Bioinformatics Lessons Schedule

- RNA-seq
- single cell RNA-seq
- RRBS

Date	Subject
12-24	Christmas break
12-31	Christmas break
01-07	Process RNA-seq
01-14	Process RNA-seq, continued
01-21	Process RNA-seq, continued
01-28	Analyze RNA-seq
01-28	Analyze RNA-seq, continued

RNA-seq

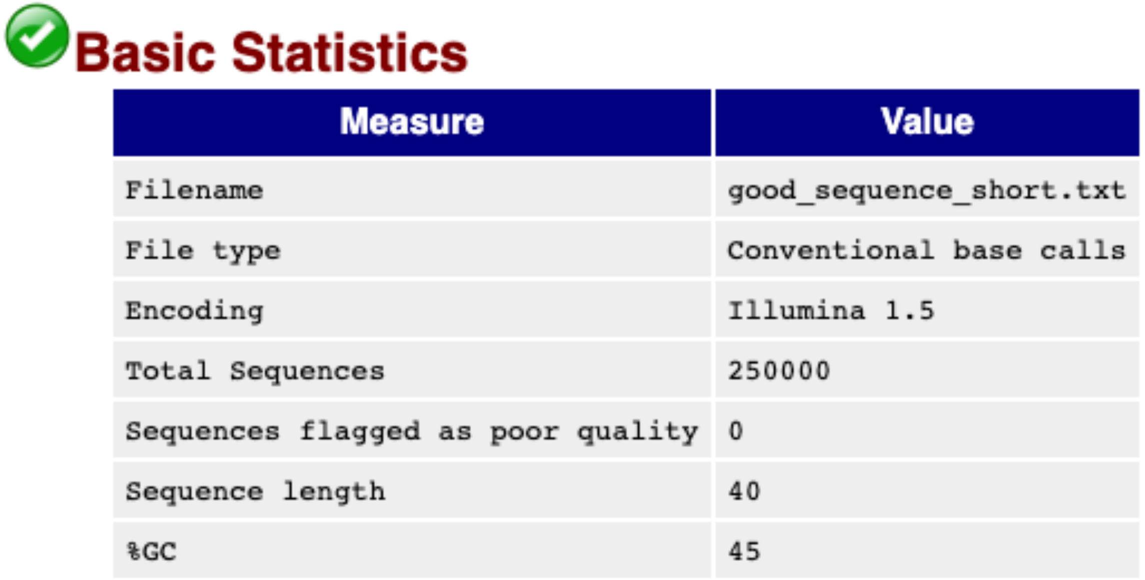
Quality Check

FastQC

- Before going forward, we want to check the quality of the data
  - How much did the sequencer fail?
  - Did we sequence mostly our sample DNA?
- FastQC is a program from the Babraham Institute in the UK that creates an html report on the quality of the sequencing data
  - Has 11 quality control checks that it does

Basic Statistics

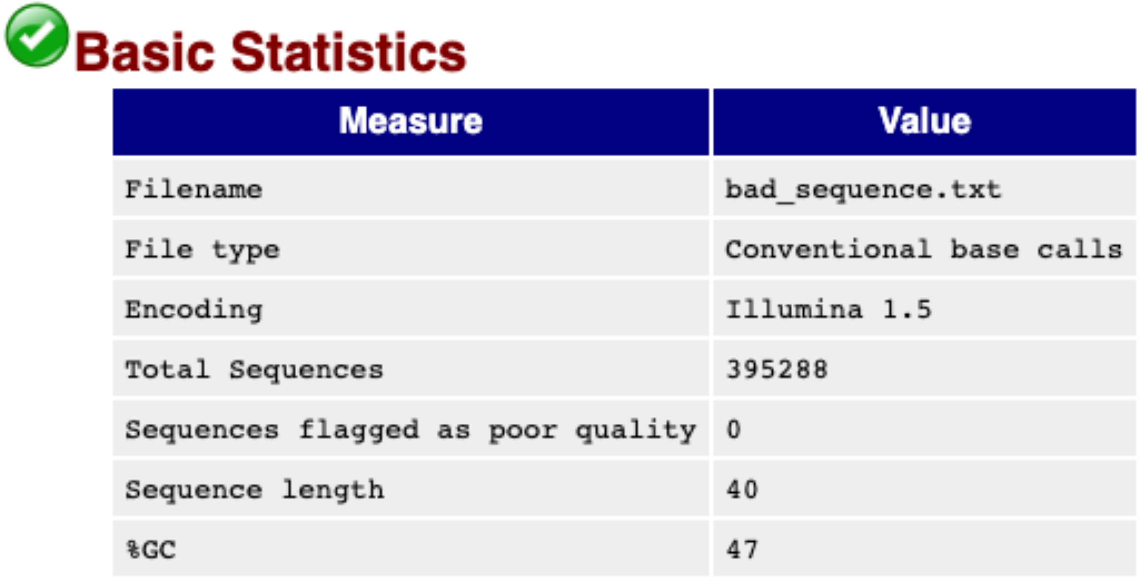
Good Quality



✓ Basic Statistics

Measure	Value
Filename	good_sequence_short.txt
File type	Conventional base calls
Encoding	Illumina 1.5
Total Sequences	250000
Sequences flagged as poor quality	0
Sequence length	40
%GC	45

Bad Quality

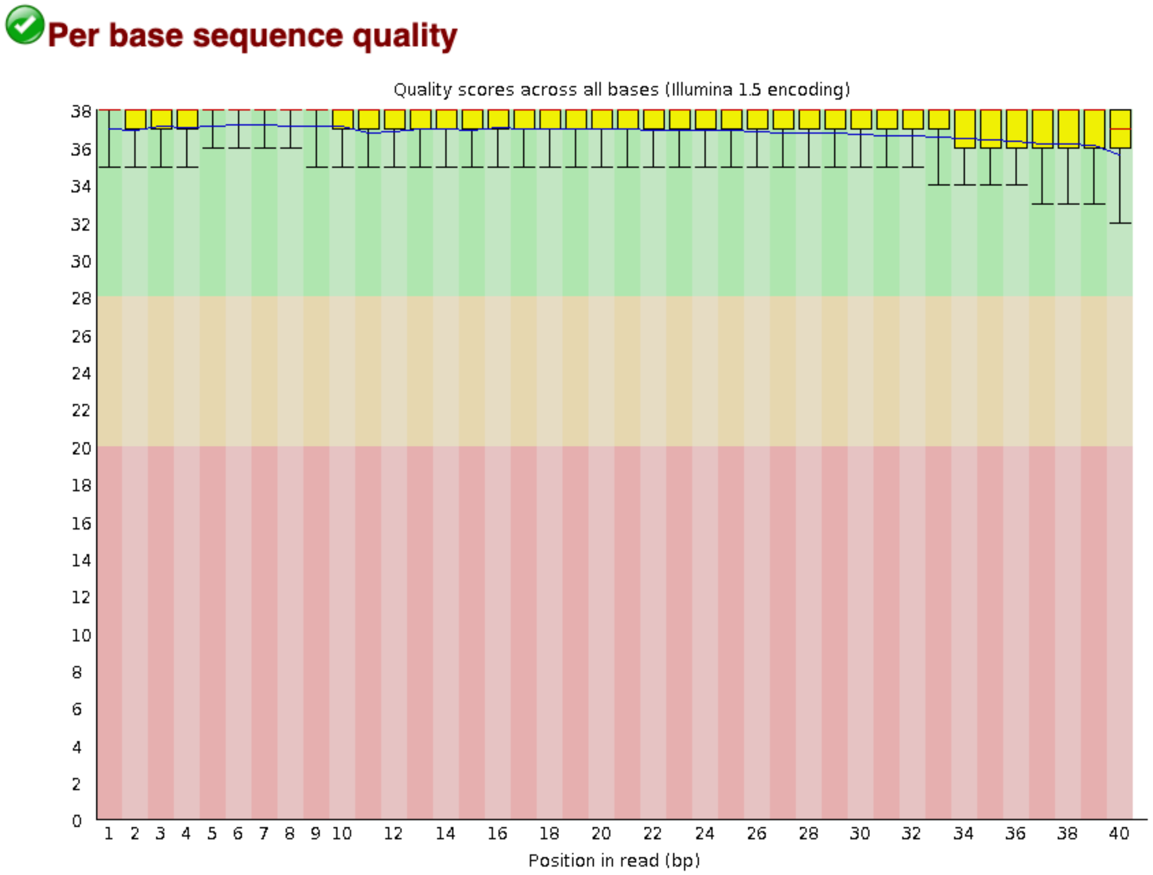


✓ Basic Statistics

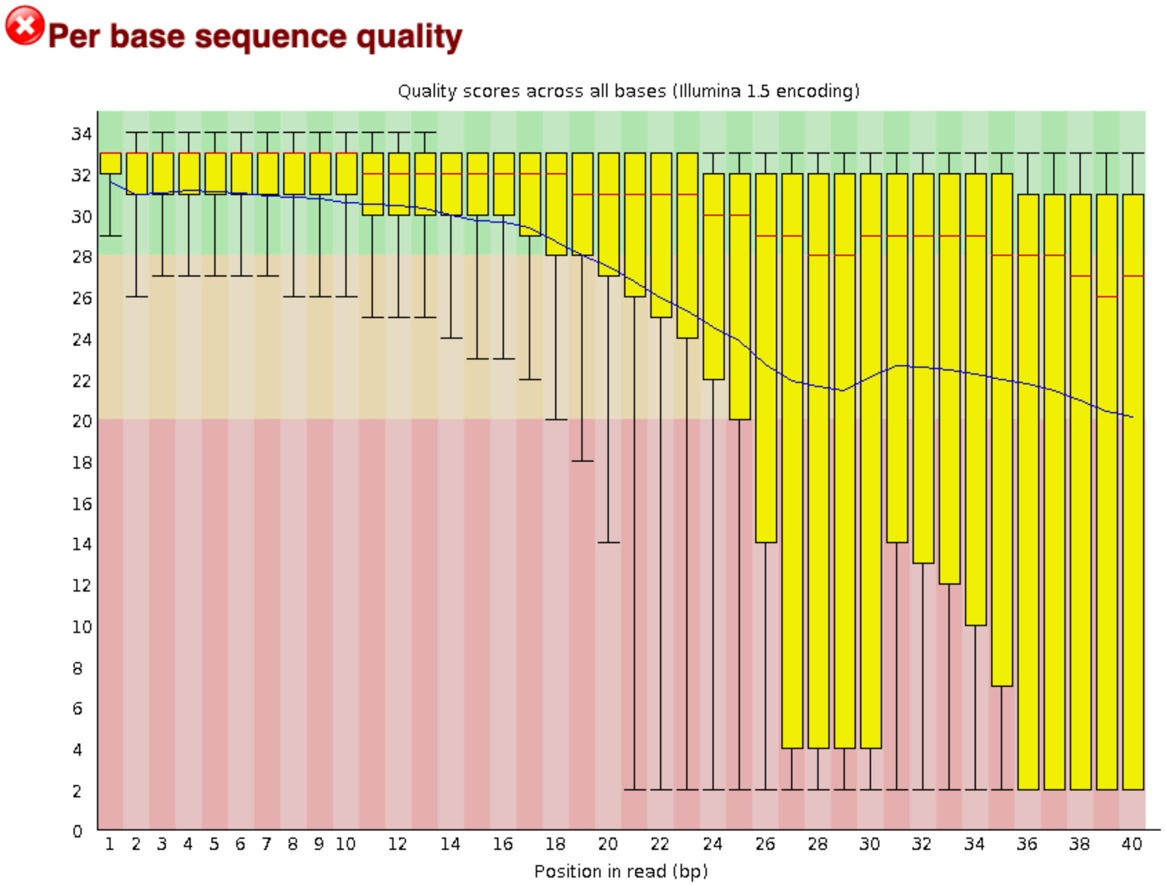
Measure	Value
Filename	bad_sequence.txt
File type	Conventional base calls
Encoding	Illumina 1.5
Total Sequences	395288
Sequences flagged as poor quality	0
Sequence length	40
%GC	47

Per base sequence quality

Good Quality



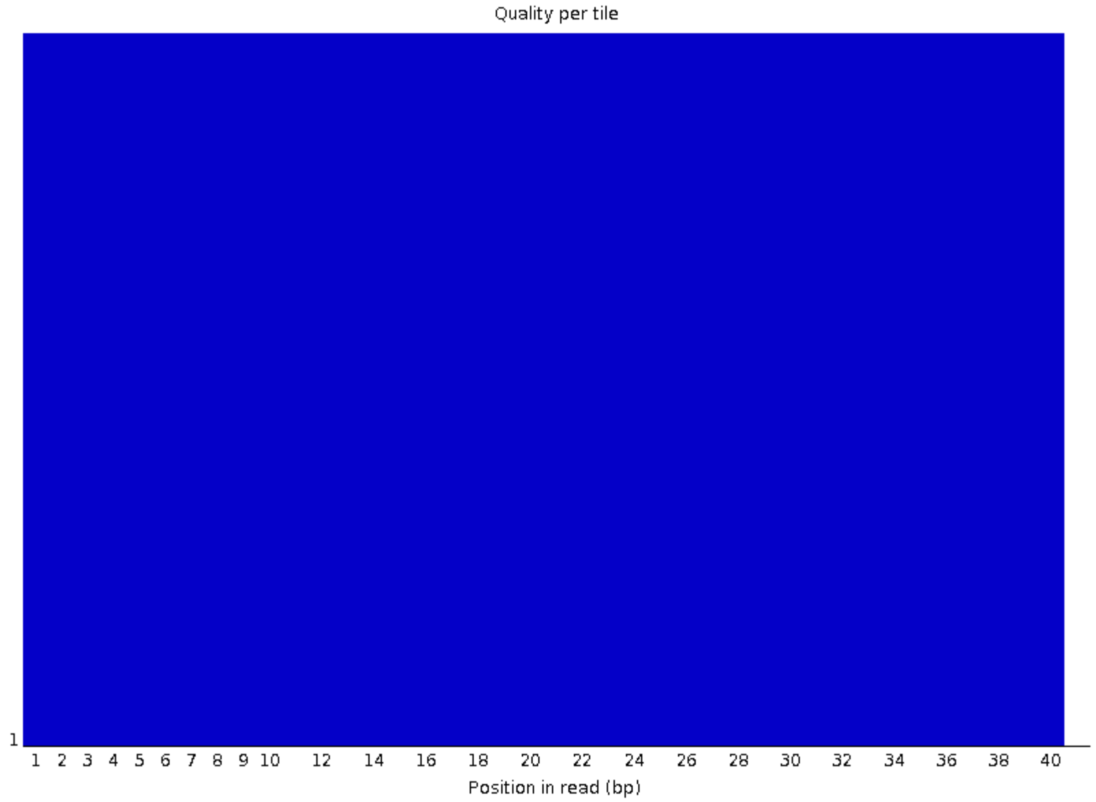
Bad Quality



Per tile sequence quality

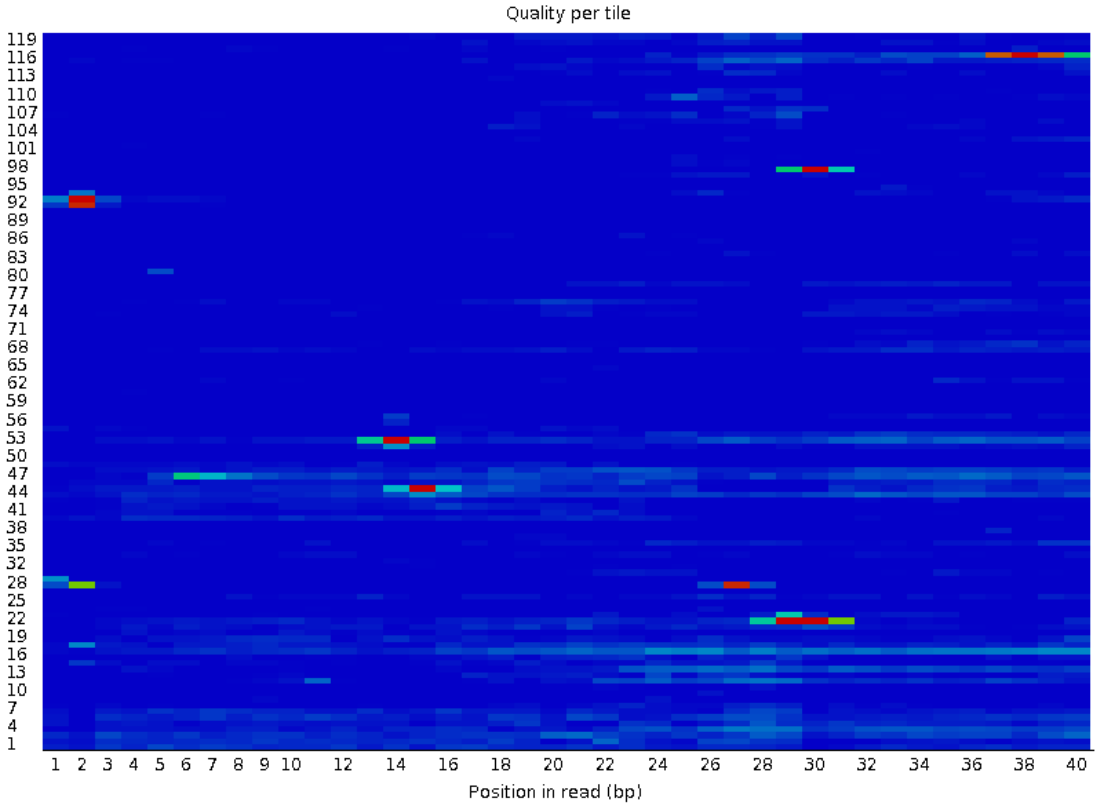
Good Quality

✓ Per tile sequence quality



Bad Quality

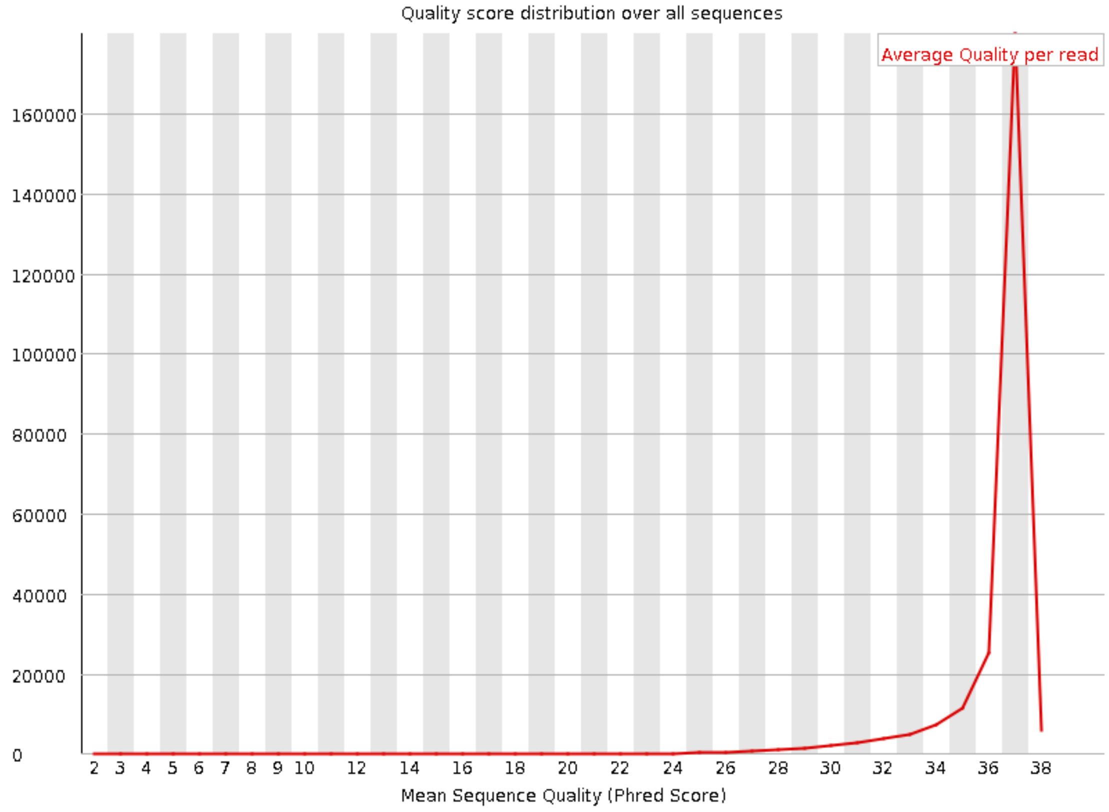
✗ Per tile sequence quality



Per sequence quality scores

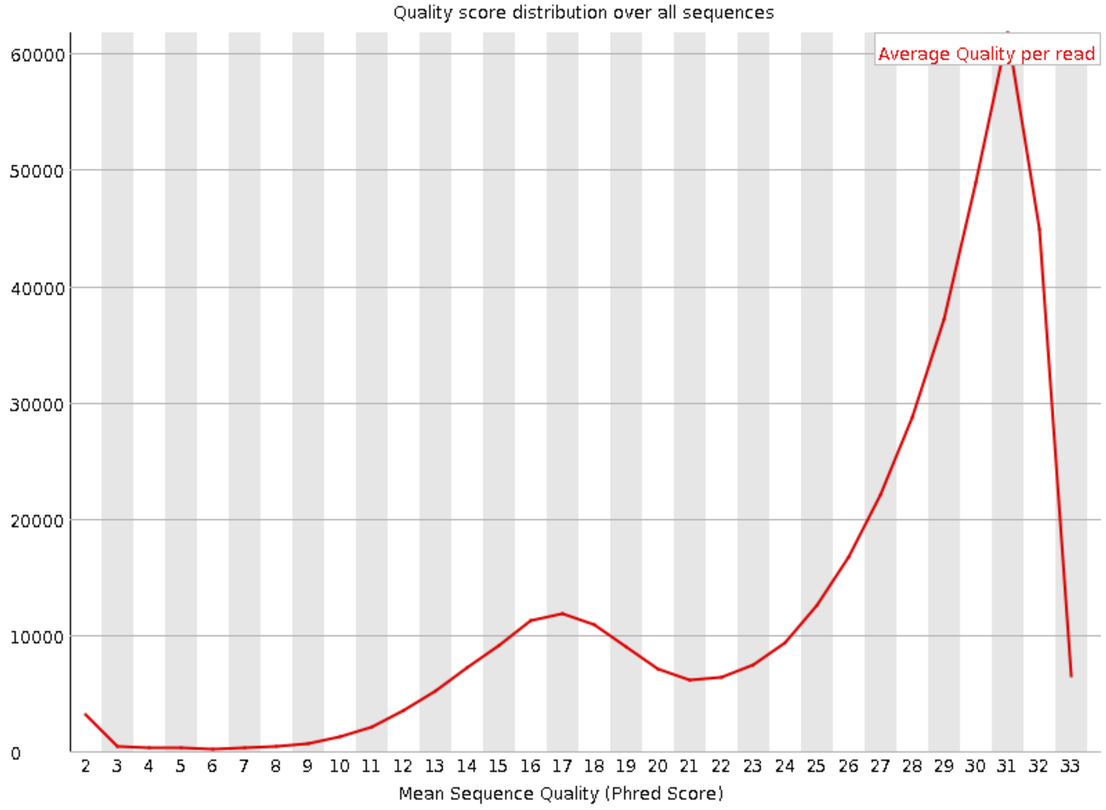
Good Quality

✓ **Per sequence quality scores**



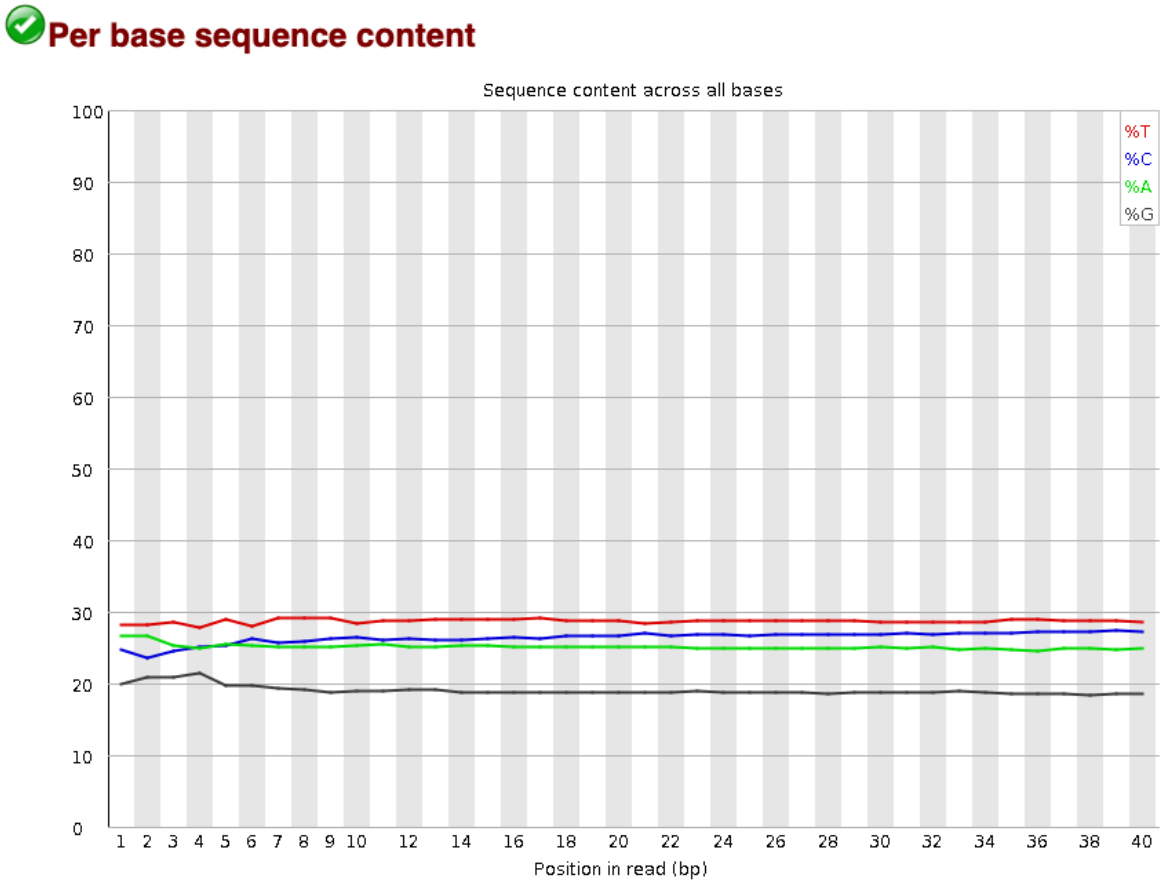
Bad Quality

✓ **Per sequence quality scores**

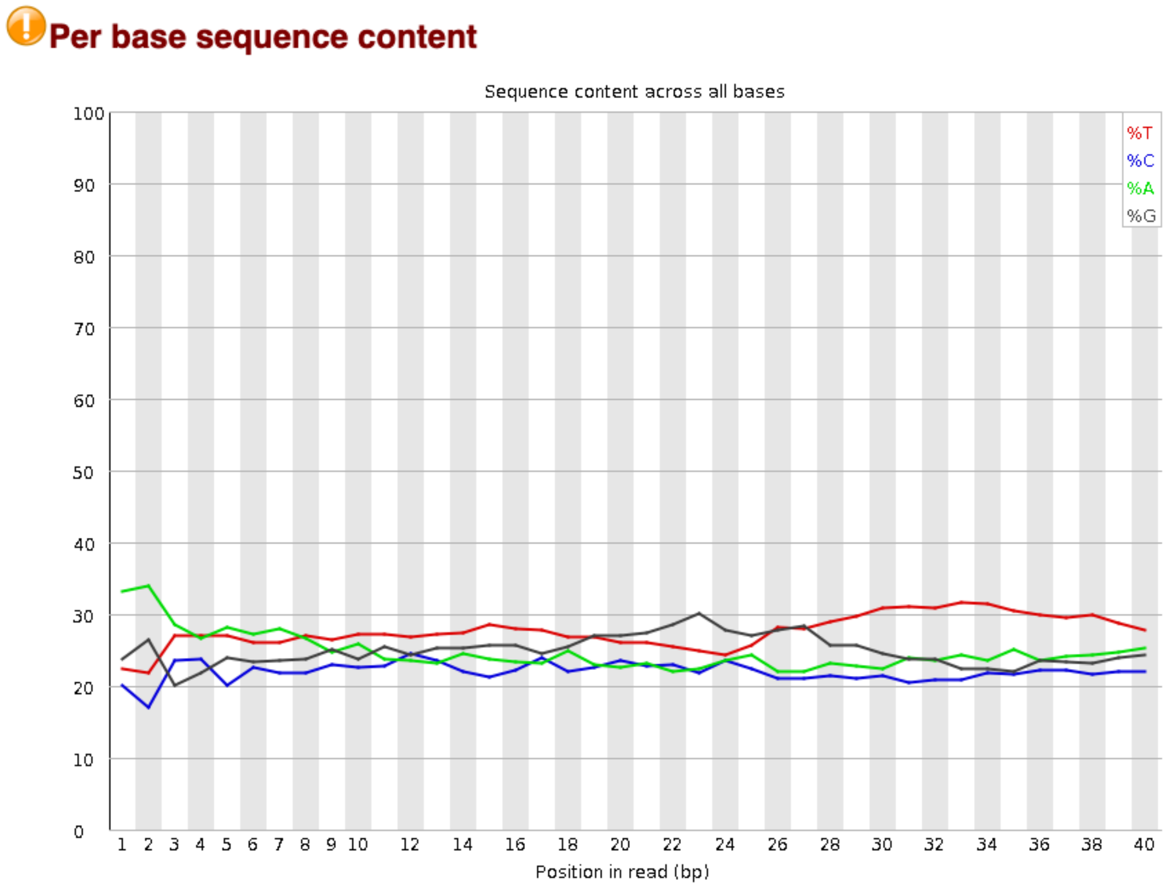


Per base sequence content

Good Quality

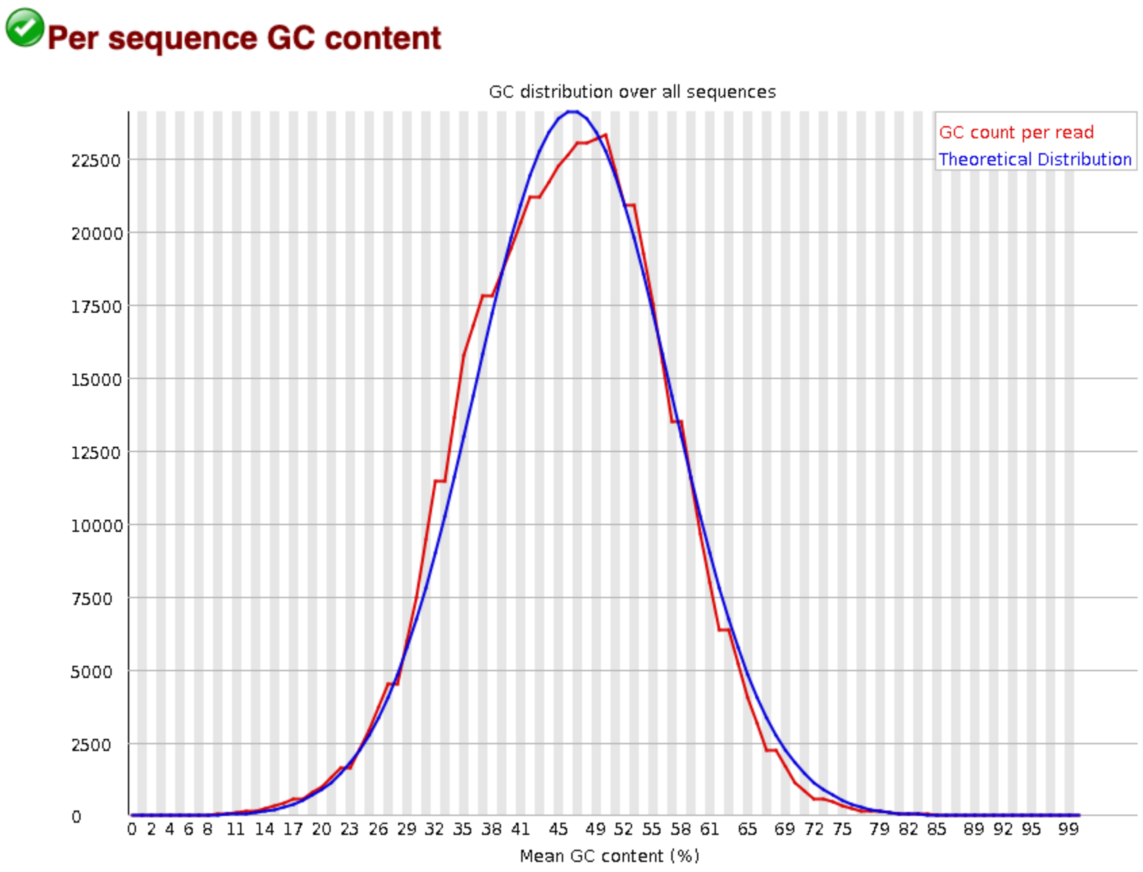


Bad Quality



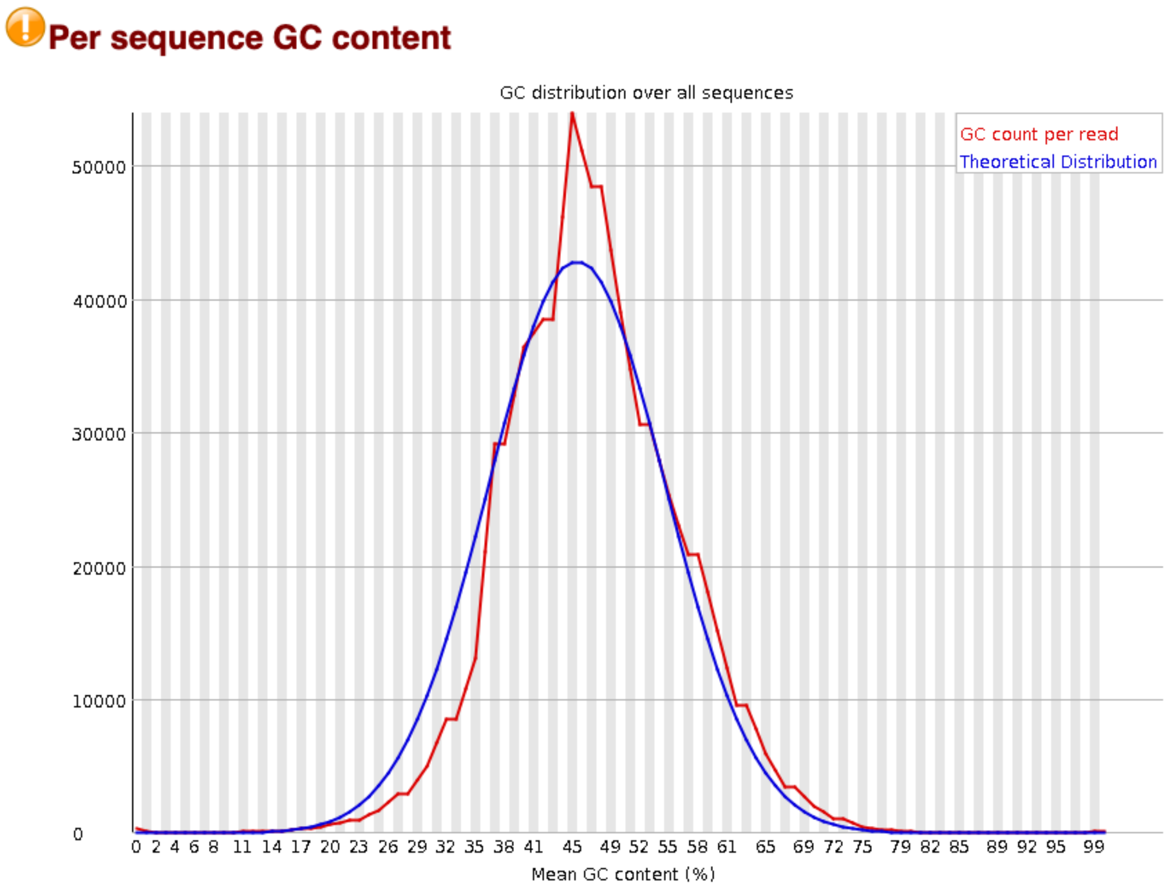
Per sequence GC content

Good Quality



Per sequence GC content

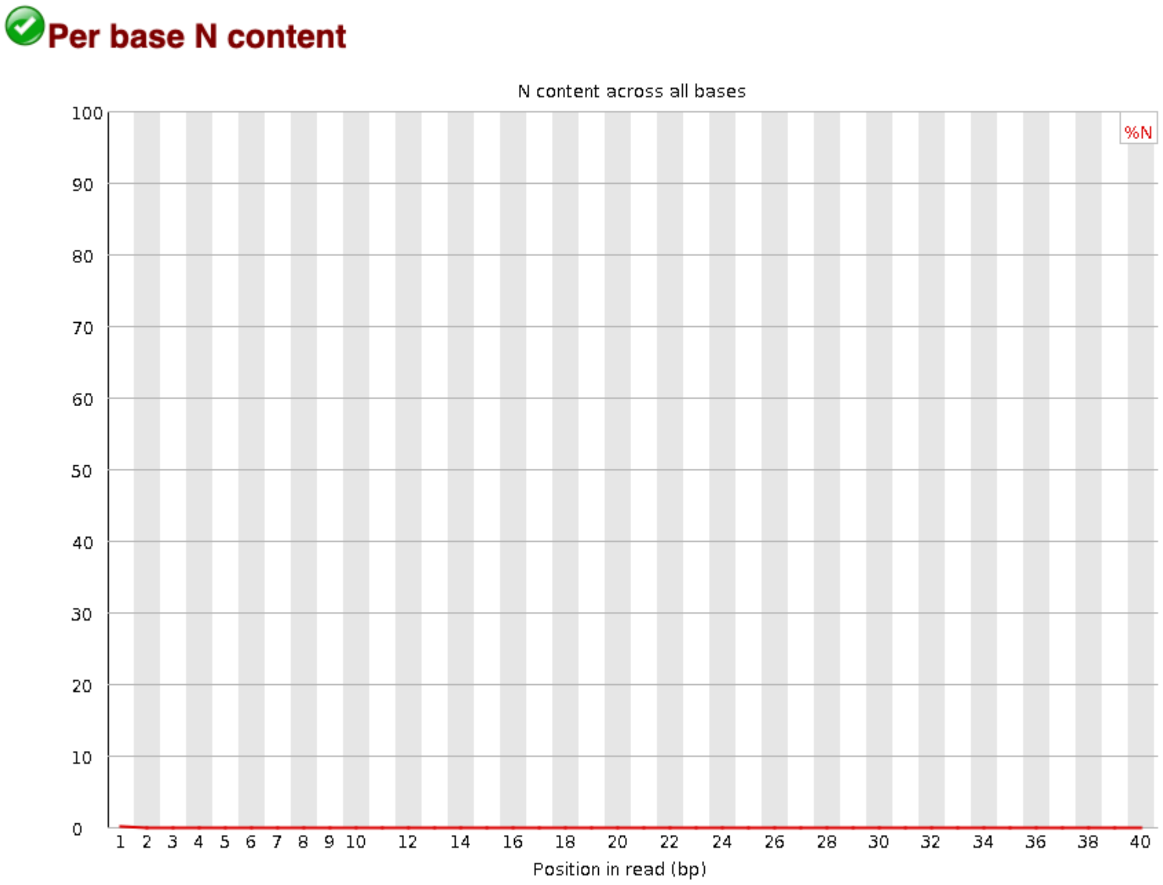
Bad Quality



Per sequence GC content

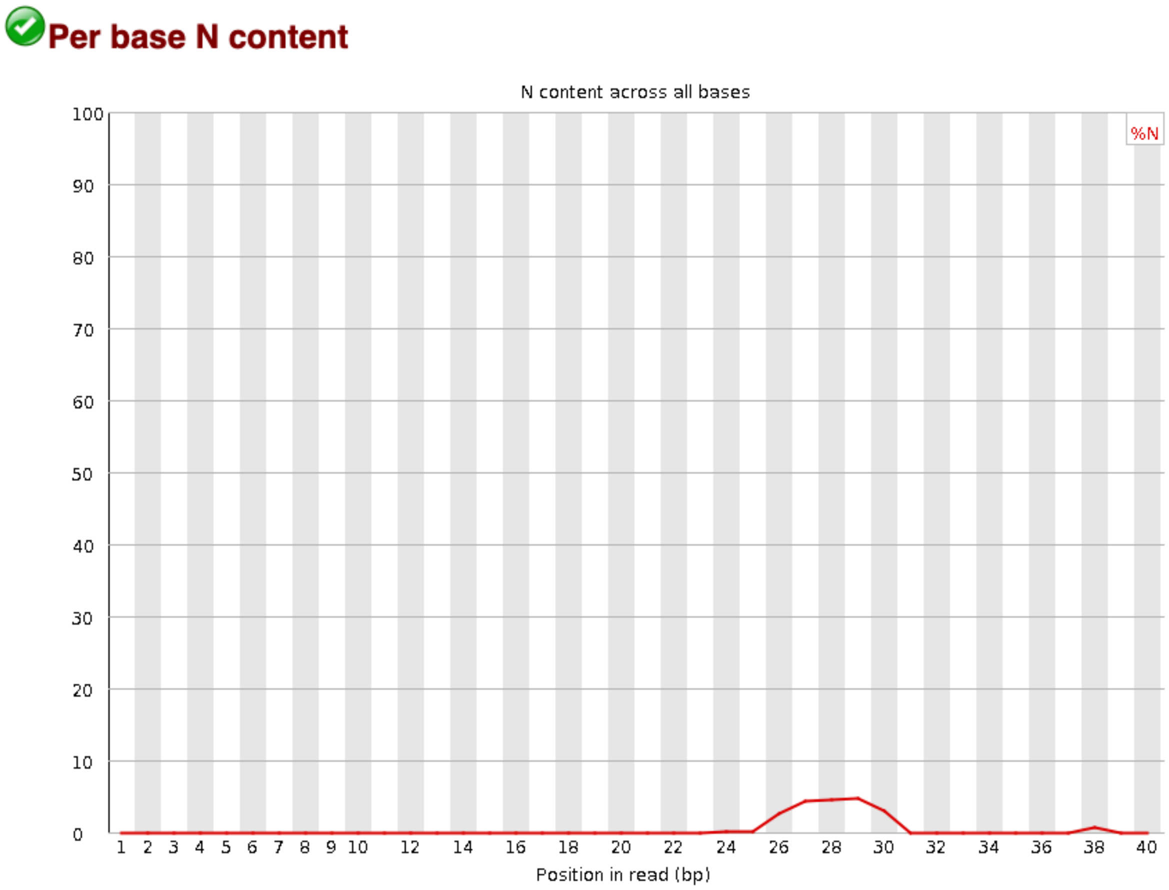
Per base sequence quality

Good Quality



Per base N content

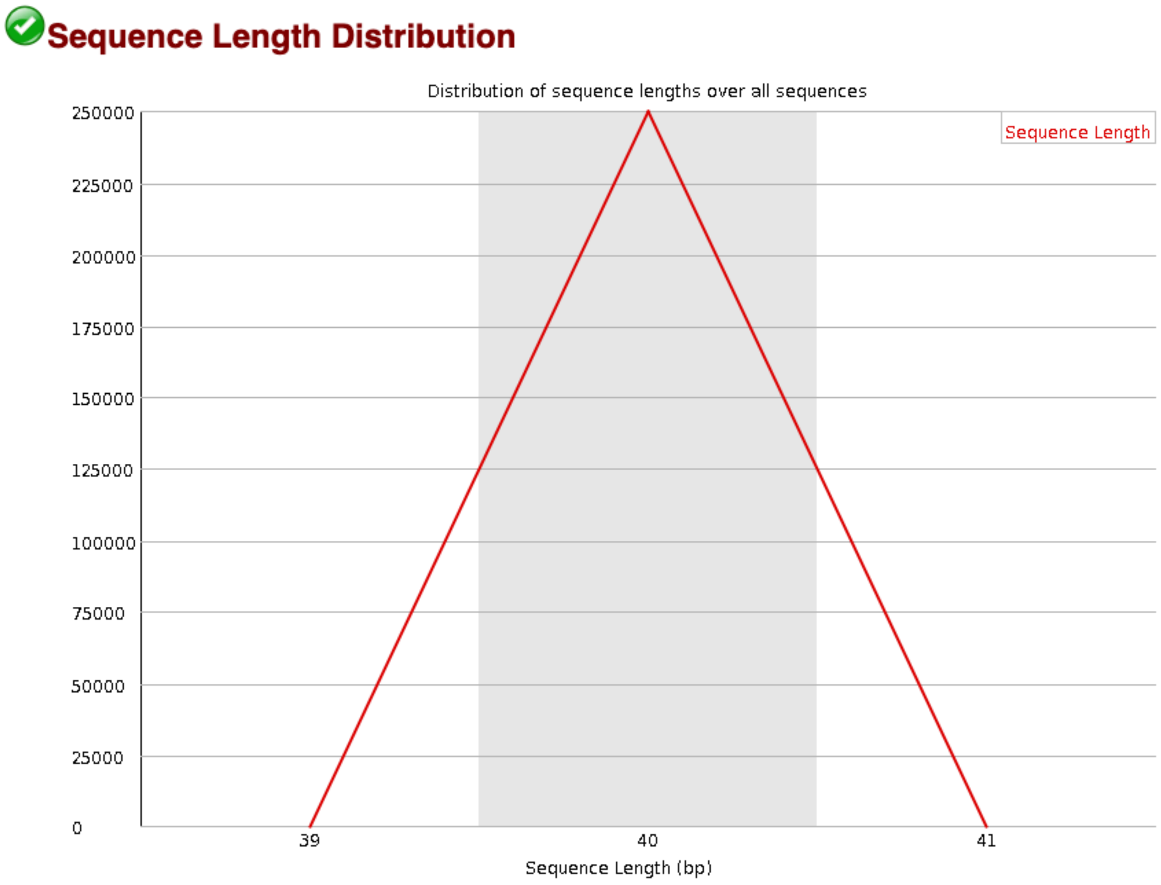
Bad Quality



Per base N content

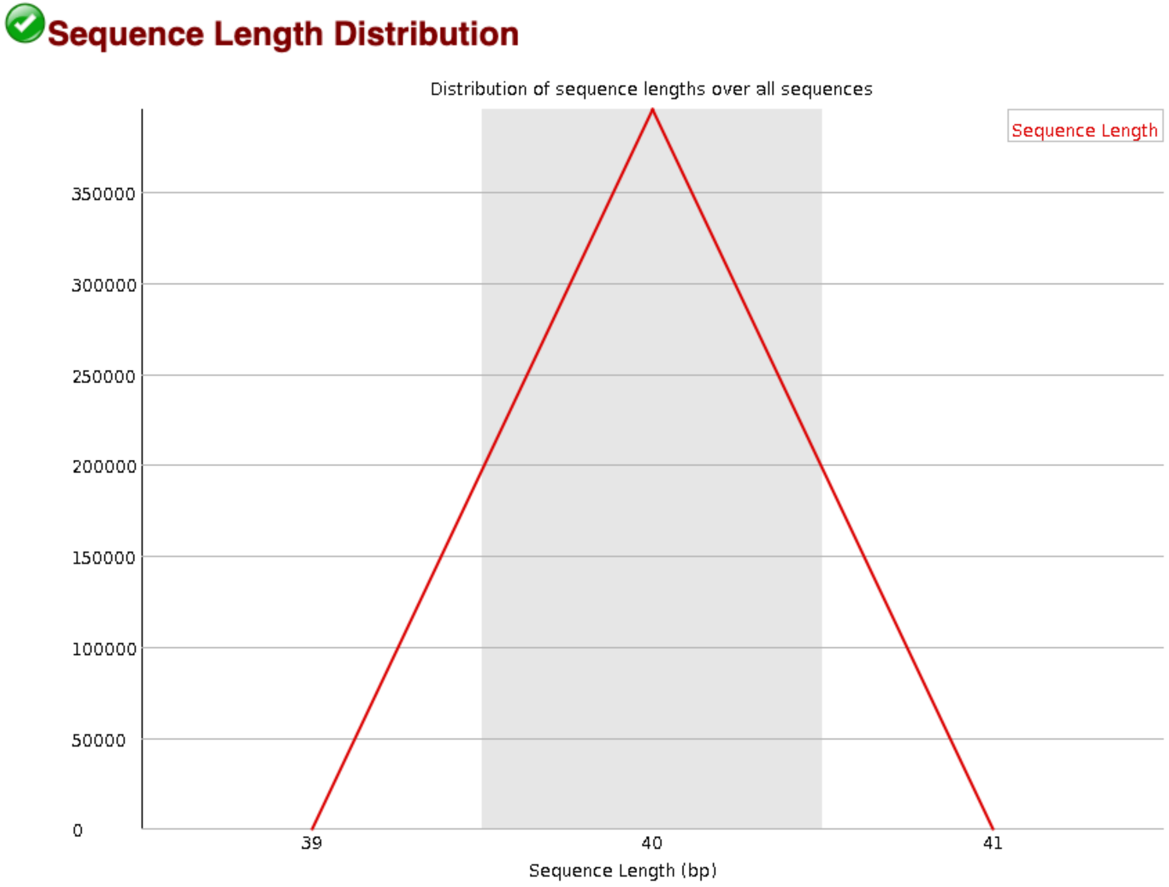
Per base sequence quality

Good Quality



✓ Sequence Length Distribution

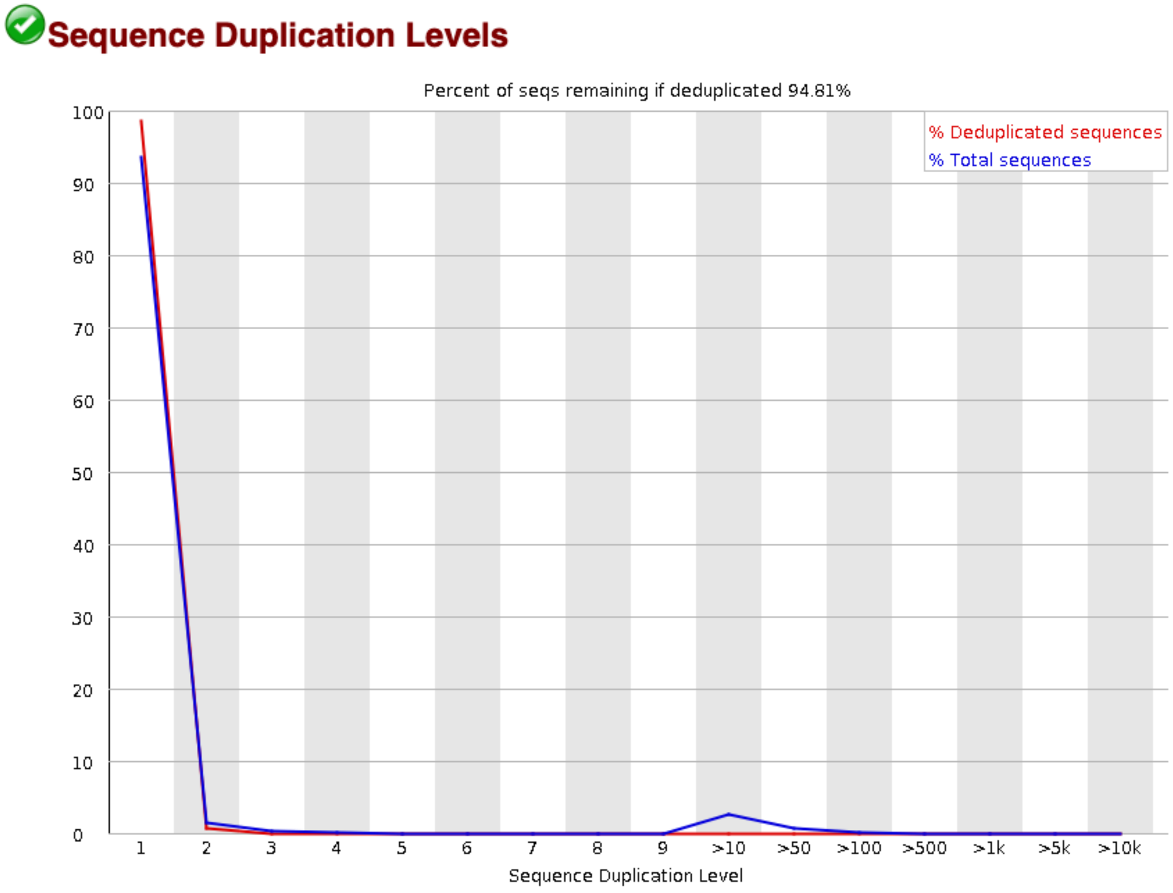
Bad Quality



✓ Sequence Length Distribution

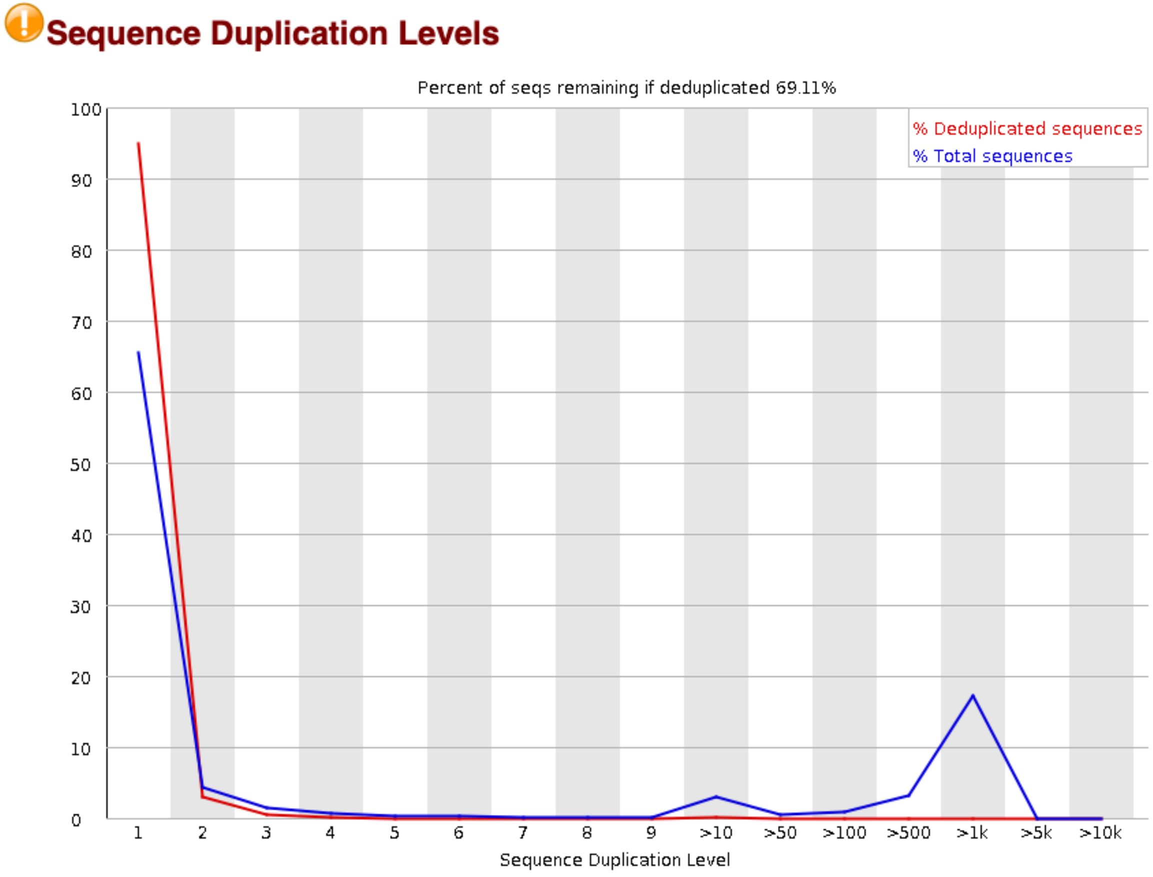
Sequence Duplication Levels

Good Quality


Sequence Duplication Levels

Percent of seqs remaining if deduplicated 94.81%

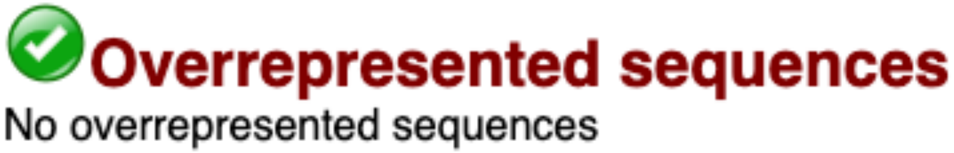
Bad Quality


Sequence Duplication Levels

Percent of seqs remaining if deduplicated 69.11%

Overrepresented sequences

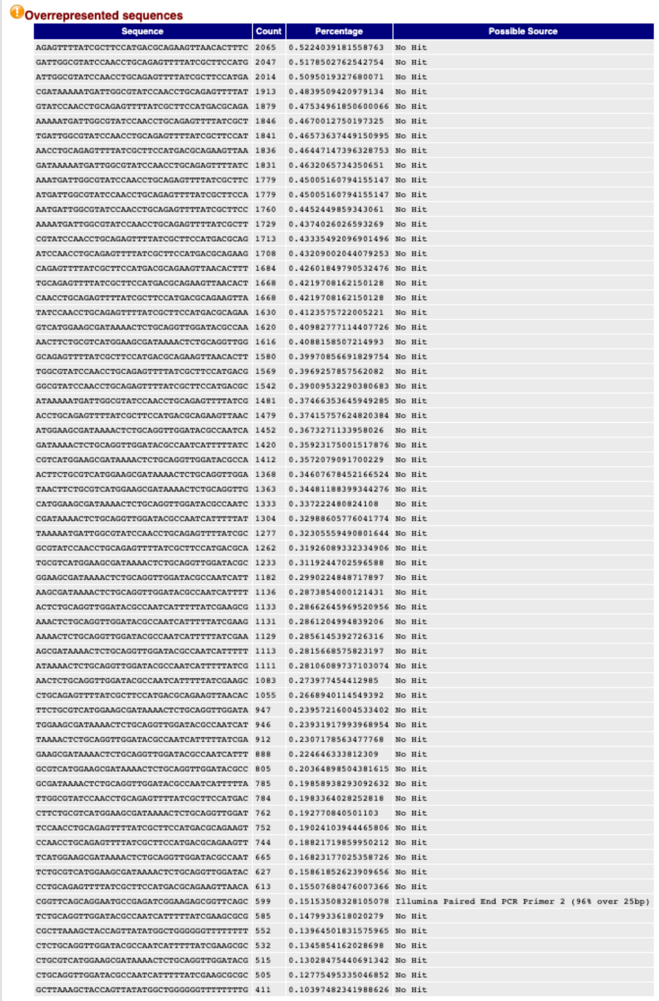
Good Quality



Overrepresented sequences

No overrepresented sequences

Bad Quality

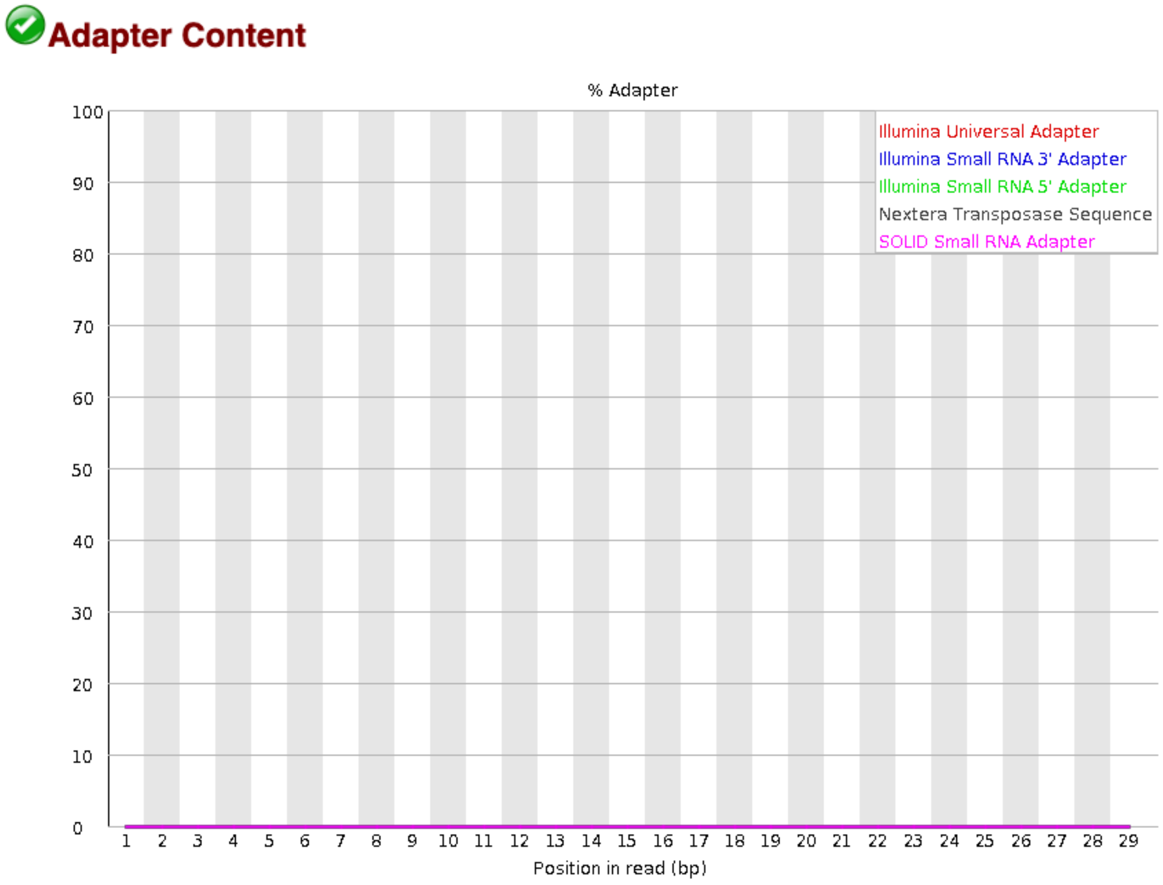


Overrepresented sequences

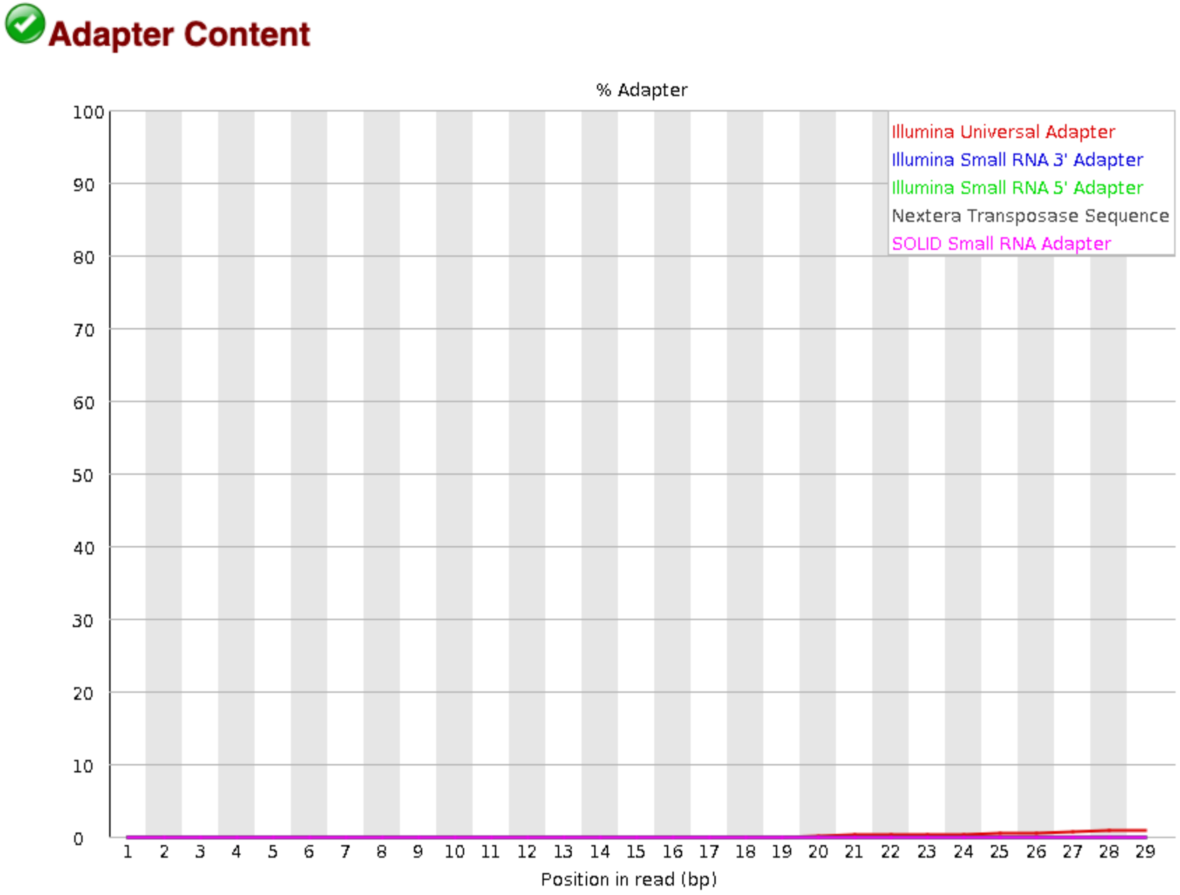
Sequence	Count	Percentage	Possible Source
AGAGTTTTATCGCTTCCATGACGCAGAAGTTAACACTTTC	2065	0.5224039181558763	No Hit
GATTGGCGTATCCAACCTGCAGAGTTTTATCGCTTCCATG	2047	0.5178502762542754	No Hit
ATTGGCGTATCCAACCTGCAGAGTTTTATCGCTTCCATGA	2014	0.5095019327680071	No Hit
CGATAAAAATGATTGGCGTATCCAACCTGCAGAGTTTTAT	1913	0.4839509420979134	No Hit
GTATCCAACCTGCAGAGTTTTATCGCTTCCATGACGCAGA	1879	0.47534961850600066	No Hit
AAAAATGATTGGCGTATCCAACCTGCAGAGTTTTATCGCT	1846	0.4670012750197325	No Hit
TGATTGGCGTATCCAACCTGCAGAGTTTTATCGCTTCCAT	1841	0.46573637449150995	No Hit
AACCTGCAGAGTTTTATCGCTTCCATGACGCAGAAGTTAA	1836	0.46447147396328753	No Hit
GATAAAAATGATTGGCGTATCCAACCTGCAGAGTTTTATC	1831	0.4632065734350651	No Hit
AAATGATTGGCGTATCCAACCTGCAGAGTTTTATCGCTTC	1779	0.45005160794155147	No Hit
ATGATTGGCGTATCCAACCTGCAGAGTTTTATCGCTTCCA	1779	0.45005160794155147	No Hit
AATGATTGGCGTATCCAACCTGCAGAGTTTTATCGCTTCC	1760	0.4452449859343061	No Hit
AAAATGATTGGCGTATCCAACCTGCAGAGTTTTATCGCTT	1729	0.4374026026593269	No Hit
CGTATCCAACCTGCAGAGTTTTATCGCTTCCATGACGCAG	1713	0.43335492096901496	No Hit
ATCCAACCTGCAGAGTTTTATCGCTTCCATGACGCAGAAG	1708	0.43209002044079253	No Hit
CAGAGTTTTATCGCTTCCATGACGCAGAAGTTAACACTTT	1684	0.42601849790532476	No Hit
TGCAGAGTTTTATCGCTTCCATGACGCAGAAGTTAACACT	1668	0.4219708162150128	No Hit
CAACCTGCAGAGTTTTATCGCTTCCATGACGCAGAAGTTA	1668	0.4219708162150128	No Hit
TATCCAACCTGCAGAGTTTTATCGCTTCCATGACGCAGAA	1630	0.4123575722005221	No Hit
GTCATGGAAGCGATAAAACTCTGCAGGTTGGATACGCCAA	1620	0.40982777114407726	No Hit
AACTTCTGCGTCATGGAAGCGATAAAACTCTGCAGGTTGG	1616	0.4088158507214993	No Hit
GCAGAGTTTTATCGCTTCCATGACGCAGAAGTTAACACTT	1580	0.39970856691829754	No Hit
TGGCGTATCCAACCTGCAGAGTTTTATCGCTTCCATGACG	1569	0.3969257857562082	No Hit
GGCGTATCCAACCTGCAGAGTTTTATCGCTTCCATGACGC	1542	0.39009532290380683	No Hit
ATAAAAATGATTGGCGTATCCAACCTGCAGAGTTTTATCG	1481	0.37466353645949285	No Hit
ACCTGCAGAGTTTTATCGCTTCCATGACGCAGAAGTTAAC	1479	0.37415757624820384	No Hit
ATGGAAGCGATAAAACTCTGCAGGTTGGATACGCCAATCA	1452	0.3673271133958026	No Hit
GATAAAACTCTGCAGGTTGGATACGCCAATCATTTTTATC	1420	0.35923175001517876	No Hit
CGTCATGGAAGCGATAAAACTCTGCAGGTTGGATACGCCA	1412	0.3572079091700229	No Hit
ACTTCTGCGTCATGGAAGCGATAAAACTCTGCAGGTTGGA	1368	0.34607678452166524	No Hit
TAACTTCTGCGTCATGGAAGCGATAAAACTCTGCAGGTTG	1363	0.34481188399344276	No Hit
CATGGAAGCGATAAAACTCTGCAGGTTGGATACGCCAATC	1333	0.337222480824108	No Hit
CGATAAAACTCTGCAGGTTGGATACGCCAATCATTTTTAT	1304	0.32988605776041774	No Hit
TAAAAATGATTGGCGTATCCAACCTGCAGAGTTTTATCGC	1277	0.32305559490801644	No Hit
GCGTATCCAACCTGCAGAGTTTTATCGCTTCCATGACGCA	1262	0.31926089332334906	No Hit
TGCGTCATGGAAGCGATAAAACTCTGCAGGTTGGATACGC	1233	0.3119244702596588	No Hit
GGAAGCGATAAAACTCTGCAGGTTGGATACGCCAATCATT	1182	0.2990224848717897	No Hit
AAGCGATAAAACTCTGCAGGTTGGATACGCCAATCATTTT	1136	0.2873854000121431	No Hit
ACTCTGCAGGTTGGATACGCCAATCATTTTTATCGAAGCG	1133	0.28662645969520956	No Hit
AAACTCTGCAGGTTGGATACGCCAATCATTTTTATCGAAG	1131	0.2861204994839206	No Hit
AAAACTCTGCAGGTTGGATACGCCAATCATTTTTATCGAA	1129	0.2856145392726316	No Hit
AGCGATAAAACTCTGCAGGTTGGATACGCCAATCATTTTT	1113	0.2815668575823197	No Hit
ATAAAACTCTGCAGGTTGGATACGCCAATCATTTTTATCG	1111	0.28106089737103074	No Hit
AACTCTGCAGGTTGGATACGCCAATCATTTTTATCGAAGC	1083	0.273977454412985	No Hit
CTGCAGAGTTTTATCGCTTCCATGACGCAGAAGTTAACAC	1055	0.2668940114549392	No Hit
TTCTGCGTCATGGAAGCGATAAAACTCTGCAGGTTGGATA	947	0.23957216004533402	No Hit
TGGAAGCGATAAAACTCTGCAGGTTGGATACGCCAATCAT	946	0.23931917993968954	No Hit
TAAAACTCTGCAGGTTGGATACGCCAATCATTTTTATCGA	912	0.2307178563477768	No Hit
GAAGCGATAAAACTCTGCAGGTTGGATACGCCAATCATTT	888	0.224646333812309	No Hit
GCGTCATGGAAGCGATAAAACTCTGCAGGTTGGATACGCC	805	0.20364898504381615	No Hit
GCGATAAAACTCTGCAGGTTGGATACGCCAATCATTTTTA	785	0.19858938293092632	No Hit
TTGGCGTATCCAACCTGCAGAGTTTTATCGCTTCCATGAC	784	0.1983364028252818	No Hit
CTTCTGCGTCATGGAAGCGATAAAACTCTGCAGGTTGGAT	762	0.192770840501103	No Hit
TCCAACCTGCAGAGTTTTATCGCTTCCATGACGCAGAAGT	752	0.19024103944465806	No Hit
CCAACCTGCAGAGTTTTATCGCTTCCATGACGCAGAAGTT	744	0.18821719859950212	No Hit
TCATGGAAGCGATAAAACTCTGCAGGTTGGATACGCCAAT	665	0.16823177025358726	No Hit
TCTGCGTCATGGAAGCGATAAAACTCTGCAGGTTGGATAC	627	0.15861852623909656	No Hit
CCTGCAGAGTTTTATCGCTTCCATGACGCAGAAGTTAACA	613	0.15507680476007366	No Hit
CGGTTCAGCAGGAATGCCGAGATCGGAAGAGCGGTTCAGC	599	0.15153508328105078	Illumina Paired End PCR Primer 2 (96% over 25bp)
TCTGCAGGTTGGATACGCCAATCATTTTTATCGAAGCGCG	585	0.1479933618020279	No Hit
CGCTTAAAGCTACCAGTTATATGGCTGGGGGGTTTTTTTT	552	0.13964501831575965	No Hit
CTCTGCAGGTTGGATACGCCAATCATTTTTATCGAAGCGC	532	0.1345854162028698	No Hit
CTGCGTCATGGAAGCGATAAAACTCTGCAGGTTGGATACG	515	0.13028475440691342	No Hit
CTGCAGGTTGGATACGCCAATCATTTTTATCGAAGCGCGC	505	0.12775495335046852	No Hit
GCTTAAAGCTACCAGTTATATGGCTGGGGGGTTTTTTTTG	411	0.10397482341988626	No Hit

Adapter Content

Good Quality



Bad Quality



Sidenote - Loops in Bash

What’s a loop?

What’s a loop?

A loop is a piece of code that repeatedly executes a command for a certain condition until that condition is no longer true.

What’s a loop?

A loop is a piece of code that repeatedly executes a command for a certain condition until that condition is no longer true.

```
for i in *.fastq.gz;
   do echo $i;
done
```

What’s a loop?

A loop is a piece of code that repeatedly executes a command for a certain condition until that condition is no longer true.

```
for i in *.fastq.gz;
   do echo $i;
done
```

condition

What’s a loop?

A loop is a piece of code that repeatedly executes a command for a certain condition until that condition is no longer true.

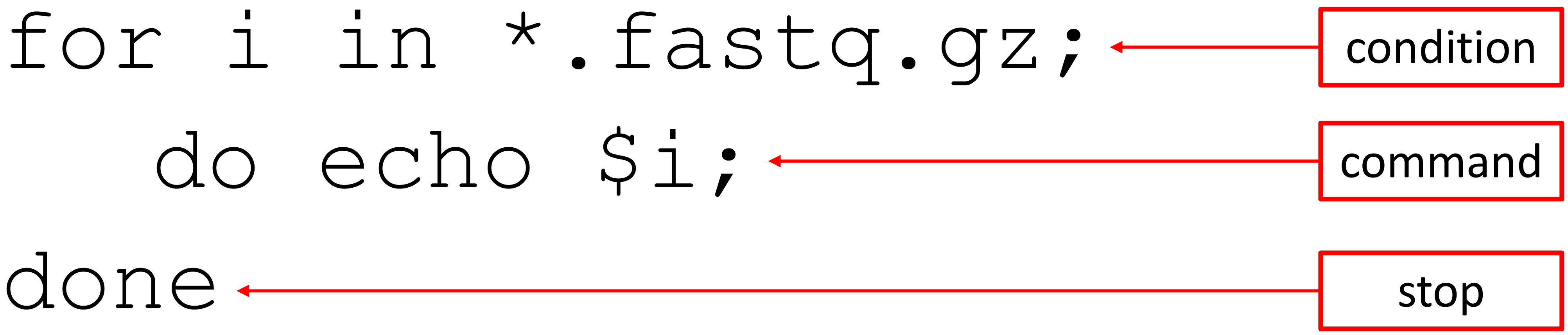
```
for i in *.fastq.gz;
    do echo $i;
done
```

condition

command

What’s a loop?

A loop is a piece of code that repeatedly executes a command for a certain condition until that condition is no longer true.



```
for i in *.fastq.gz;
   do echo $i;
done
```

How do you write a loop in bash?

How do you write a loop in bash?

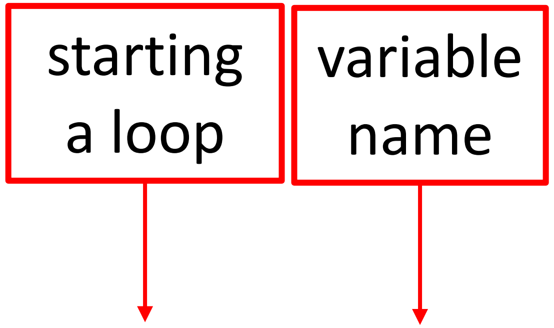
```
for i in *.fastq.gz;
   do echo $i;
done
```

How do you write a loop in bash?

starting a loop

```
for i in *.fastq.gz;
   do echo $i;
done
```

How do you write a loop in bash?

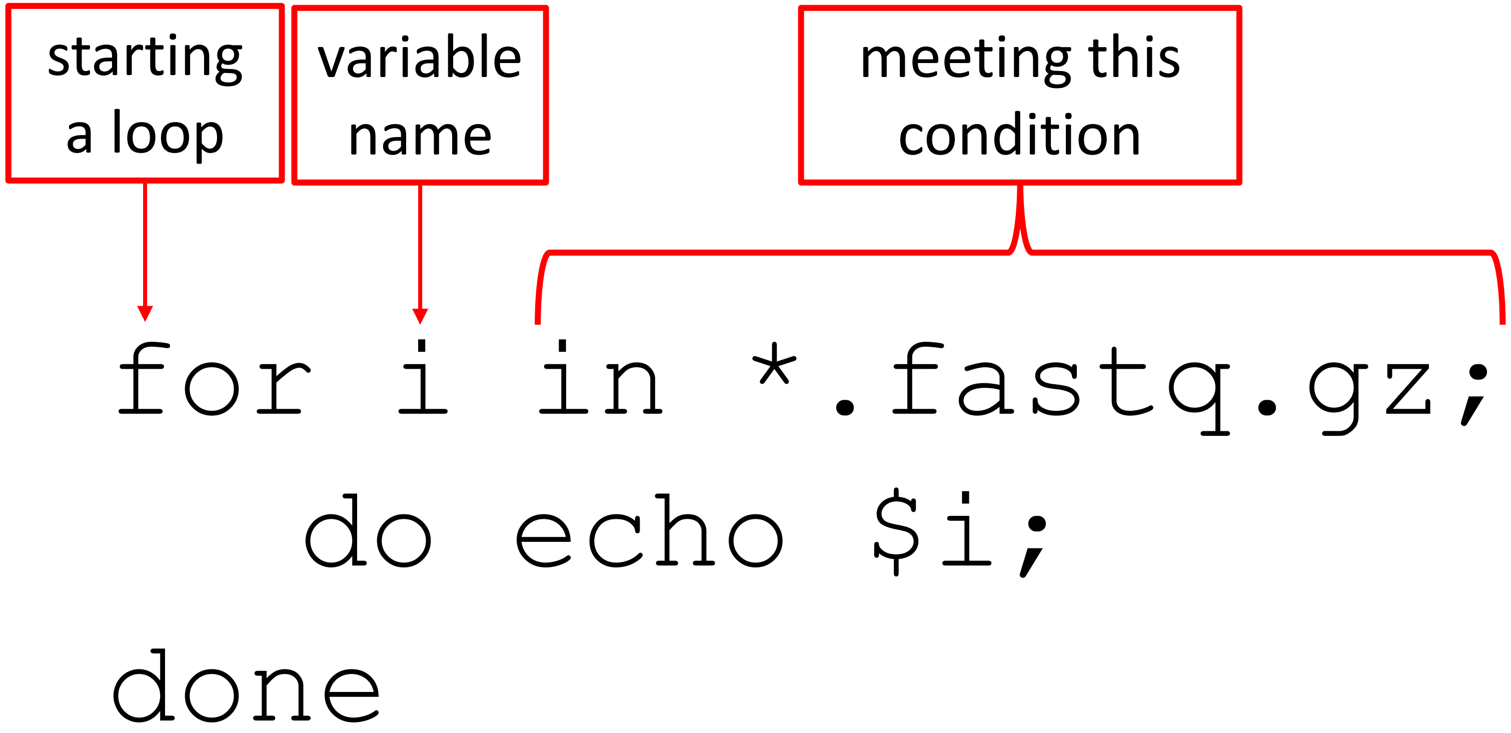


starting a loop

variable name

```
for i in *.fastq.gz;
    do echo $i;
done
```

How do you write a loop in bash?



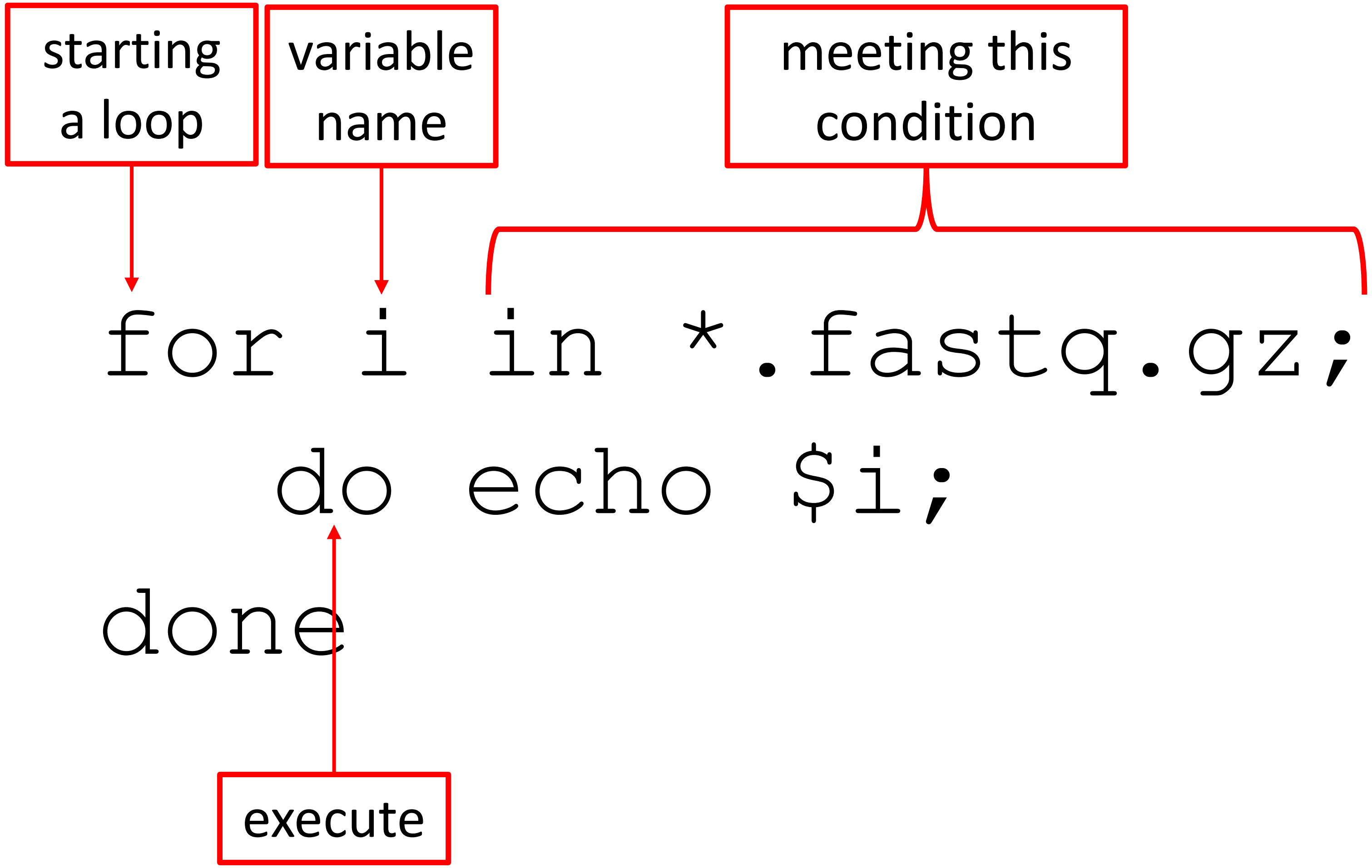
starting a loop

variable name

meeting this condition

```
for i in *.fastq.gz;
   do echo $i;
done
```

How do you write a loop in bash?



starting a loop

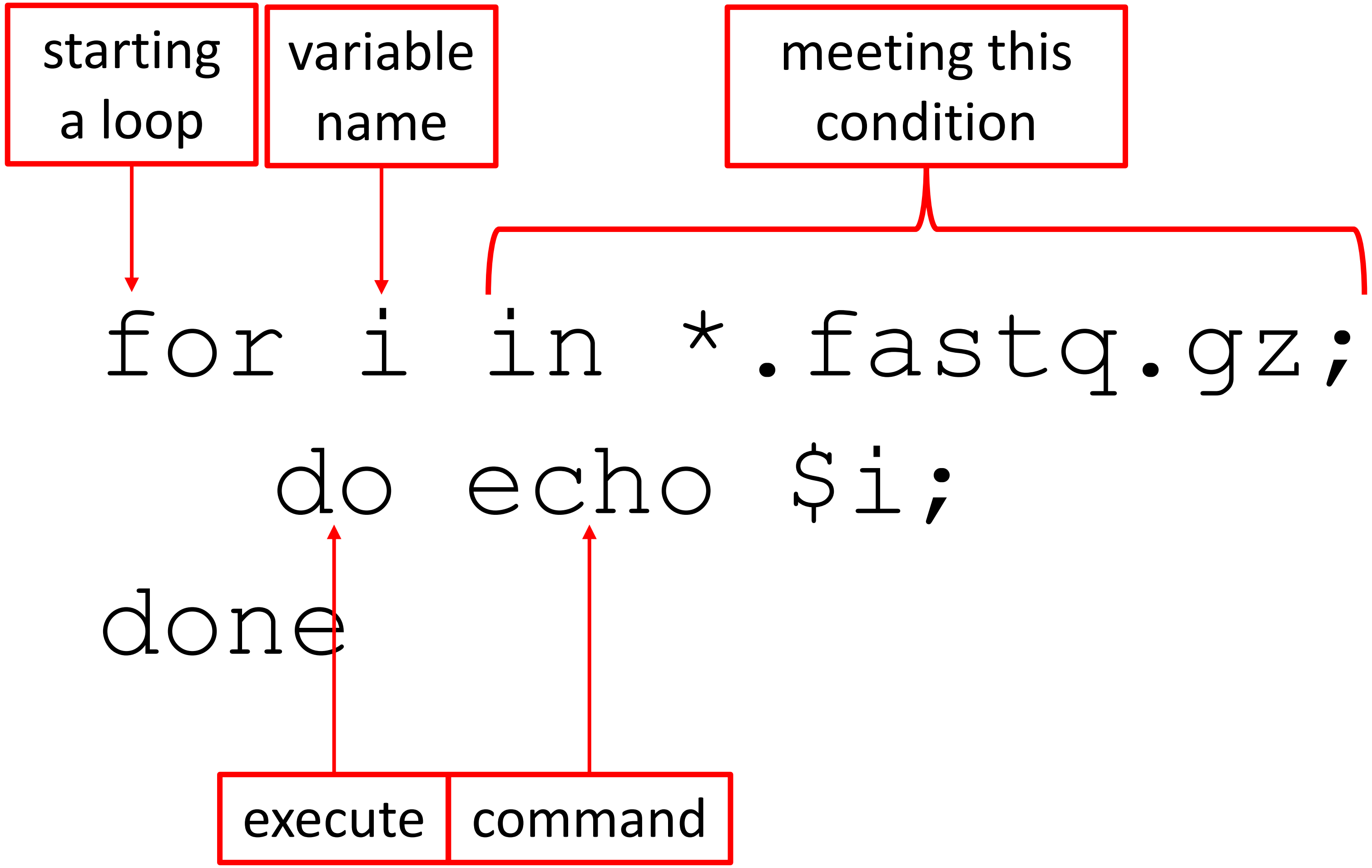
variable name

meeting this condition

```
for i in *.fastq.gz;
    do echo $i;
done
```

execute

How do you write a loop in bash?



starting a loop

variable name

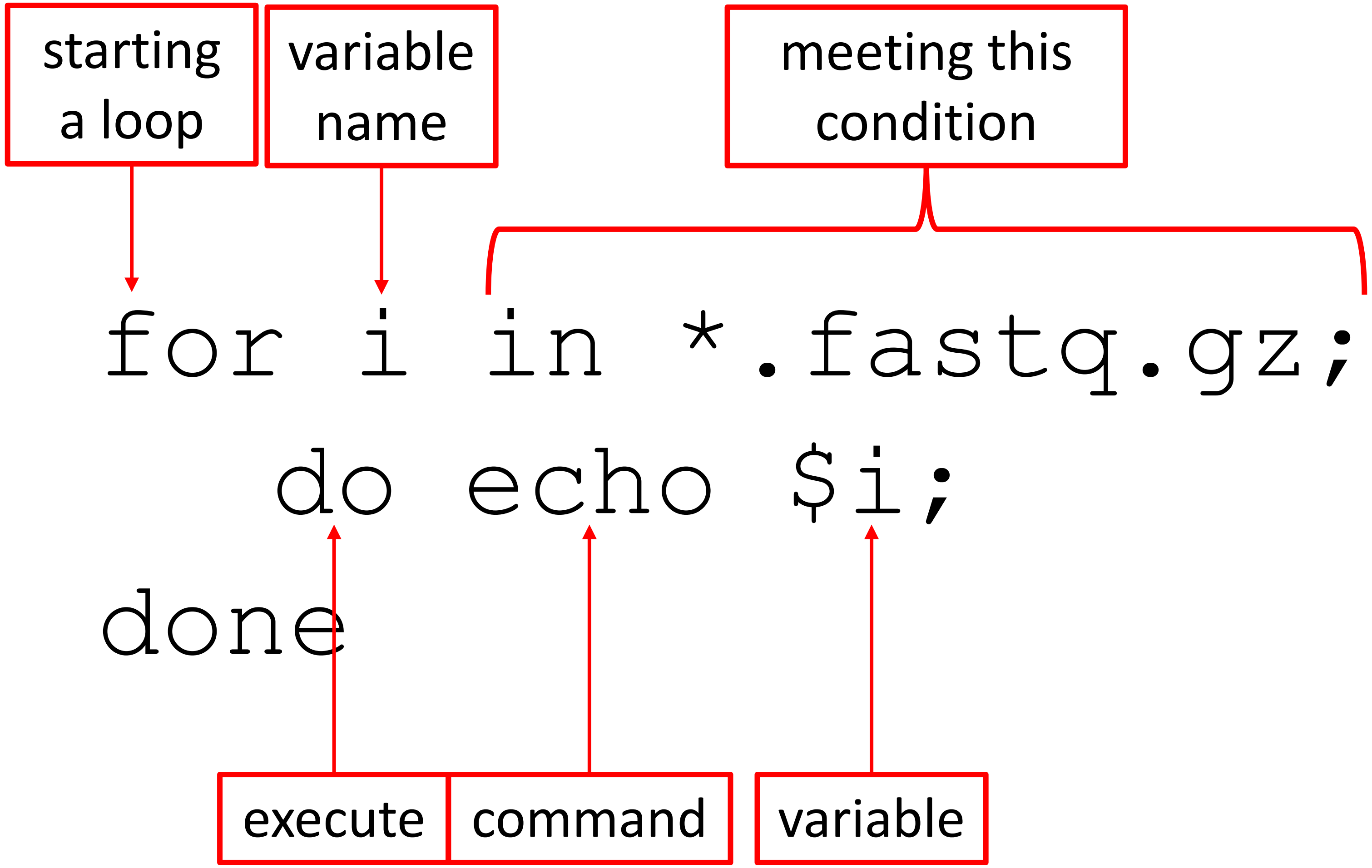
meeting this condition

```
for i in *.fastq.gz;
    do echo $i;
done
```

execute

command

How do you write a loop in bash?



starting a loop

variable name

meeting this condition

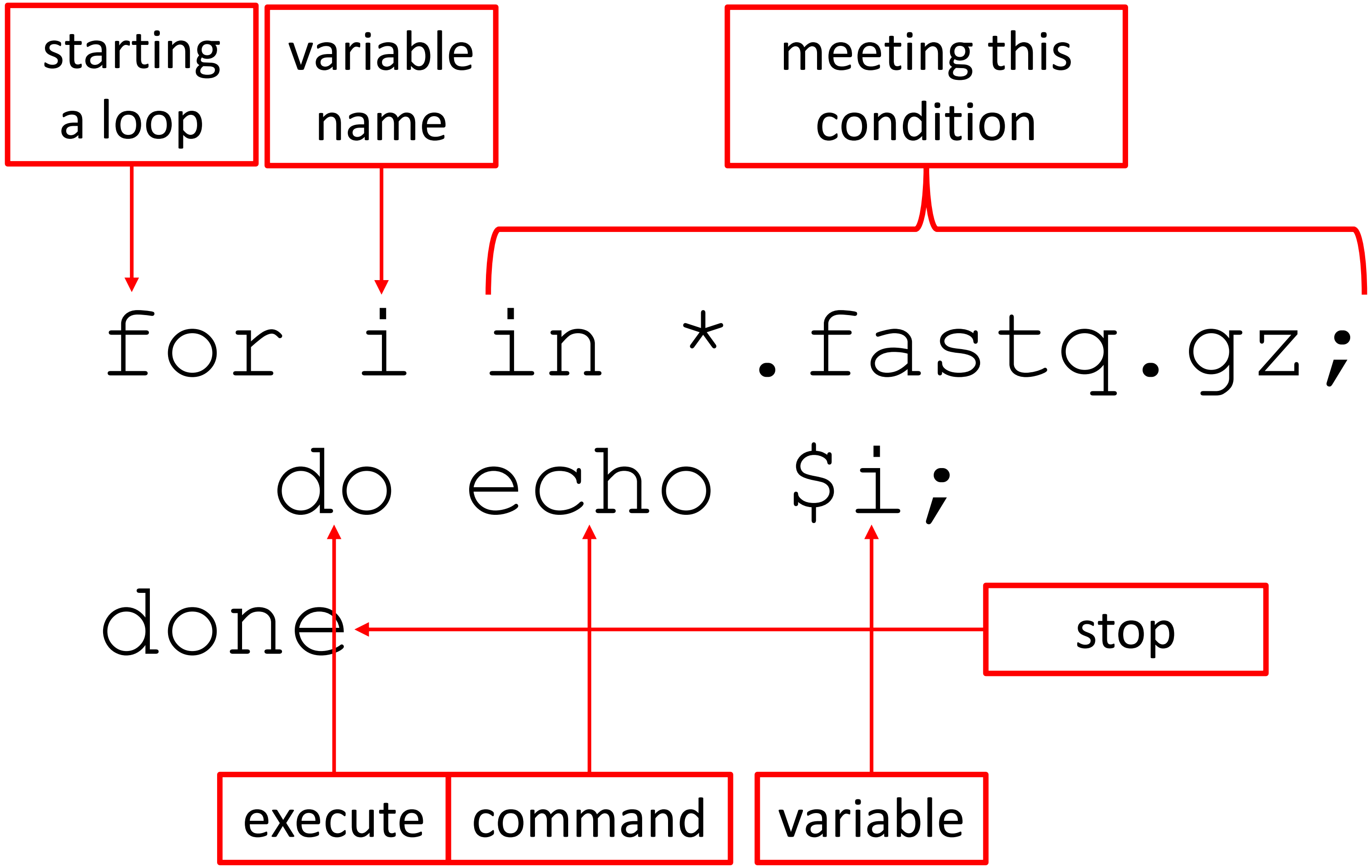
```
for i in *.fastq.gz;
    do echo $i;
done
```

execute

command

variable

How do you write a loop in bash?



starting a loop

variable name

meeting this condition

```
for i in *.fastq.gz;
    do echo $i;
done
```

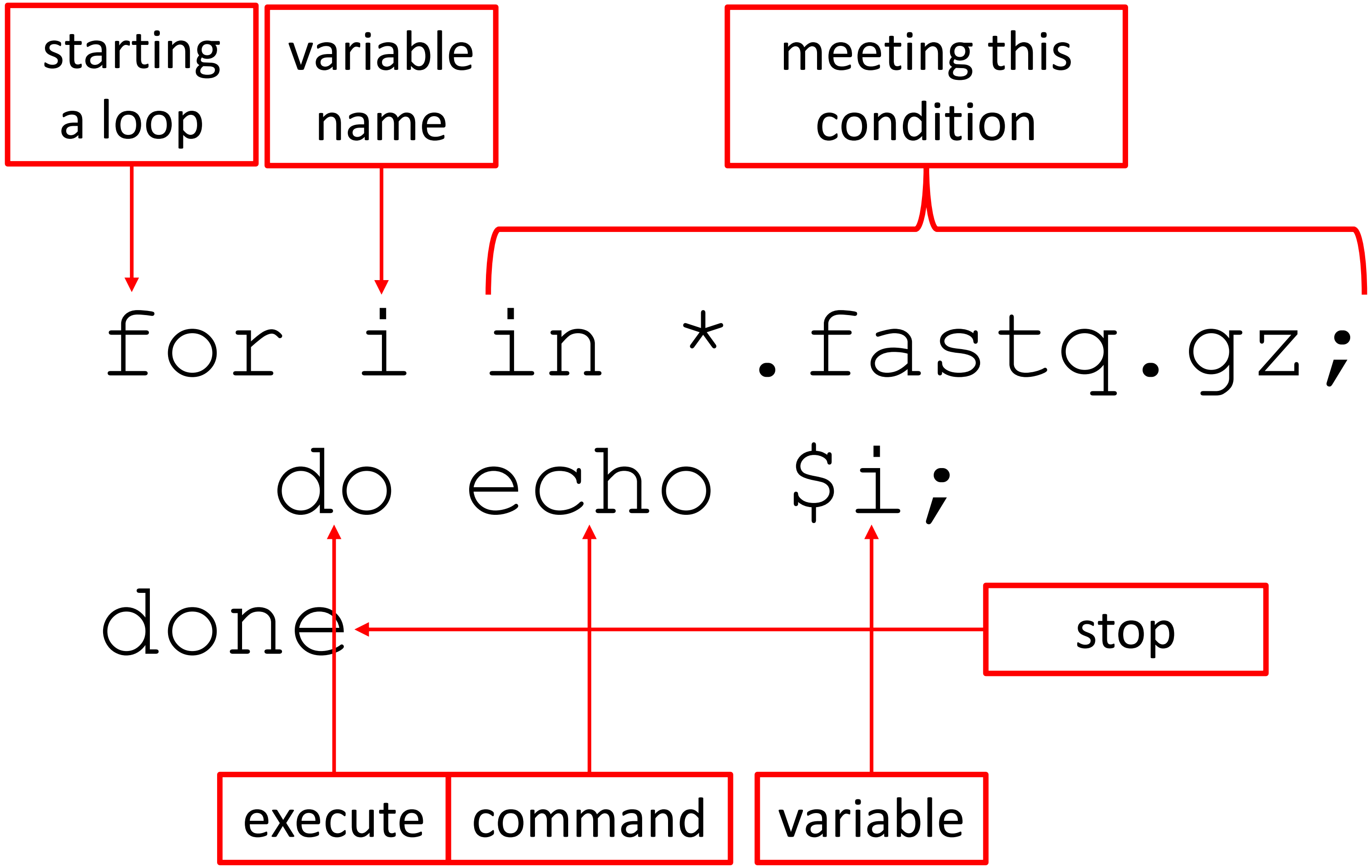
stop

execute

command

variable

How do you write a loop in bash?



```
for i in *.fastq.gz;
    do echo $i;
done
```

starting a loop

variable name

meeting this condition

execute

command

variable

stop

Now go try this loop on the server

Run FastQC

1. Go to the RNA-seq data directory
2. Make a directory to put the FastQC reports into, `fastqc`
3. Run fastqc on the samples

```
for i in *.fastq.gz; do fastqc $i -o fastqc/;
done
```

Trim Bad Quality Sequences

What is trimming and why do it?

What is trimming and why do it?

- Trimming removes sequencing adapters, bad quality sequences, and/or other biased sequence information

What is trimming and why do it?

- Trimming removes sequencing adapters, bad quality sequences, and/or other biased sequence information
- Why is that important?
  - Helps prevent incorrect base calls by removing poor quality information
  - Increases speed and accuracy of alignment by removing artificial sequences and low quality sequences

What is trimming and why do it?

- Trimming removes sequencing adapters, bad quality sequences, and/or other biased sequence information
- Why is that important?
  - Helps prevent incorrect base calls by removing poor quality information
  - Increases speed and accuracy of alignment by removing artificial sequences and low quality sequences

- Trimming does two complementary things:
  1. Removes any sequence information that comes from library preparation or sequencing
  2. Removes low quality bases / low quality reads

Trim Sequences

1. Go back up to the rnaseq directory
2. Make a folder to put the analysis results in, `analysis`
3. Make a folder inside the analysis folder to put the trimmed reads in, `analysis/01_trim`

Trim Sequences

```
for i in *R1.fastq.gz;
     do trim_galore
          --paired
          --fastqc
          --illumina
          --output analysis/01_trim/
          --retain_unpaired
          -q 30
          $i
          ${i/R1/R2};
done
```

Trim Sequences

```
for i in *R1.fastq.gz;
     do trim_galore
          --paired
          --fastqc
          --illumina
          --output analysis/01_trim/
          --retain_unpaired
          -q 30
          $i
          ${i/R1/R2};
done
```

loop condition

Trim Sequences

```
for i in *R1.fastq.gz;
     do trim_galore
          --paired
          --fastqc
          --illumina
          --output analysis/01_trim/
          --retain_unpaired
          -q 30
          $i
          ${i/R1/R2};
done
```

loop condition

call the program

Trim Sequences

```
for i in *R1.fastq.gz;
     do trim_galore
          --paired
          --fastqc
          --illumina
          --output analysis/01_trim/
          --retain_unpaired
          -q 30
          $i
          ${i/R1/R2};
done
```

loop condition

call the program

reads are paired-end

Trim Sequences

```
for i in *R1.fastq.gz;
     do trim_galore
          --paired
          --fastqc
          --illumina
          --output analysis/01_trim/
          --retain_unpaired
          -q 30
          $i
          ${i/R1/R2};
done
```

loop condition

call the program

reads are paired-end

run FastQC again after trimming

Trim Sequences

```
for i in *R1.fastq.gz;
      do trim_galore
            --paired
            --fastqc
            --illumina
            --output analysis/01_trim/
            --retain_unpaired
            -q 30
            $i
            ${i/R1/R2};
done
```

loop condition

call the program

reads are paired-end

run FastQC again after trimming

trim Illumina adapters

Trim Sequences

```
for i in *R1.fastq.gz;
     do trim_galore
          --paired
          --fastqc
          --illumina
          --output analysis/01_trim/
          --retain_unpaired
          -q 30
          $i
          ${i/R1/R2};
done
```

loop condition

call the program

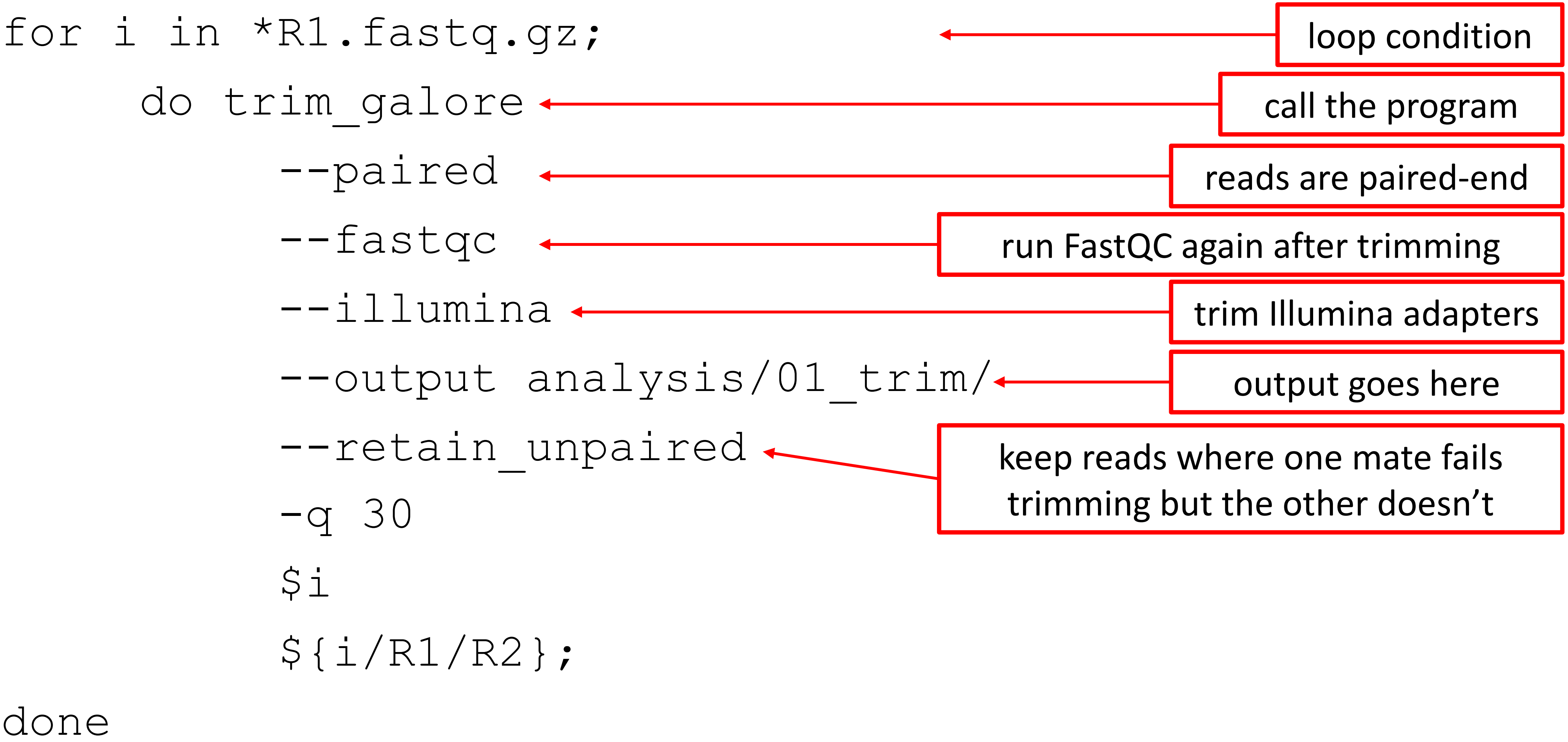
reads are paired-end

run FastQC again after trimming

trim Illumina adapters

output goes here

Trim Sequences



```
for i in *R1.fastq.gz;
     do trim_galore
          --paired
          --fastqc
          --illumina
          --output analysis/01_trim/
          --retain_unpaired
          -q 30
          $i
          ${i/R1/R2};
done
```

loop condition

call the program

reads are paired-end

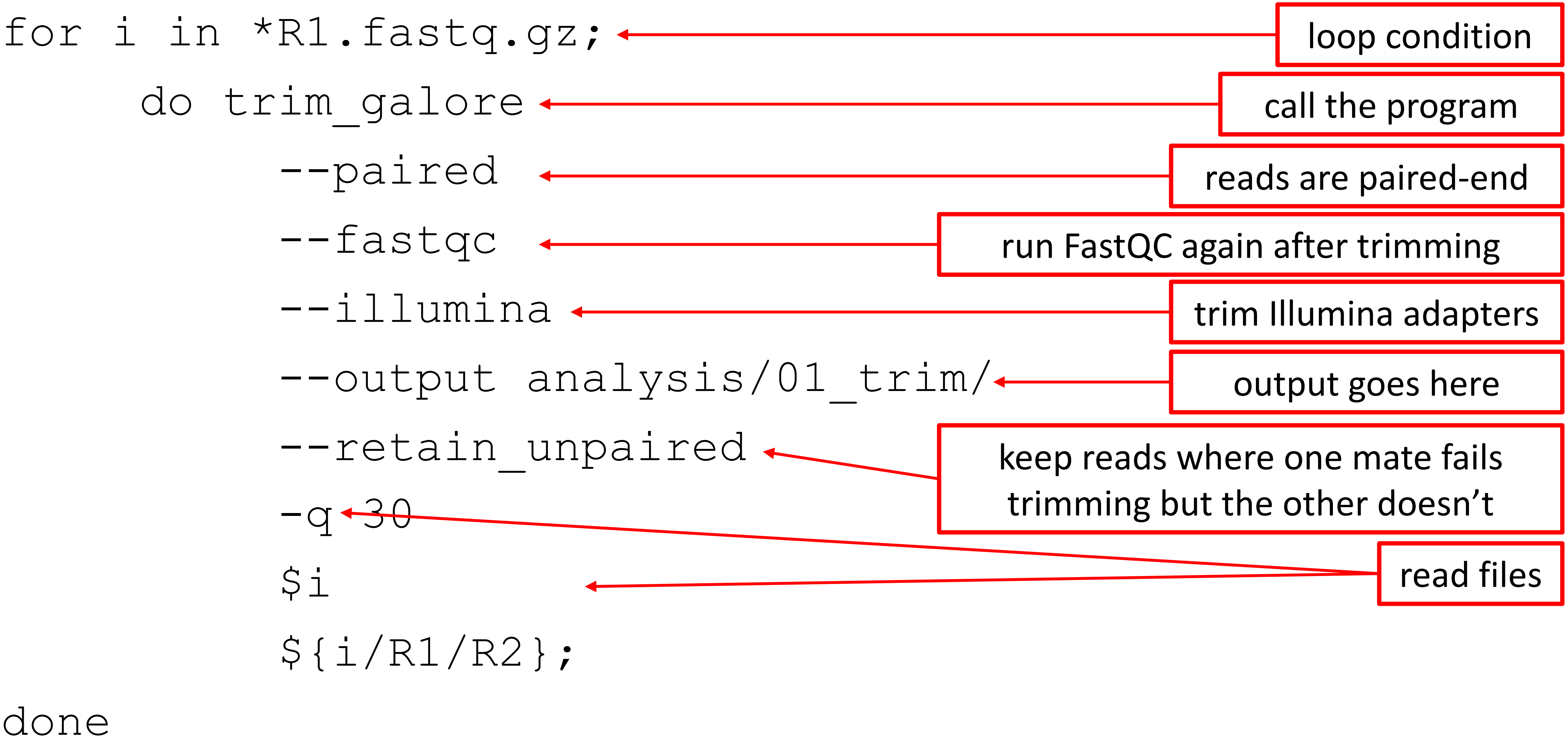
run FastQC again after trimming

trim Illumina adapters

output goes here

keep reads where one mate fails trimming but the other doesn't

Trim Sequences



```
for i in *R1.fastq.gz;
        do trim_galore
                --paired
                --fastqc
                --illumina
                --output analysis/01_trim/
                --retain_unpaired
                -q 30
                $i
                ${i/R1/R2};
done
```

loop condition

call the program

reads are paired-end

run FastQC again after trimming

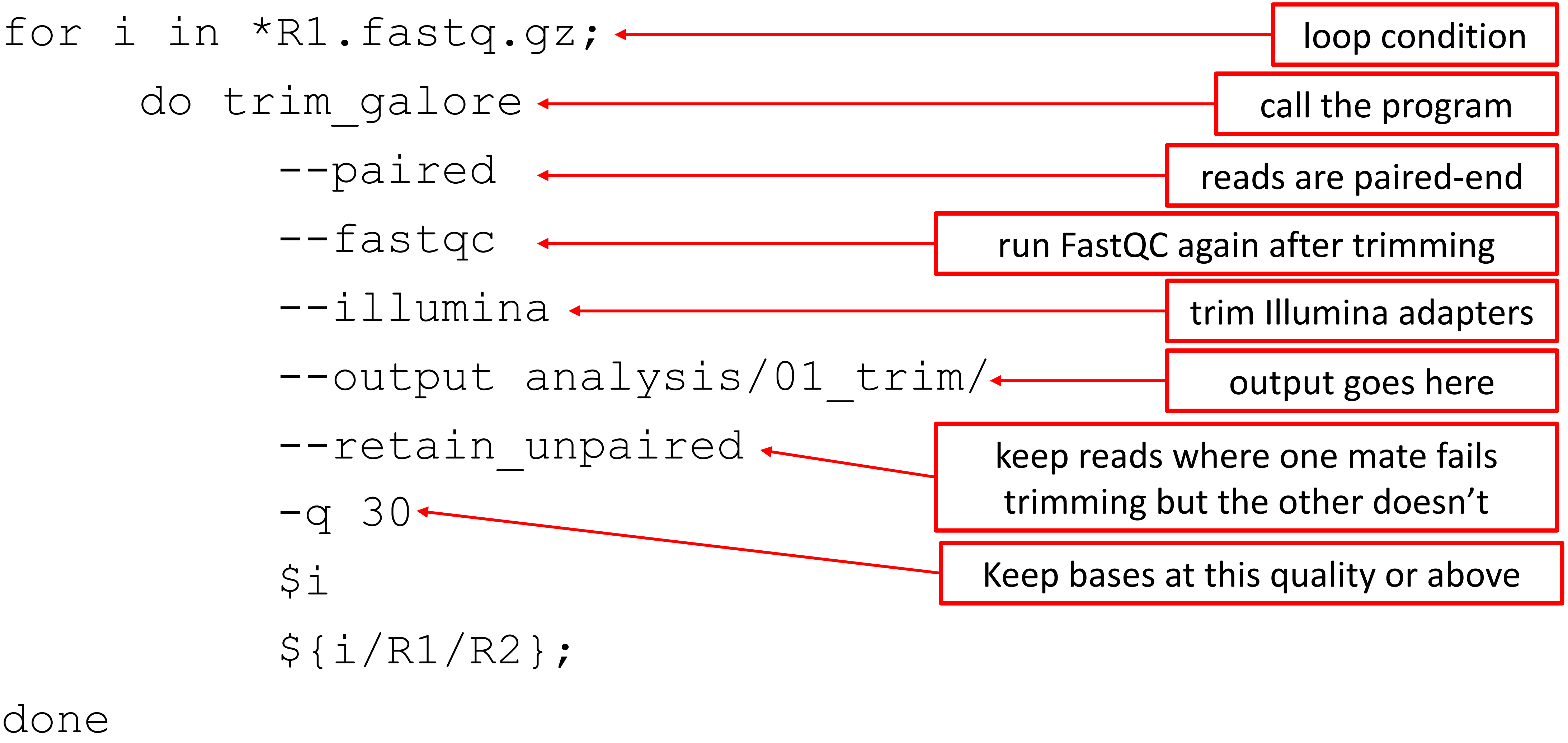
trim Illumina adapters

output goes here

keep reads where one mate fails trimming but the other doesn’t

read files

Trim Sequences



```
for i in *R1.fastq.gz;
     do trim_galore
          --paired
          --fastqc
          --illumina
          --output analysis/01_trim/
          --retain_unpaired
          -q 30
          $i
          ${i/R1/R2};
done
```

loop condition

call the program

reads are paired-end

run FastQC again after trimming

trim Illumina adapters

output goes here

keep reads where one mate fails trimming but the other doesn’t

Keep bases at this quality or above

Trim Sequences

```
for i in *R1.fastq.gz;
        do trim_galore
            --paired
            --fastqc
            --illumina
            --output analysis/01_trim/
            --retain_unpaired
            -q 30
            $i
            ${i/R1/R2};
done
```

loop condition

call the program

reads are paired-end

run FastQC again after trimming

trim Illumina adapters

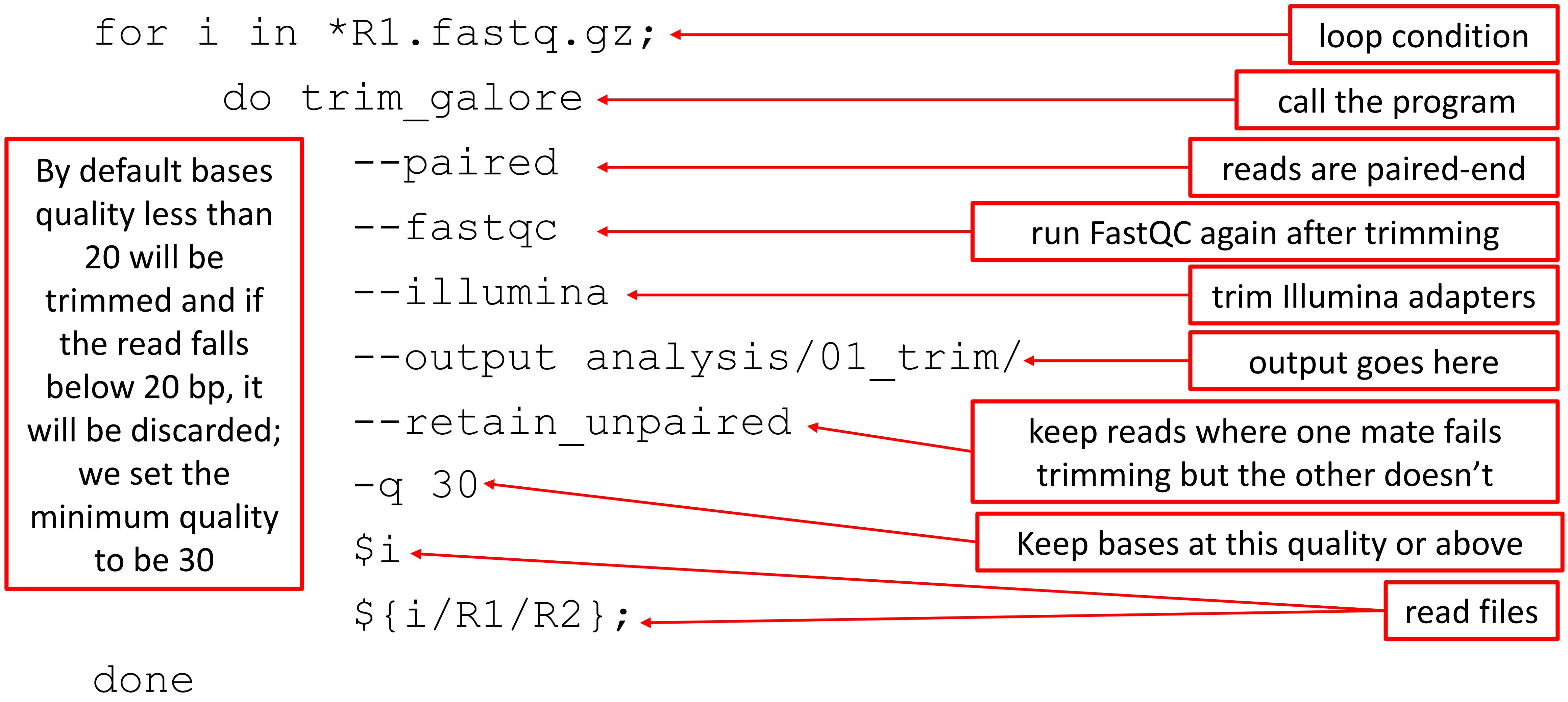
output goes here

keep reads where one mate fails trimming but the other doesn’t

Keep bases at this quality or above

By default bases quality less than 20 will be trimmed and if the read falls below 20 bp, it will be discarded; we set the minimum quality to be 30

Trim Sequences



```
for i in *R1.fastq.gz;
     do trim_galore
          --paired
          --fastqc
          --illumina
          --output analysis/01_trim/
          --retain_unpaired
          -q 30
          $i
          ${i/R1/R2};
done
```

By default bases quality less than 20 will be trimmed and if the read falls below 20 bp, it will be discarded; we set the minimum quality to be 30

loop condition

call the program

reads are paired-end

run FastQC again after trimming

trim Illumina adapters

output goes here

keep reads where one mate fails trimming but the other doesn’t

Keep bases at this quality or above

read files

Trim Command

```
for i in *R1.fastq.gz; do trim_galore
--paired --fastqc --illumina --output analysis/01_trim/
--retain_unpaired –q 30 $i ${i/R1/R2}; done
```