Guidelines for Programming Project Documentation

The project does not only include the program itself, but also the documentation of the program. Writing a document gives people practise with producing "manuals" and "technical documentations", such as those required for real life programming projects. It is important that both the functionality and the implementation of a program are well illustrated for other people who might need this information – even a thoroughly commented source code will not be sufficient to describe larger, more complex software. The documentation will be used as the primary source of information, so designing it should not be taken lightly. The quality of the documentation has a considerable effect on the final project grade.

The project documentation should include the following sections and information:

- Title Page

  This page includes the course id and name, name of the student, name of the submitted person, project title, type of the project ("programming project"), and submission date.

- Introduction

  An introduction stating the problem definition and an overview of the solution detailed in the following sections of the document is given. It may also include some necessary terminology and definitions that will be used throughout the document; in the case that there are plenty of terminology and definitions, they may be written in a separate section.

- Program Interface

  This section explains how the user communicates with the program. It must be stated how the user will start running the program, i.e. how the programming environment will be set and activated, what is the command to be entered to execute the program, and what are the parameters (if any) that must accompany the execution command. It must also be stated how the user terminates the program. This section is just for activating and deactivating the programming environment; the details of the communication steps will be explained in the following sections.

- Program Execution

  This section explains in detail the execution of the program from an end-user's point of view. The descriptions of the algorithms, data structures, implementation, etc. should not be included. This section covers what types of inputs will be supplied to the program by the user, what are the outputs of the program, what is the functionality of the program, the descriptions of menu options and different parts of the program, etc. This section can be thought as the core of the "user manual" of the program. It is good practice to include screen outputs and some figures (e.g. expressing the structure and parts of the system). Do not give all the explanations in a single section; instead, divide this section into subsections for ease of reading.

- Input and Output

  The format of the input and output will be explained in this section. If the program also gets some input from input files and produces some output on output files, then the exact

format of the files must also be given. It is advisable to include some example inputs and outputs, and their explanations.

- Program Structure

  This is the "technical manual" of the program. An overall structure of the program should be given. Then, each part of the program (subprograms, modules, classes, etc.) should be explained in sufficient detail such that the reader can understand the statements, algorithmic logic, etc. The data structures (data types, variables, domains, etc.) should be explained. If desired, the explanation of the data structures can be covered in a separate section (this may be better if the entire program makes use of similar data structures) and this section is reserved for detailing the program code. Do not give all the explanations in a single section; instead, divide this section into subsections for ease of reading.

- Examples

  It will facilitate understanding the program and the document if some examples regarding the execution are included. For instance, a particular input can be given and the operations and the output of the program under this input may be specified. The examples should be chosen in a manner such that different behaviours of the program may be observed under different inputs.

- Improvements and Extensions

  In this section, the parts of the program that need improvement and some possible future extensions are discussed. The weak and strong points of the program can be identified. Any shortcomings or defects of the program can be stated. Also, the deviations from the plan at the beginning of the project can be indicated, i.e. the features, user options, data structures, etc. that were initially planned to be included, but could not be done so for some reasons.

- Difficulties Encountered

  It is quite valuable for an educational project to identify clearly the difficulties encountered within the project period. The difficulties due to the programming language (maybe it is a new language for you), due to the nature of the problem, etc. should be stated. Also, the effects of these difficulties (forcing to use different ways, abandoning some planned features of the program, etc.) should be explained.

- Conclusion

  This section is for the conclusions about the project.

- Appendices

  Program source code must be included as an appendix. Also, if necessary, some information can be given inside appendices. For instance, an appendix is a suitable place for the execution details for some complex sample executions.