Probably everybody’s passion for computers originated from playing video games, and I am no exception. I was crazy about a shooting game called “Counter-Strike” when I was in middle-school. Apart from improving my skills of playing the game, I did plenty of research on internet to change some source code in the game such as the initial money and gravity index which intrigue my first touch of codes. Sampling changing indexes was not programming, but it inspired me to explore more in this magical field.

My passion for programming peaked in high school. As an inquisitive teenager, I was constantly seeking more challenges and began to learn Java by myself by reading a book called “Introduction to Java Programming” by Y. Daniel Liang. Later, when I was learning Matrix-Multiplication in math, I coded an algorithm using a triple-loop and two-dimensional array. During the debug process, I noted that some calculations are duplicated which intrigues my consideration of efficiency of algorithms. I enjoyed thinking high-efficiency algorithms to make my mind more active by making comparison with old ones and I always could learn a new way of programming to save time. When I read Strassen algorithm in ‘Introduction to Algorithms’ , I found out that divide-and-conquer strategy could make the recursion tree smaller. The Big O notation in that book allows a further development of my understanding of efficiency of algorithm. I am able to digitally imitate the speed of algorithm which can helps me to determine which algorithm should be used. The classic game Tower of Hanoi gives me the idea of recursion method. I found that the recursion method is an ideal means to solve some problem such as factorial but it also sometimes complicate the calculate when solving Fibonacci numbers which may lead to a huge recursion tree. In this case, record all the numbers in an array might be a better idea.

Computer Science never stops surprising me. In a summer vacation, I made a personalised TicTacToe AI for my cousin. Then I came across the minimax algorithm. I found that actually the essence of minimax algorithm is search algorithm just like the Knapsack problem. It utilize the high performance of the computer to purposefully solve a part or all of the possible situations to determine the best solution. After deeper understanding search algorithm, DFS and BFS drew my interest. DFS is the algorithm that first select a possible situation to explore forward (child nodes), and in the process of exploration, once the original selection is found to be inconsistent with the requirements, go back to the father node and re-select another node. The algorithm continue to explore forward, repeat this until find the optimal solution. The BFS is quite the opposite. It seems like we cannot simply judge whether DFS or BFS is better. However for a board game, it is not possible for computer to calculate all possibilities in a short time, so a ‘depth’ is added in the minimax algorithm which limits the number of layers of searches. The ‘depth’ is a product of experience and is not scientifically sound. But with the development of machine learning, AI should able to adjust the value of ‘depth’ by play game with itself. The search algorithm drives me to seek more in Computer Science.

Furthermore, extra-curricular activities have strengthened my learning skills. In a summer program organized by a top UK University in 2018, I learned Python language. Academically, I also received a Global Silver Award in the British Physics Olympiad, the Gold Award in the Rising Star Chemistry Challenge (top three percent) and a high distinction and credit in Australian’s Chemistry and Math Competitions.

With the knowledge I had for Computer Science, I hope to be immersed in computer science at a higher level, exploring the area that I am interested in and good at. I believe the opportunity to study in one of the most outstanding institutions will guide me in my exploration and construct new knowledges to me.