Practical Computer Concepts

Dr. Nadim Alkharouf

AS.410.634.81.SU24

Alex Arvanitis

Matthew Montalvo

Christine Whittle

Final Group Project Report

The assignment for this project was to write a program that finds all open reading frames (ORFs) from an input file with any number of FASTA-formatted sequences. The workload was divided into three parts based on the number of individuals in this team, and with the goal of equalizing the time input from each group member. Group member 1 (Christine Whittle) was charged with working on the input handling and parsing. Input handling primarily included opening and reading the file, stripping whitespaces and special characters, case conversion, and placing cleaned input into a dictionary of headers and sequences for further processing. As this task involved less coding work, she was also tasked with writing up this report.

Group member 2 (Alex Arvanitis) was tasked with creating functions to find ORFs from the aforementioned dictionary of sequences. This included verifying the minimum length, converting input sequences to ‘framed’ sense and antisense strands, and scanning the resulting frames for start and stop codon pairs. This part contains two main functions; the first, make\_antisense, takes a sequence as an argument and returns the reverse complement of that sequence. The main function, orf\_finder first generates the sense and antisense frames by slicing the first nucleotide (or two) from the left end. It then scans the six frames, three nucleotides (codons) at a time, until it detects the start codon. It then begins scanning for one of three valid stop codons, collecting position, length, frame number, and DNA sequence data of the detected ORF. This data is stored in a dictionary, which is returned at the end of the function call.

Group member 3 (Matthew Montalvo) was asked to handle final integration and output formatting. Integration involved creating and using variables across functions to ensure that each subsequent function received the correct data from the previous. Formatting the output included introducing spaces between codons, wrapping lines after fifteen codons, and ensuring the correct information was placed in the headers. To accomplish this set of tasks, multiple functions were defined. The first function formatted the desired output of the codons into triplets and set the limit for only fifteen codons to be displayed on each line. The second function created a new dictionary to sort through the created sequences and print the keys and values. The third function determined the parameters of the final formatting output. To unify and run all the functions created by the group, a fourth and final main function was created. This main function prompts the user for an input file, integrates the input from each group members code using a new dictionary, and removes open reading frames that are not the correct length.

One of the obstacles that arose during the project was a frame reading shift in the detection of the ORFs. The team was trusted to troubleshoot the issue as soon as it arose, and it was resolved and tested within the same day. All members took the initiative to troubleshoot and check the codes developed by the other members of the team. As all parts of the code were integrated, another challenge arose in creating output that did not include a “None” result. Team members worked together to create two alternative options that eliminated this outcome, and the two options were blended to obtain the final result. The communication of the team was regular and effective, which reduced the challenges faced for troubleshooting.

By solving this coding problem as a team, multiple lessons were learned. Each individual approached their portion of the code independently. In the future, if more discussions are had up front about how each portion of the code is being approached, overall integration will run smoother. This is a valuable lesson to take forward in a team approach to coding. While the code we developed does not detect nested open reading frames, it may be possible to create functions that solve this part of the problem in the future. Tackling this coding problem as a group allowed multiple perspectives to see what is missing from the solutions and where different parts of the code could be improved.