**BIOL5381 Biological Computing in Python - Assessment** **2021/22**

The assessment for this course has two components:

1. A coding problem, worth 80% of the course grade
2. A report, worth 20% of the course grade

**Coding Problem - 80%**

Your task is to write a Python script that does the following:

1. Take in a VCF file, a GFF file, and a genome fasta file using a command line interface
2. Iterate through the VCF file. For each variant in the VCF file, determine if the value in the QUAL field is greater than 20. The count of variants where QUAL <= 20 should be reported in a log file
3. For variants where the quality is greater than 20:
   1. Determine if the variant is in a coding region.
   2. If the variant is in a coding region, determine if it results in an amino acid change.
4. The script should output the following:
   1. A log file containing:
      1. Confirmation of the filenames given at the command line.
      2. The count of the variants where QUAL <= 20.
      3. If the script runs correctly, the location of the output files.
      4. If the script did not run correctly, errors should be reported in the log file.
   2. A bar plot showing the proportion of the variants with QUAL > 20 that meet the following criteria:
      1. Non-coding (not in a coding region).
      2. Synonymous (in a coding region, but does not result in an amino acid change).
      3. Non-synonymous (in a coding region, and results in an amino acid change).
   3. A tab-separated table with one row for each variant with QUAL > 20, with columns as in the example below (values in this table are examples only).
      1. Chrom, Pos, Ref and Alt columns are taken from the VCF file.
      2. Type should be “Non-coding”, “Synonymous” or “Non-synonymous” as determined above.
      3. Transcript contains the id of the transcript in which the variant is located if that variant is in a coding region. The value ‘NA’ should be used for non-coding variants. See FAQ below for more information.
      4. Protein Location contains the coordinate of the amino acid at the location of the variant. The value ‘NA’ should be used for non-coding variants.
      5. Ref AA contains the amino acid encoded by the reference sequence at the location of the variant. The value ‘NA’ should be used for non-coding variants.
      6. Alt AA contains the amino acid encoded by the alt variant. The value ‘NA’ should be used for non-coding or synonymous variants.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Chrom** | **Pos** | **Ref** | **Alt** | **Type** | **Transcript** | **Protein Location** | **Ref AA** | **Alt AA** |
| 1 | 3654 | A | T | Non-coding | NA | NA | NA | NA |
| 3 | 7839 | G | A | Synonymous | transcript\_id | 24 | H | NA |
| 5 | 99765 | C | A | Non-synonymous | transcript\_id | 146 | V | L |

You should write “good code” in the sense of what was discussed in the BIOL4292 Programming and Databases course. Your code should be correct, efficient and maintainable with descriptive variables, consistent formatting, good spacing and useful comments.

Additionally, your code should catch any errors and handle them appropriately.

It is possible to implement this code using the libraries we have explored in this course and base Python, however if you have a preference for other libraries, you are free to use them.

Please remember the plagiarism rules - you cannot look at anybody else's code or show your own code to anybody else. You also cannot cut and paste code that you find externally (e.g. on websites) or directly use model code from lab sessions. Your submission will be checked for similarities with other code.

**Report - 20%**

The report should be a maximum of 1000 words and should contain documentation for your script. The documentation should contain the following:

1. A description of how to run your script.
2. A description of the input files required and their formats.
3. A brief written description of how your script works
4. A description of the output files and an explanation of their contents.
5. A description of any problems you’ve identified in the script ,with suggestions of how these could be overcome. See FAQ below for more information.

**FAQ**

1. What if my variant is in a gene, but not within a coding region (e.g., the variant is in an intron or UTR)?
   1. For the purposes of this assessment, I am only asking you to consider whether or not a variant is in a coding (CDS) region. There is no requirement to report a transcript id for variants that are in UTR or intron regions.
2. What if my variant is in a CDS that is used by more than one transcript?
   1. In this case, you should write one row in your table for each transcript.
3. In the report, what do you mean by “a description of any known problems”?
   1. This is an opportunity for you to discuss any aspects of your code that you were not able to implement to your satisfaction. Aspects to discuss include:
      1. What was the behaviour you were trying to achieve?
      2. What is the behaviour you actually implemented?
      3. What would you do differently to achieve your original aim given more time?
4. How long should my script be?
   1. There is no minimum or maximum length for your script. For guidance, you should be able to implement the required features in 100-200 lines of code. If you are writing much more than that, you should re-consider your approach.
5. Do I need to write classes or functions in my code?
   1. For this assement, the most important thing is that the code is functional. However, writing good code in the sense of what you have learnt in Ronan’s course will usually improve your grade. If you can factor something out to a class or a function, you should do this, but not at the expense of having something that works.