The first time I wanted to dig into the world of Computer Science was a year ago when I encountered my favourite anime, which showed me how splendid a game could be with advanced technologies: background stories are automatically generated by AI which has independent thinking abilities. Delving deeper into it, CS became more and more charming, particularly those spectacular AI, VR, etc., offering the possibility to break the limit of the material world and provide players with an immersive game environment.

Last year, I attended a summer school at the University of Oxford, which was my first time to attend a CS course, and programming attracted me. It was like communicating with a friend who would give me expected responses, which was even more comfortable than interacting with a human. We had several tasks, involving designing websites and implementing an encryption and decryption program. While the tutor only taught us fundamental programming skills like selection statements and loops and I had no basis of programming, I had to learn coding from websites tutors recommended, from which I gradually developed my self-learning ability and I confirmed that I can learn any other programming language fast as they are different in syntax but share the similar theories. To accomplish the program, I searched online for functions I needed and came up with several plans. At first, I considered writing a dictionary. However, it was impossible to cover the characters of all existing languages. As I learned more, I then knew that all characters are uniquely encoded as numbers using Unicode. Hence, I decided to encode strings into numbers, which can be encrypted with mathematical functions. I used various functions for characters of different positions in a string to increase the complexity of my encryption and the decryption processes used corresponding inverse functions.

After the summer school, I still immersed in the ocean of encryption and wanted to find methods to enhance encryption security further, so I learned another two algorithms. One is the XOR encryption, a relatively simple symmetric encryption, and the other is a widely used algorithm: RSA. The theory of the XOR encryption was easy to understand: each binary digit of the numerical encoding of the plaintext is applied with the key using XOR operation. When the receiver decrypts the cipher, they use the same method and key to get the plaintext as the operation is reversible. This is convenient but also vulnerable, for example, if the plaintext has sequences of repeated characters. On the other hand, potential danger in key distribution increases the vulnerability of the XOR encryption. The RSA algorithm is more robust because breaking its key requires large number prime factorisation. As this is an NP-hard problem, it is not solvable by any practical algorithm whose time complexity does not grow exponentially. More specifically, when the key is short, a simple algorithm is adequate, whereas with a key of length more than 1024 bits, breaking the key cannot be done within a reasonably short time. Therefore, the RSA encryption’s security is guaranteed unless the P=NP question is solved or a usable quantum computer is invented. Besides, the digital signature is also widely used using the RSA algorithm. Meanwhile, it is an asymmetric encryption, which avoids danger in key distribution as private key should be kept secret entirely while only public key is published for encryption or signature verification. I learned in my research that security can be improved by increasing time consumed to decrypt through specific maths theorems and this experience stimulated me to learn more fascinating algorithms.

As I discovered I was short of knowledge about algorithms, I read a book named Aha Algorithm. After reading, I realised that Apart from the feasibility of algorithms, their time complexity is also a vital consideration not only for encryption but all kinds of algorithms. Algorithms originated from our life rather than profound concepts I used to consider; Basic events and details that we naturally neglect are arranged systematically and turned into well-regulated approaches to address problems efficiently, which fascinated me further.

In 2019, I participated in the Euclid Contest and STEP 1 examination (grade 1), harvesting quite good results. While I was preparing, I acquired extra mathematical knowledge beyond the A-level syllabus and I found some of them were related to CS such as number theory, which allowed me to understand RSA algorithm easier. Also, the strict logic link between sub-questions helped me learn algorithms as they both need every step to be correct otherwise the solution would be wrong. I also attended the BPhO competition, in which I practiced logical thinking and problem-solving ability for science questions.

Gradually knowing more about complicated theories hidden behind seemingly simple computer operations, I feel eager and become more excited of expanding knowledge in university and hope to be a game developer in the future.