UMUC - CMIS 242
Solution Description Document – Indications

For each programming project you are required to submit a Solution description document.

Solution description document is important because:
(1) It allows you to practice technical writing by professionally documenting a program that you have designed, implemented and tested;
(2) It is worth 20% of the project mark;

Below please find indications about the sections (1) ***Assumptions, Main Design Decisions and Error handling*** *and (2)* ***Test cases table*** *of this document. Please note that beside these sections the document should also include section (3)* ***Screen captures*** showing successful program compilation and test cases execution and *(4)* ***Lessons learned from the project.***

1. Assumptions, Design Decisions, Error handling

Assumptions

Here it is expected to indicate the (technical) assumptions you have made about the project.

These assumptions are actually establishing the frame of the implementation and may simplify the amount of work (code) required by the project.

For example, if you need input data of type integer from a file, you may assume that data items are separated by spaces and organized on lines. This way you may use the space for tokenizing the data.

However, with this assumption you have to test each token if it is an integer or not and generate an error (MalformedInputData exception) if not an integer value.

By using stronger assumptions your code may be further simplified. For example, you may assume that the data items are separated by spaces, organized on lines and each data item is a valid integer value.

Notes:
1. Many times, project specifications may clearly indicate some of the assumptions.
2. You may not assume things that contradict the specification (i.e. specification always rules).

Main Design Decisions

Here you are expected to describe and give reasons for the major project specific decisions you have taken during project design and how they improve the efficiency and/or help to better organize the project.

Examples of design decisions to be discussed:

-decisions about defining classes, their instance variables and methods (public, private, inner, static/non static, describe the meaning and the role of instance variables);

-decisions about the chosen data structures as instance variables of a class or as local variables in a method, details about the meaning and the role of instance or local variables;
-decisions about defining helper methods;
-decisions about method design (recursive, iterative), method parameters;
-decisions about the chosen algorithms;
-decisions about program control flow;

Error handling

List the errors (and their types) that are considered by the program and indicate how they are handled by the program.

2. Test cases table

According to Project specifications, a table of test cases should include the test cases that you have created to test the program. The table should include 5 columns indicating (i) what aspect is tested, (ii) the input values, (iii) the expected output, (iv) the actual output and (v) if the test case passed or failed.

Each test case will be defined in a table row.

Below is a skeleton of a test cases table for Project1 having defined one test case (**other relevant test cases should be defined by the students**).

Project 1 – Test cases table

What is tested	**Input**	**Expected Output**	**Actual Output**	**Pass / Fail**
Missing the input file (or wrong file name)	The file is not included in the current folder or a wrong name is used for the input file	A warning message indicating missing or incorrect input file name	?	?
…	…	…	…	…