NEA Project Evaluation

Jacob Halleron
The Sandon School

Reflection

The final outcome of my project meets all of my original requirements while still being generally effective, understandable for the end user and arguably stylish for what it is. The project is easy to run and operate, and competently demonstrates the workings of multiple algorithms.

I will repeat the objectives from the analysis document:

> 1. *The product must effectively teach students how the algorithms work. This one's fairly vague but the others should provide some detail.*

I've informally shown the program to others, and they find it to effectively show multiple aspects of the algorithms.

> 2. *It should be able to run on almost all computers. The client's system is very above average, and isn't a typical setup the average student would have, but slowed-down bubble sort shouldn't be too resource-intensive.*

The user accesses the program using a web browser, which is available on almost any device. The program can be run on any computer running Linux or another Unix-based OS, but the server is of course not necessarily the same machine as the client.

> 3. *It must work on most operating systems. Windows is the main one, but web browsers tend to be the same on all platforms. I'll test it on Chromium, Safari and Firefox since almost everyone uses them or a derivative. Since it's made for the web, I'll also need to think about server-side stuff.*

As stated, I tested the site on Chromium, Safari and Firefox, and it works the same on all 3.

> 4. *The UI must be simple and intuitive. It's aimed at minimum 14 year olds, who might not have advanced technical expertise. I think my CSS skills are good enough; my aim is a grid layout with some fairly plain-looking grey boxes and a very colourful diagram so that the focus is on the animation rather than unimportant details.*

The aforementioned other people I informally showed the program understood the UI and easily controlled it.

> 5. *There must be a feature to compare multiple animations side-by-side in parallel. I'm not yet sure if the code blocks will still be visible, or if the controls will be universal for every animation or specific to each.*

I implemented this as a separate web page. The code blocks are not visible in this mode. Some controls are universal and some are specific to each animation.

> 6. *A code block must be available beside the animation, ideally with the current step highlighted. This will aid the learner. For readability, I'll probably write it in Python or pseudocode so it's easier to read so it will be separate from the actual internal JS.*

This was programmed into the project; pseudocode is shown and the code shown is not the same as the code actually used in the scripts.

> 7. *The speed of the animation will be adjustable. I'll put a separate control box below the diagram, which will contain a play/pause button, next/previous buttons, an input box for the tick rate, and a reset button.*

In the final product, there are controls at the top of the page, specifically play/pause/reset buttons, speed control, and input size control.

> 8. *This isn't based on his answers, but I'd like there to be some data on each algorithm, e.g. the big O and/or efficiency with different given inputs.*

The big O complexity is shown next to the algorithm name. There is no data on efficiency.

Independent Feedback

Here is some brief feedback from an independent third party:

> *The app produced is easy to use and also very educational. I tried it with many combinations of program. Good work!*

That's some very positive feedback.

Improvement

If the project were revisited, it's clear it can be expanded upon to demonstrate more algorithms and types of algorithm. I didn't have time to visualise trees or graphs, so that could be done. The core programming, i.e. the UI and main classes are satisfactory to my standards.