INF-xxxx: course name

Assignment x

Your names

UiT id: abc123@uit.no & def456@uit.no

GitHub users: username1 & username2

GitHub Classroom: team name

October 17, 2022

1 How to use this template

This template is meant as a general guideline on how to write a report, and to give some tips about what you should and should not be writing. You may find yourself moving or cutting sections depending on the assignment or your own needs.

Also, make sure to clean up the template text before submitting your report.

2 Introduction

This section should be brief. Describe the assignment and the requirements in your own words. Avoid listing the requirements directly.

Here are some examples on how to start your introduction:

1. This report describes the design and implementation of a list ADT using a linked list. It will go into detail about the design choices made, and discuss benefits and tradeoffs of those choices.
2. Boids is a computer model created by Craig Reynolds that simulates the flocking behavior of birds [1]. In this report, we present an implementation of the model using the Python programming language.
3. SQL is a widely used querying language used to process queries into table-based databases. This text details the implementation of a simplified server that implements a subset of the SQL language built over sqlite.

3 Methods

3.1 Design

This is where you describe how you solved the assignment, at least on paper. Give a high-level view of your design. As a rule of thumb, if you are describing your code, you need to go to a higher abstraction level.

This section is also a good place to put illustrations to enhance the text. There are multiple tools out there to create good illustrations. draw.io is a strong tool that can be run in a browser. For more advanced users there are stronger, free tools such as Yed Graph Editor.

While illustrations are good at making your report clearer and look more polished, avoid using them to fill up space if you can convey the same information clearly using just text.

Examples of what the design section should cover:

- The interface of the list ADT supports six methods. These are create_list(), destroy_list(), add_list(), remove_list(), iterate_list() and sort_list(). When a list is created, it is provided with a comparator method that is used to handle sorting...
- The Boids simulation consists of a set of entities called Boids, Each boid moves independently according to a set of criteria, specified in three rules. Firstly, boids avoid crashing into obstacles, including other boids. Secondly, boids attempt to maintain the same speed and heading as nearby boids. Finally, all boids attempt to move closer to each other to form a cohesive flock.
- The server parses incoming data requests into an SQL query and runs them on its database. The result is then processed into JSON and returned to the client.

Remember to avoid low-level details! An expert should in theory be able to implement your design in any programming language based on what you write in this section.

3.2 Implementation

This is where you go into detail about your specific implementation. Questions you should answer here are things such as "How does your implementation match your design?" and "Are there any bugs, and do you have any ideas about what may be causing them?". What sort of difficulties did you experience when working, and how did you overcome them? If you found a clever solution to the problem, this is also the place to write about that.

3.2.1 Technical Details

You may want to include a short section giving high-level details about your implementation, such as the programming language used and other information you find relevant for your report. In most cases however, this section is unnecessary, as the assignment usually decides those details for you. Even if you have the freedom of choice, consider whether this information is really relevant to the report, which should avoid low-level implementation details most of the time.

3.3 Experiments

A core pillar of computer science is testing. In this section, you should include your methodology for testing your implementation. How do you know your implementation meets the requirements? What sort of performance metrics have you chosen to benchmark your solution, and how did you go about performing tests to gather those metrics?

4 Results

You should present the results of your tests here, either using an illustration and/or a table of results. These will be valuable in the discussion section. The following is an example for how to format a table of results in LaTeX. Notice how you can customize the table formatting with more hlines and the contents of the row formatting argument to the tabular section.

Test	run 1	run 2	run 3	average	std deviation
Join	4.789s	4.424s	4.274s	4.495s	0.294s
Leave	1.904s	1.826s	1.825s	1.852s	0.052s

Table 1: Accumulated join and leave times

If presenting data and benchmarks are not important to the assignment, you may likely find yourself cutting this section. If you decide to do so, make sure you at least mention your correctness criteria and testing methodology somewhere else.

5 Discussion

The discussion section is the most important section in a report, This is where you show that you understand the theory behind the solution, and also a chance to argue the pros and cons of your solution. You should discuss about the results of your measurements and why you think they are the way they are. Also bring up tradeoffs, and why you made the choices you did; show that you understand the alternatives, and why they may be a good idea (or not) for the particular problem the assignment asked you to solve.

Here is an example of a discussion subsection:

5.1 Recovery of a simulated crash

When a node recovers from a simulated crash, it will check if it's neighbors is still connected to it. If not, it will try and start an internal join to its previous successor. This works as long as the previous successor is still active in the network. The case where the previous successor is not active, is not dealt with, and will result in the node not being able to recover.

5.2 Conclusion

Here you sum up the report and reiterate the results. Does not need to be very long, a few sentences is fine.

6 Sources

Make sure to list your sources! We don't require a strict schema for sourcing, but remember to be consistent and correct with whatever method you choose.

This is an example of an inlined bibliography. This is a simple way to cite sources but not optimal for large projects. If you need more advanced citation management look into bibtex.

Note that in this case, section *Sources* and section *References* are the same section.

References

[1] Tanenbaum, Andrew S. Modern Operating Systems, 4th edition. Pearson, 2015. Chapters 3.3-3.4.

[2] OS dev contributors. (31 August 2018). Paging. In OSDev wiki. Retrieved 22:24, April 27th, 2019, from `https://wiki.osdev.org/Paging`

[3] Wikipedia contributors. (2019, March 7). Page replacement algorithm. In Wikipedia, The Free Encyclopedia. Retrieved 18:45, April 29, 2019, from `https://en.wikipedia.org/w/index.php?title=Page_replacement_algorithm&oldid=886600514`