. Shown below is an example of an iteration board:



Sketch an iteration board for the case study and divide the requirements over the board, using at least three iterations. (1 point)

1. Take one of the requirements for the first iteration and expand it into a user story (1 point)
2. Using planning poker, you made an iteration planning, using a velocity of 0.7. After two weeks, one of the three developers gets sick. It is not to be expected that he will recuperate within six weeks. Explain how to adapt your planning, using velocity as the central idea in your explanation (so don’t give a new planning, explain how you would create it.) (2 points)

**Question 7 (3 points)**

The following code is give for the utility class Matrix:

**public** **class** Matrix {

**private** **double** numbers[][];

**private** **int** width;

**private** **int** height;

**public** Matrix(**int** width, **int** height) {

**if** (width < 1) {

**throw** **new** IllegalArgumentException("Width cannot be less than 1");

}

**if** (height < 1) {

**throw** **new** IllegalArgumentException("Height cannot be less than 1");

}

**this**.width = width;

**this**.height = height;

numbers = **new** **double**[width][height];

**for** (**int** i=0; i<width; i++) {

**for** (**int** j=0; j<height; j++) {

numbers[i][j] = 0;

}

}

}

**public** **int** getWidth() {

**return** width;

}

**public** **int** getHeight() {

**return** height;

}

**public** **void** setNumber(**int** width, **int** height, **double** value) {

**if** (width < 1) {

**throw** **new** IllegalArgumentException("Width cannot be less than 1");

}

**if** (height < 1) {

**throw** **new** IllegalArgumentException("Height cannot be less than 1");

}

numbers[width-1][height-1] = value;

}

**public** **double** getNumber(**int** width, **int** height) {

**if** (width < 1) {

**throw** **new** IllegalArgumentException("Width cannot be less than 1");

}

**if** (height < 1) {

**throw** **new** IllegalArgumentException("Height cannot be less than 1");

}

**return** numbers[width-1][height-1];

}

**public** **void** scalarMultiply(**double** scalar) {

**for** (**int** i=0; i<width; i++) {

**for** (**int** j=0; j<height; j++) {

numbers[i-1][j-1] \*= scalar;

}

}

}

**public** **void** add(Matrix other) {

**if** (**this**.height != other.getHeight()) {

**throw** **new** IllegalArgumentException("Height of both matrices must be the same");

}

**if** (**this**.width != other.getWidth()) {

**throw** **new** IllegalArgumentException("Width of both matrices must be the same");

}

**for** (**int** i=0; i<width; i++) {

**for** (**int** j=0; j<height; j++) {

numbers[i-1][j-1] += other.getNumber(i, j);

}

}

}

**public** Matrix multiply(Matrix other) {

**if** (**this**.width != other.getHeight()) {

**throw** **new** IllegalArgumentException("Width of this matrix should be equal to height of other matrix");

}

Matrix result = **new** Matrix(**this**.height, other.getWidth());

**for** (**int** i=1; i<=width; i++) {

BoundVector resultColumn = **new** BoundVector(numbers[i]);

BoundVector resultRow = **new** BoundVector(height);

**for** (**int** j=1; j<=height; j++) {

resultRow.setCoordinate(j-1, other.getNumber(i, j));

}

result.setNumber(i, i, resultRow.inproduct(resultColumn));

}

**return** result;

}

@Override

**public** String toString() {

StringBuilder result = **new** StringBuilder();

**for** (**int** i=1; i<=width; i++) {

**for** (**int** j=1; j<=height; j++) {

result.append(**this**.getNumber(i, j));

result.append(" ");

}

result.append("\n");

}

**return** **new** String(result);

}

@Override

**public** **int** hashCode() {

// primitive implementation, does not work for larger matrices

**int** result = 1;

**for** (**int** i=1; i<=width; i++) {

**for** (**int** j=1; j<=height; j++) {

result = (**int**) (result \* **this**.getNumber(i, j)) \* i;

}

}

**return** result;

}

@Override

**public** **boolean** equals(Object other) {

**if** (!(other **instanceof** Matrix)) {

**return** **false**;

}

Matrix otherMatrix = (Matrix)other;

// Matrices are equal if all cells are equal

**for** (**int** i=1; i<=width; i++) {

**for** (**int** j=1; j<=height; j++) {

**if** (**this**.getNumber(i, i) != otherMatrix.getNumber(i, i)) {

**return** **false**;

}

}

}

**return** **true**;

}

}

There’s also a test class written, starting with:

import static org.junit.Assert.\*;

import math.BoundVector;

import org.junit.Before;

import org.junit.Test;

public class TestMatrix {

1. Write a JUnit method called initialize() preparing a 2x3 matrix, a 3x2 matrix and a 3x4 matrix. (1 point)
2. Write a JUnit test to test the method multiply(Matrix other) for the 2x3 matrix by the 3x2 matrix. (1 point). You do not have to calculate the real expected value for the matrix multiplication, you can use the pseudo-variable EXPECTED\_VALUE, which holds the hashCode for the expected matrix;
3. Write a JUnit test to test the method multiply for the 3x4 matrix times the 3x2 matrix, testing for an exception (which should be thrown.) (1 point)