Product: My Product
Team: My Group

Abstract

The abstract should consist of one sentence describing the intended functionality of your system, followed by a few sentences (100–200 words) summarising the key advances made for this demo. This should give the reader a clear expectation of what will be demonstrated. This report will not be read by the experts that will judge you during demo days.

Introduction

This document provides a template for the SDP demo report. This template structures the report into sections, which you are required to use. You can change the subsection headings if you wish. In this template the text in each section will include an outline of what you should include in each section, along with some practical LaTeX examples (for example figures, tables, algorithms). Your document should be no longer than **3 pages** for the first demo, **4** for the second demo and **6** for the final demo. You can include a bibliography and an Appendix with no length restrictions but you won't have the guarantee that it will be read.

The logo, group number, product name and team name should be updated accordingly in the template. You should also update the demo number header in the template (see line 11 in the source).

You should delete this introduction section (no introduction is required).

1. Project management update

The following items are expected in this section

1. This section should start with your goals:
    - Use a bullet point list to indicate each of the goals you had set for this demo, appended with "achieved", "partly achieved" "not achieved".
2. Concisely summarise the reasons for any deviations from achievement of your intended goals.
3. Provide a description of how your group organised the work towards the goals. Highlight any methods used to ensure effective group work such as protocols for code integration, task tracking, automated testing, etc. Copy and paste from your own previous reports is authorised but should be clearly indicated using text colours.
4. Provide an overview of which teams members worked on which aspects, and account for the total number of hours spent by each team member on the project so far.
5. Provide a summary of how the budget allocated to your team (both technician time and money) has been spent so far.
6. Provide a clear statement of any modification (relative to your original plan) that you wish to make to your goals for the next demonstration, with the appropriate justification.

Demo 3 instructions

For the final report, this section should follow the same structure, but provide a synthesis of the whole project rather than just focusing on what happened since demo 2.

Item 1 should only list partially achieved / unachieved tasks in the project.

Item 6 should be replaced by a concise post-mortem on what went well and wrong during the project.

2. Quantitative analysis and testing

This Section provides a clear summary of the tests designed and run to prove the reliability of the code / system as a whole, with details allowed in the appendix.

This section should first outline any testing methods you used (e.g. repeated runs of subsystems, data-logging, naive user testing).

It should then present relevant quantitative results. If you are using graphs, please make sure they are properly labelled and logically illustrate the point you want to make (e.g. to compare two algorithms).

For each test, the following points should be answered:

- Relevance: How is the test relevant to the system (what does the outcome demonstrate)? How does the outcome of the test allows to validate a milestone / affects the project plan ?
- Interpretation: What do the actual results say about the system and why (eg. does a 72% success rate on a classification algorithm mean that the system is robust or on the contrary that it is not)?
- Robustness: is the test exhaustive / are all edge cases considered?

At an absolute minimum, this section should provide a table (for instance, table 1, using the `table` environment) of success rates for repeated runs of the whole system.

Test	Time(mins)	Errors	Success
1	1:30	0	√
2	3:00	2	×
3	2:20	1	√
4	1:50	1	×
5	2:10	0	√

Table 1. Results for 5 tests of the system.

If you need a figure or table to stretch across two columns use the `figure*` or `table*` environment instead of the `figure` or `table` environment. Use the `subfigure` environment if you want to include multiple graphics in a single figure.

Demo 3 instructions

In your final report all the updated relevant tests that were performed during the project should be presented and accordingly commented. The final version of this section should convince the reader of the validity of the decisions made and the reliability of the system.

3. Budget

Each report should contain an actualization of the estimated total budget of your system.

4. Miscellaneous

This optional section includes any description of work that seems of relevance to the team. It can be the result of a user study, a commercialisation analysis, the description of a clever algorithm / specific solution to a problem etc.

This section should remain short and be completed if necessary with an appendix.

Pictures should be used if appropriate (for instance, figure 1), using the `\includegraphics` environment to include an image (pdf, png, or jpg formats), ideally with informative labels added.

To keep your folders clean, it is often a good idea to keep your images in a separate folder. In this example, we've put the figures in the `figs/` folder. To include images from different folders, give the relative path from this file. Example: `\includegraphics{figs/image_filename}`.

If you present algorithms, you can use the `algorithm` and `algorithmic` environments to format pseudocode (for instance, Algorithm 1). These require the corresponding style files, `algorithm.sty` and `algorithmic.sty` which are supplied with this package.

You can also include references to scientific papers (Langley, 2000) or studies that will help justifying the value / interest of your system or validate a specific design choice.

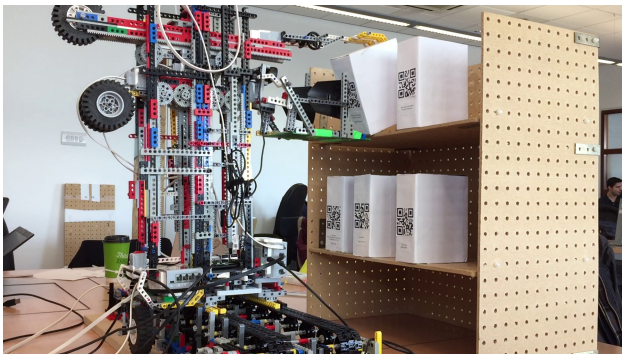

Figure 1. Lego construction: highlight any salient features in the caption

Algorithm 1 Bubble Sort

```
Input: data x_i, size m
repeat
  Initialize noChange = true.
  for i = 1 to m − 1 do
    if x_i > x_{i+1} then
      Swap x_i and x_{i+1}
      noChange = false
    end if
  end for
until noChange is true
```

Submission

This section is to be deleted.

The document should be submitted on Learn by one group member. The filename must be `group-[g]-demoX.pdf` where `[g]` is the group number and again `X` is the demo number. This document should be submitted by a group member nominated for this purpose, and also emailed to the group mentor at the time of submission.

A. Appendix

The optional Appendix should complement the report. An updated Gantt Chart might be relevant. It could also include a developed analysis of a conducted user study or the details of a nice algorithm.

References

Langley, P. Crafting papers on machine learning. In Langley, Pat (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.