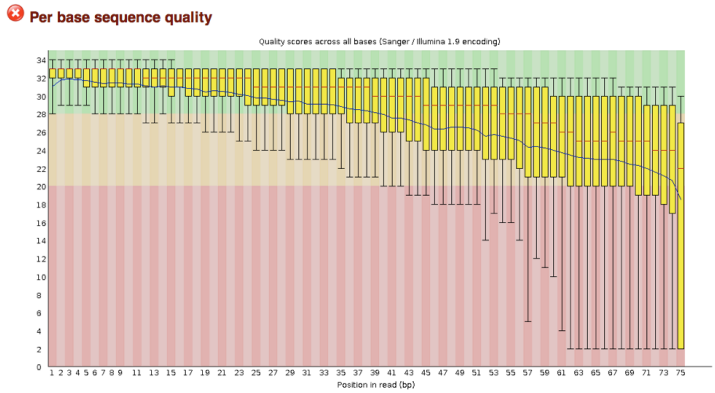
P3: Python programming (old exam)

Description

Next-generation sequencing machines produce vast amounts of DNA or RNA reads. Illumina's sequencers produce output in the form of FASTQ files. Quality control of the produced reads is a necessary step before any downstream analysis, such as assembly or mapping. Typically, the average quality at the 3' end of the reads is lower than at the 5' end of the read, caused by the chemistry and process of sequencing. When plotting the per-base quality for all reads in a FASTQ file, a typical pattern looks like this:



In this assignment you will calculate the average quality score at each position of the read, use a command line tool to trim off low-quality bases, and assess whether the average per-base quality has improved.

Assignment

Write a script that performs the following tasks:

1. Parse a FASTQ file. Translate the quality values (Illumina 1.5+ encoding) to a scale from 0 to 41. Use the built-in function **`ord()`** for the translation.
2. Calculate the length of the shortest sequence in the input FASTQ file, the longest sequence in the file, and calculate the average sequence length.
3. Calculate the average quality score (on a scale from 0 to 41) at each position of the read. In the raw FASTQ file all sequences have the same length. When calculating the average quality value at position 0, you average over the quality values at position 0 of all the reads, etc. Your script should be able to work on input sequences of any length (e.g. the tiny example below).
4. In your Python script, trim off low-quality bases using the program **`fastq_quality_trimmer`**. Set the quality threshold to 30, and specify that the data is on illumina 1.5 scale (-Q 64). Name the output file 'trimmed.fq'.
5. Calculate the average quality score at each position of the read in the trimmed file. At each position, calculate the improvement with respect to that of the original FASTQ file (step 3).
6. Report the minimum, maximum, average sequence length for both the original FASTQ file, and the trimmed FASTQ file.
7. For each read position, report the average quality score in the original file, the average quality score in the trimmed file, and the improvement in average quality, in tab-delimited columns.

Input

A FASTQ file containing 10000 records. It is a sample of genomic reads from a tomato plant. Use the command **`wget`** to download the file from this location:
http://www.bioinformatics.nl/courses/BIF-30806/docs/tomatosample.fq
For development purposes or if you fail to get step 4 working, a trimmed FASTQ file is also provided:
http://www.bioinformatics.nl/courses/BIF-30806/docs/trimmedsample.fq

Output
The output of your script should look like this: (Note that the numbers in this output are made up. The numbers in your output will be different!)

```
ORIGINAL: min=100, max=100, avg=100.00
TRIMMED: min=27, max=100, avg=96.48
1    33.18    33.18    0.00
2    33.42    33.42    0.00
3    33.11    33.11    0.00
...
99   28.22    32.90    4.67
100  27.82    33.59    5.76
```

A tiny example

Label	Original	Quality in original	Trimmed at t=30	Quality in trimmed
Seq1 Qual1	AGACA bbbee	34, 34, 34, 37, 37	AGACA bbbee	34, 34, 34, 37, 37
Seq2 Qual 2	CCCAA hhhg[	40, 40, 40, 39, 27	CCCA hhhg	40, 40, 40, 39
Seq3 Qual 3	ATAAT cccCB	35, 35, 35, 3, 2	ATA ccc	35, 35, 35
	Pos 1 avg	36.33		36.33
	Pos 4 avg	26.33		38.00

The tiny FASTQ file used for this example may be downloaded for development or validation purposes: **http://www.bioinformatics.nl/courses/BIF-30806/docs/tiny.fq**

Environment
You should work on the **altschul.bioinformatics.nl** server.
The program fastq_quality_trimmer is installed there. Try it by typing

```
fastq_quality_trimmer –h
```

on the command line. You should see information on the usage and options. Note that the –Q option to specify the offset of the quality scores is not shown in the manual. However, you need to set this to **–Q 64**.

Additional notes
- Put your full name and student number as a comment in your script and put your name in the file name of your script.
- You may use the slides and the code from your exercises from this week. You cannot use BioPython or comparable packages, you should write your own fastq parser. You may not directly copy code from the internet, but you may use it as inspiration.
- The FASTQ format and quality values are explained on: **http://en.wikipedia.org/wiki/FASTQ_format**
- Running **`fastq_quality_trimmer`** takes only a few seconds or so. But to avoid running it over and over again, make sure your code checks whether the trimmed file exists.
- Think about your code organization. Use subroutines.
- Document your code.
- Make sure you hand in a working script. If it is unfinished, you can leave the unfinished part in comments.

How to hand it in?
Make sure your **name** and **student number** are in your python script. **Turn in your script on BlackBoard (under P3)**.