**Medical Genomic – Exercise 1 (03.05.17)**

**Ben Wulf**

**Task 1**

If we look on the files, we could see that the headers are nearly identical. The only difference in the headers is the 1 or 2 in the end. This leads to the conclusion the these files together contains read pairs and read in file 1 is the read pair to the read in the same position in the other file.

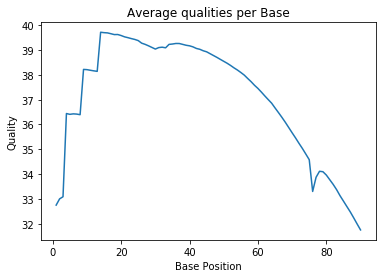
**Task 2**

The Phred-score is defined as .  
Hence Q =0 the error rate p is 1. In case of Q=42 the error rate is 0.0000630957. That means that the reported error rate is between 0.0000630957 and 1. With 1 is the worst possible  
case and 0.0000630957 is the best case.

**Task 3**

The results for Task3 are:

[32.7536463843,33.004507285,33.0898270127,36.4414169778,36.4095424106,36.4282075791,36.4190598276,36.3951808253,38.2181958115,38.2090927371,38.1839674502,38.1605638939,38.1399871704,39.7150631072,39.6980522644,39.6890381192,39.6558036985,39.620681442,39.6239694185,39.5825967637,39.5297231297,39.4963598464,39.4585221311,39.4251169321,39.3784899438,39.2755473471,39.2279142399,39.1682819795,39.103036199,39.0378359453,39.0974461149,39.1133130552,39.0883966388,39.2258624938,39.2407340065,39.2626497673,39.2611777609,39.2261544169,39.1910172208,39.168442208,39.1264673688,39.0627426616,39.0285283193,38.9691399773,38.9270102906,38.8520729164,38.7773793896,38.7032250322,38.6229109758,38.5442629985,38.4660662523,38.3774289521,38.280805504,38.1934953461,38.0944686121,37.9893185014,37.8540293608,37.7241941491,37.5785631547,37.4519659322,37.30588137,37.1520822624,37.0039938174,36.8594844933,36.6672582306,36.4804916317,36.2898115067,36.0902533918,35.8783414687,35.6659330708,35.4590751778,35.2417206544,35.0341302984,34.8099111557,34.5829449757,33.3035818114,33.8746533103,34.1153121942,34.0987129326,33.963673233,33.7757728565,33.5879020758,33.3764812463,33.1331784859,32.9163078341,32.6969283992,32.4802211621,32.2410692615,31.9950513255,31.7546435819]



The code is commented in the .py file

**Task 4**

For every sequence, the program counts the number of A, C, G, T and N using the count  
function. In this way, the performance is a little bit better then iterating over each string.

It is an overall average and like the quality for each position.

I decided to don’t count the N to the total number. Otherwise a N counts the same as an A or T and that is not correct, because it can be also an unclear G or C.

The average N is calculated by dividing all N by the number of total bases in the file.

Results:

average GC-content: 44.3516%

average N-content: 0.0036%

**Task 5**

More or less, this part is skipped. I wrote a function which counts the kmers but increases  
the runtime around 5 times.

**Task 6**

This task was a bit difficult. I perform the trimming one time for the paired reads. In the other case I assume that the daughter1 file contains single reads. Otherwise I would not be   
able to compare the trimmed reads with the original run.

Paired end summery by trimmomatic:  
Input Read Pairs: 14123580

Both Surviving: 14018200 (99,25%)

Forward Only Surviving: 77105 (0,55%)

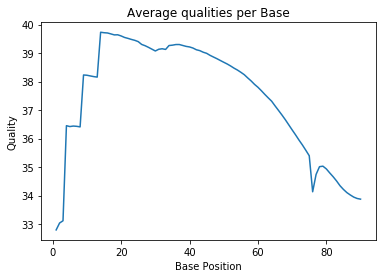
Reverse Only Surviving: 24698 (0,17%)

Dropped: 3577 (0,03%)

single end summery by trimmomatic:  
Input Reads: 14123580

Surviving: 14095305 (99,80%)

Dropped: 28275 (0,20%)



If we compare the average qualities per base form the untrimmed and the trimmed reads we see a small gain of qualities in the end (base position 80+) the rest seems to be unaffected.

average GC-content: 44.254792%  
average N-content: 0.000440%

The GC contend differs by 0.1% but the N-content is 10times smaller than without trimming.