COMPENG 2SH4 Project – Statement of Contribution

Your Group Name IBio Dream Team

Your Name Mohammed Al-Hindi

Your Team Member's Name Stefan Candeloro

You must complete this statement of contribution without discussing it with your project partner, i.e., individually. Your statement should be concise (at most one-and-a-half page). It has three parts:

1. Tell us about your own contribution to the development of your COMPENG 2SH4 project. For example, you can tell us about which project iterations (as mentioned in the project manual) and C++ project classes that you worked on and completed. You can provide a concise answer either in paragraph form or through bullet points.

As Developer B, I was responsible for iteration 1B I created the gameMechs class methods, project pointer, and all private data members (***exitFlag, loseFlag, boardSizeX/Y, getInput(),*** and ***getScore()***). I was also responsible for iteration 2B where I made the decision of making a separate Food class for ease of additional features implementation. Iteration 2B involved prototyping and writing the ***generateFood()*** and ***getFoodPos()*** methods which allowed us to generate a single food item at random positions within the board, not overlapping with the snake. In iteration 3, I worked on transitioning our snake from an **objPos** data type to **objPosArrayList** in our player class as well as updating the relevant method functions. This also involved updating our ***generateFood()*** method to accept the **objPosArrayList** reference as the input to make sure no food item overlaps with any element of the snake's body/array. For our Above and Beyond segment, I modified the food class to create a food bucket that prints five food items (2 special ones, 3 normal ones) instead. Normal food items 'o' simply increase the length and score by 1, while the special ones '+' increase the length by 3!

2. Repeat Part 1 above but this time tell us about your project partner's contribution to the development of your COMPENG 2SH4 project.

Stefan was responsible for iteration 1A where he implemented the Player class including the main ***updatePlayerDir()*** and ***movePlayer()*** methods. These are important as they require parameters that can only be accessed through my **gameMechs** class like the gameboard size, which means he also created the object pointer to the **gameMechs** class. After his work on this iteration, we were able to control our player object's movement. Stefan was also responsible for iteration 2A where he implemented the **objPosArrayList** class, necessary for our ***generateFood()*** that I was responsible of writing. He created the snake movement mechanism where the head prints and tail is deleted within the ***movePlayer()*** method along with all it's test cases. This enabled us to make our special food items "special" by making the + food items that increase the length of our snake. Lastly, he helped check the snake's death in-case of collision with itself, printing the exit message.

3. Tell us about your experience in your first collaborated software development through this project – what was working and what wasn't. If you are a one-person team, tell us what you think may work better if you had a second collaborator working with you.

My experience throughout this project has been a rewarding and pleasant one as I was able to put my knowledge from lectures towards developing a working snake game. The project was designed in a way to allow for individual work to be done before needing to connect back with the team and implement all elements together which I found very practical. Using GitHub, however, required some time from both Stefan and I to learn how to merge commits and work on the same files simultaneously. When we both write code at the same time and decide to push our changes to the repository, the person pushing second requires to first pull, merge the code, and then push their new version. We found that this might not work as having to merge changes every time would be very time consuming and thus, we actively communicated and dedicated working hours to organize our work. Once we figured it out, we were able to meet all milestones and iterations without many problems. While coding for long periods of time might cause burnout, having a partner that could take over at certain points was very helpful especially when you need a second perspective on certain design and logic decisions. I recall specifically how I was stuck implementing the main logic for our generateFood() function and Stefan proposed the idea of utilizing the validation flag logic from PPA3, saving us a lot of time in developing our algorithm.