Group C Members:

Rokus Jankunas - 180017115

Momchil Badzhev - 180012872

Mark Capewell - 180010926

Words count (inc. headings) – 558

## **Our Approach**

During our first meeting after the groups were released, we thoroughly discussed the assignment brief. Following this, we then discussed the requirements in detail and distributed them between team members, with emphases on our individual strengths and weaknesses. Before beginning work on the project, we carried out some research to further understand what an automaton was and how it worked. Consequently, this allowed us to come up with some ideas on how to handle this assignment. After this, we had regular meetings in person to discuss progress on the project, while utilizing GitHub for version control and Trello to keep track of tasks and their current status. Trello is somewhat similar to agile methodology, which is why it is widely used in the IT industry.

## **Problems faced and how we overcame**

Whilst working on this project, we came across some difficulties. Below are the problems we faced and how we overcome that issue.

1. **Design**

The first decision we had to make was weather to use C or C++ for this assignment. From the start we knew that arrays would be most fitting for this project. After a lengthy discussion we settled on using C++ together with the vector library. This would allow us to utilize the functionality of arrays, while having a flexible number of elements.

1. **C++ Syntax**

Two out of the three members of our group had practically no experience with the C++ language. In the beginning it took a while for us to get a hand on the features we were utilising, that happen to be different to other programming languages we had knowledge of, or were completely new such as vectors. By reading the documentation available online and with the help of lab tutors we managed to quickly understand the nuances of C++.

1. ***.size()* on vectors**

While debugging our code late into the assignment we discovered that *size()* function when applied on vectors returns *size\_t*, rather than an *int.* This caused the code not to compile properly when all warnings were treated as errors. The issue occurs because of signed and unsigned integers. To solve this issue we simply began using *size\_t*, where appropriate, making the code compile without errors.

1. **2D Automaton**

## Our solution to the “different 2D automaton" task works by applying the ruleset over 2D generation from the row above, similar to how out 1D implementation works. This consequently allows to wrap top and bottom to a 3D torus as mentioned in the task description. We found that in the task description wasn't clear what this "different" approach should be, so that is why we settled on this implementation.

## **Reviewing**

Overall, this was a very successful assignment. Team communication was good, both in person and through social media. Team members were always open to meeting and discussing the assignment, even when this would require them to change up existing plans. Commitment level was high from the whole team. Coordinating was made easy with the use of GitHub and Trello. Members were very enthusiastic about using them, making getting used to this new workflow straight forward. Whenever issues were encountered, the team came to each other’s aid.

## **References**

<https://www.geeksforgeeks.org/vector-in-cpp-stl/>

<https://en.cppreference.com/w/cpp/container/vector>

<http://www.cplusplus.com/doc/tutorial/files/>

<http://www.cplusplus.com/doc/tutorial/basic_io/>

Also received help from lab tutors.