Writing, formatting, and submitting your report

Writing your Report

This page complements the Panopto lecture recording (**Advice on writing up your dissertation)** on how to write your report. Samples of previous MSc projects can be found here: **Examples of Past Projects**

The Report will be read by at least two members of staff, and marks are awarded for the Report independently of any other work that you submit, whether that is software, data, or proofs. Usually, the two markers are your Supervisor and the member of staff who attended your Demonstration.

Suggested structure of the Report

The appropriate structure of the Report varies according to the scientific or engineering method that you have used, the features you have chosen to emphasise, and the degree you are pursuing. It is your responsibility to make sure that you are clear about where your project's contribution lies and that all work is explained clearly and in the correct format.

The following Report structure should therefore be seen as a guide only, supplementing the earlier lectures. Further, it is a guide that is specifically for software development type projects. It is probably the case that few Reports will stick to it rigidly. It is your responsibility to consult with your Supervisor and adapt it to suit your particular project. Types of problem solving project other than software development projects are likely to need a different structure.

1.**Title page** , including title, author, student ID, degree (MSc in ....), name of Supervisor, name of institution ('School of Computer Science, University of Birmingham'), date.

2.**Preamble**, including (a) Table of Contents; (b) Abstract/synopsis (suggested length: half a page); (c) Acknowledgements.

Abstract should be a succinct and self-standing summary of the basis and achievements of the project. Minimally an abstract does three things: (1) it states the problem that you set out to solve, (2) it describes your solution and method, (3) it states a conclusion about the success of the solution. Be straightforward and factual and avoid vague statements, confusing details and "hype". Do not be tempted to use acronyms or jargon to keep within the half-page limit. Consider that search engines, librarians and non-computer scientists wishing to classify your Report rely on the abstract. You may if you wish provide a short list of keywords (2-6 is reasonable) at the end of the abstract.

3.**Introduction**. In this section, you should describe the problem that you set out to solve with the project. An introduction might, for example, begin by stating, "The aim of the work described in

this Report was to provide a software tool with which people can arrange meetings." Avoid starting a Report with an irrelevant history of information technology. For example, the following would not be a good introductory sentence, "Since Bill Gates launched Outlook people have been using technology to arrange meetings."

Explain whatever background the reader will need in order to understand the problem. The background might refer to previous work in the academic literature that provides evidence that the problem is a real and significant problem worth solving. Include a clear and detailed statement of the project aims and provide an overview of the structure of the solution.

Conventionally, the last part of the introduction outlines the remainder of the Report, explaining what comes in each section.

4.The next set of sections form the main part of the Report, and will normally cover the following topics:

Further background material. It is often appropriate to provide more information than was given in your Introduction. Try to limit yourself just to what the reader needs to know to understand the solution that you have developed in your project. Put your work in the context of related existing work, commercial products, and research papers (if relevant).

Analysis and Specification. How you analysed the problem, including user requirements. Give an appropriate specification of the solution. For projects involving software development you may include a formal Software Requirements Specification in an Appendix.

Design If it is a software development project then give a high-level account of the structure of your software and how it works. What algorithms does it use? How do these compare with alternatives? What were the main design decisions you took, and their justifications?

Implementation and testing A detailed account of the implementation and testing of your software. Explain the conceptual structure of the algorithms. Also explain what data structures you used, and how the algorithms were implemented. What implementation decisions did you take, and why? There is no need to list every little function and procedure and explain its working in elaborate detail; use your judgement on what is appropriate to include. Explain your testing strategy and provide convincing examples of tests. In project involving software development it is usual to relate the tests to your Software Requirements Specification.

User interface If the interface forms a substantial part of your project (e.g. an HCI project), you will wish to include quite a lot of detail and explanation. In other projects the interface may be less of an important focus.

Project management. How have you managed your project and the writing of a substantial piece of software? Discuss the appropriateness of your methods in the light of your experience on the project.

Results and evaluation. Describe and present the results of your work in an appropriate form (proofs, outputs generated by software, screen shots, etc). It is most important that you evaluate the success of your software. How does it compare with the original specification? How reliable is

it? Comment on its robustness (e.g., to unexpected data), performance (e.g., response times or processing times in different situations), etc. In most cases you will want to include evaluation of your product (software, manuals, etc) by potential users.

5.**Discussion.** Here you will summarise your achievements and also the deficiencies of your project. You can also say what you would or could have done, if you had had more time or if things had worked out differently. It is important to be completely honest about the deficiencies and inadequacies of your work, such as they are. Part of your aim is to demonstrate your ability to recognise problems that remain.

6.**Conclusion.** Give a brief statement of how the solution that you have provided addresses the problem stated in the introduction. Provide an evaluative statement based on the results. You should not introduce new material.

7.**References.** For your Final Year project it is required that you cite and reference work to which you owe an intellectual debt. It is required that you cite and reference work that provides supporting evidence. It is required that you cite and reference work so that the reader can find the sources that have been quoted.

You must also include a section that explains the sources of all your submitted code: Which parts are your own original code? Which parts are included or adapted from other sources, and what are those sources? Which parts are automatically generated?

8.**Appendices.** The Report must contain an appendix explaining file structure of the code in the git repository. The appendix must also contain an information on how the code should be run. Other appendices may include documents such as: the project proposal; Software Requirements Specification; a selection of experimental data; schedules; testing strategy; risk management plans; glossary; manual; etc. Don't include the source code as an appendix (submit it as a part of the git repository). Don't include voluminous appendices.

9.**Referencing.** It is required that you provide citations and references. The University of Birmingham encourages the use of the Harvard System, but whatever system you use you must be consistent. Look up how to cite in text and how to provide a reference at the end of the Report.

What you need to submit

You will need to submit your Report in electronic form as a PDF fila via Canvas. The name of the pdf file must include your surname and student ID. No hard copy submissions are required.

The Report must contain an appendix (maximum one page) which includes

- the address to your git project repository
- contents of the git project repository and an information on how to run your software

Length

The Report should not exceed 60 pages (10 pt font), including references but excluding appendices. An excellent Report will often be less than this (its author will have said a lot in a

small amount of space). There is no need to submit double-spaced. Single-spaced is much better. Double-sided is encouraged.

Style

All parts of your Report should be precise and clear. You should think carefully about each section; what do you want to say in it, and what is a good order to say it in? Make sure you use complete sentences. There is plenty of advice available on good technical writing available in the library. Carefully spell-check your report. Assessment

How to submit your report

Submit your electronic form of your Report (a .pdf file) via Canvas. The name of the pdf file must include your surname and student ID.

Penalty for late submission

There is a penalty system for late hand-ins: 5 penalty points per working day are deducted up to a final Cut-Off Date for handing in the Report (see Important Dates at the top of this page). If the application of the penalty takes the project below the pass mark, the project will fail. No project will be accepted after the final Cut-Off Date, and a zero mark will be recorded in such case. Students may apply for extensions according to the procedure described in the Student Handbook.

Because the project is worth a substantial number of credits, the penalty can make a significant difference to your final degree mark. If you are near a borderline then this can make the difference between e.g. a first-class degree and an upper-second.