Bioinformatics
LAB 3

Prof.ssa Elisa Ficarra
Prof.ssa Santa Di Cataldo
Eng. Marta Lovino
Eng. Francesco Ponzio
Eng. Emanuele Parisi

Politecnico di Torino
DAUIN
Dept. of Control and Computer Engineering

Organization

Schedule

This week

LAB2 & LAB3

Wednesday, May 27th	11.30 – 13.00
Thursday, May 28th	11.30 – 13.00 + 13.00 – 14.30

Projects

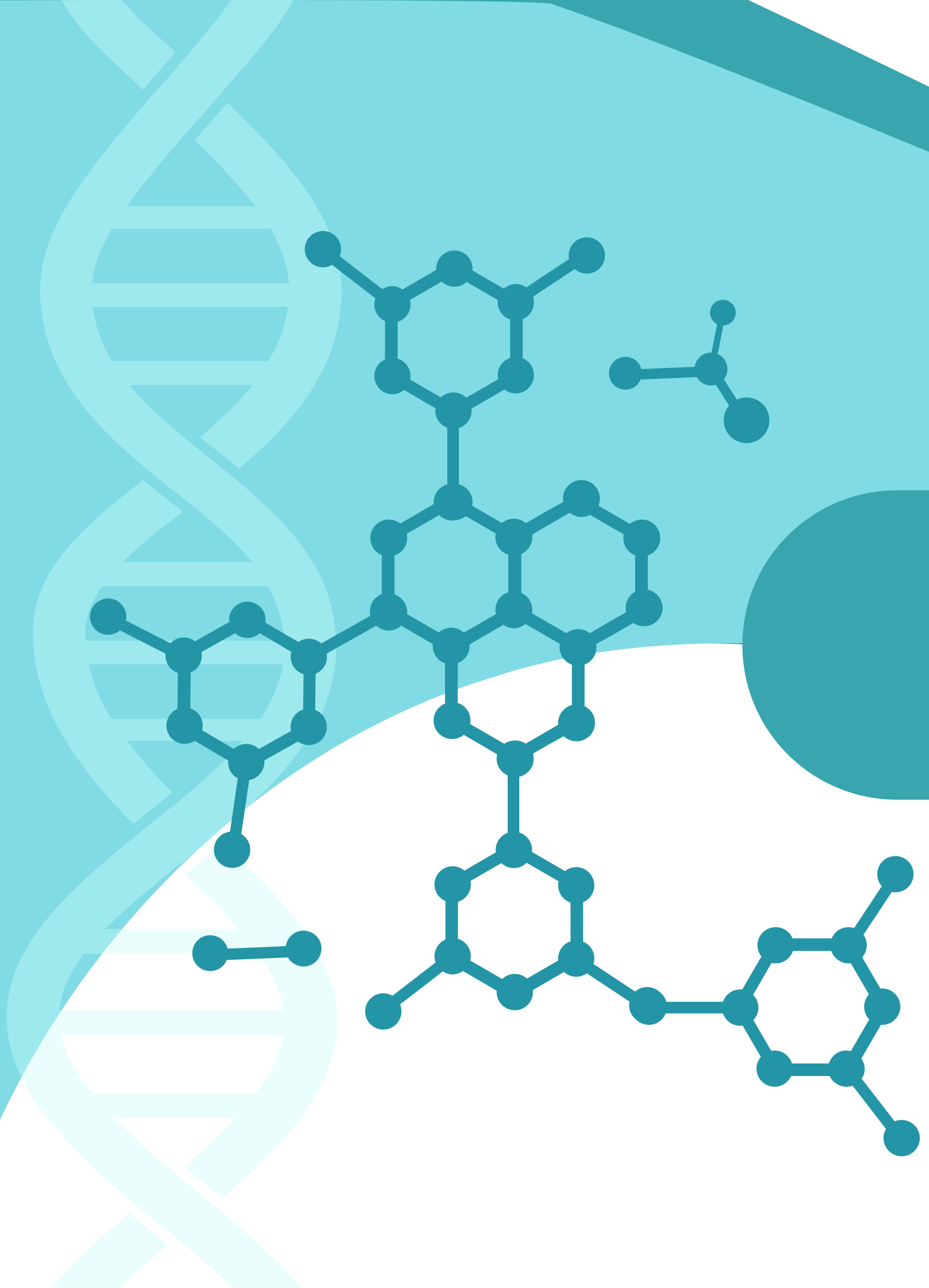
Friday, May 29th	11.30 – 13.00 (genomics) + 13.00 – 14.30 (bioimaging)

Next week

LAB4 & LAB5

Wednesday, June 3rd	11.30 – 13.00 + 13.00 – 14.30
Thursday, June 4th	11.30 – 13.00 + 13.00 – 14.30

Please check the **Teaching Portal** and the **Telegram group** to be updated.



LAB 3 - Goals

GOALS

- Understanding and manipulating different file formats (FASTA, FASTQ, SAM, BAM, GTF, VCF)
- Flags in SAM files
- Samtools, BCFtools examples

File formats – FASTA & FASTQ

FASTA

A sequence record in a FASTA format consists of a single-line description (sequence name), followed by line(s) of sequence data. The first character of the description line is a greater-than (">") symbol.

```
>read_id_0
GGTATGCTTCTGGGGCGGCAGTCGATAGGGCTAGACTCAGGTCCCGTGGC
>read_id_1
CACTGTGGCCCTCTTGGGGGGTGTCCACACGCCGCCCGTCGGCCCCCTCC
>read_id_2
GTTCTGTGGGTACCTCGCGGTTATGGTGTCGGGGGTATCCAAGGCACCCC
```

FASTQ

Similar to FASTA file, but with mapping quality information for each base. Both the sequence letter and quality score are each encoded with a single ASCII character.

```
@ERX288614.1 HWI-ST1362:33:D1J0JACXX:6:1101:1687:2354 length=101
TTTTTCTAGACGGCAGGTCAGGTCCACCACTGACACGTTGGCAGTGGGGACACGGAAGGCCATGCCA
+ERX288614.1 HWI-ST1362:33:D1J0JACXX:6:1101:1687:2354 length=101
<<BFFFF0BBF<0FFFFIBB0BFBF7BF0<<BFB0BFBF<<<0BFBB7BBFFFFFFFFBBB<77<B<
@ERX288614.2 HWI-ST1362:33:D1J0JACXX:6:1101:1519:2446 length=101
CCCTATTCTGCTAGCTTGGGTTTAGTTCTTCTTTTTGTAGGTCCTTTAAAGTGTATAGTTAGGTGAC
+ERX288614.2 HWI-ST1362:33:D1J0JACXX:6:1101:1519:2446 length=101
BBBFFFFFFBFFFFFFFFFIFFIBFFFFIIFFFFFFF<BFF7BFIII<<'BFFBBFFIBF<BB'0<F
```

File formats – SAM

SAM stands for Sequence Alignment/Map format and it is the most common file output format for aligners. It is a TAB-delimited text format consisting of a header section, which is optional, and an alignment section.

```
@SQ     SN:10   LN:133797422
@SQ     SN:18   LN:80373285
@PG     ID:bwa  PN:bwa  VN:0.7.17-r1188 CL:bwa mem
/home/marta/Documents/BIOINFORMATICS/BioInfoCourse/LAB_alignment/tools/bwa_index/bwa_index
mate_1.fq mate_2.fq
ERX288614.1     99      10      55667614        39      12S79M1I9M      =       55667850        3
18      TTTTTCTAGACGGCAGGTCAGGTCCACCACTGACACGTTGGCAGTGGGGACACGGAAGGCCATGCCAGTGAGCTTCCCGTTCAGCTCAGG
GATGACCTTGC   <<BFFFF0BBF<0FFFFIBB0BFBF7BF0<<BFB0BFBF<<<0BFBB7BBFFFFFFFFBBB<77<B<<<<BBBBBBBBBBBBB
B<<BBB<BBBBBB<BBB   NM:i:9   MD:Z:15G7G0T5G2A1A5A24A21       MC:Z:19S82M    AS:i:41 XS:i:51
XA:Z:10,+15093507,101M,10;
ERX288614.1     147     10      55667850        39      19S82M  =       55667614        -
318     AGTCCTTCCACGATACCAAAGTTGTCATGGATGTCCTTGGCCAGGGGTGCTAAGCAGTTGGTGGTGCAGGAGGCATTGCTGATGATCTTG
AGGCTGTTGTC   B<<<BB<<<<<B<B<B<<<B<B<BBBB<<<<77'<7<BBBBBBBBBBBBB<B<BBBB<BBBBB<'0'<BB700<<<BBB0<7BB
B0<BB0<<<BB<B<<<<   NM:i:4   MD:Z:14A39T8C0A17       MC:Z:12S79M1I9M AS:i:62XS:i:53 XA:Z:10,-
91667278,15S83M3S,6;
```

```
chr10
ref    ATTTGACCGCAGCACTTTGACACGCAGCATTTTGGGCCCATTATATACGGCTTA
0based 01234……                          27
1based 1234567….
read0      GACCGCAGCACTTTG
Read1                                    CCGGGTAA
                                         ←
```

read0

0 based coordinate system→4 → + sign forward strand

1 based coordinate system→5 → + sign forward strand

Read1

0 based coordinate system→27 → - sign backward strand

1 based coordinate system→28 → - sign backward strand

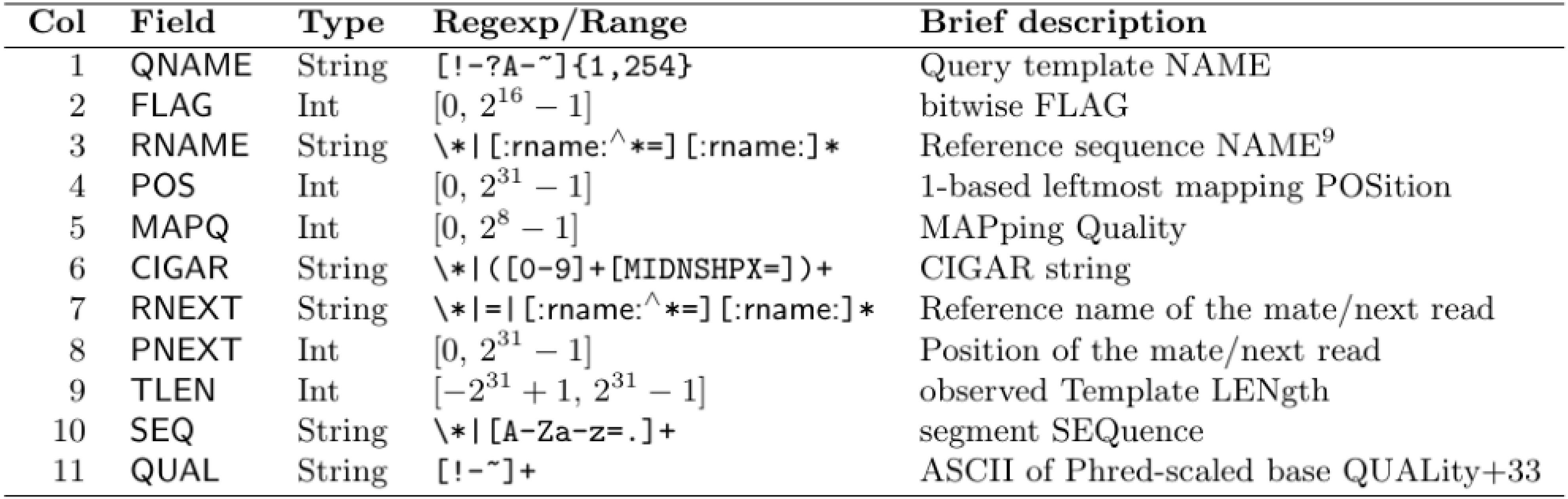
```
chr10
ref    ATTTGACCGCAGCACTTTGACACGCAGCATTTTGGGCCCATTATATACGGCTTA

R0                          ACGCAGCACTTTGGGCC
R1                          ACGCAGCACTTTGGGCC
R2                          ACGCAGCACTTTGGGCC
R4                          ACGCAGCACTTTGGGCC

T → C

ref    ATTTGACCGCAGCACTTTGACACGCAGCATTTTGGGCCCATTATATACGGCTTA

R0                          ACGCAGCACTTTGGGCC
R1                          ACGCAGCAATTTGGGCC
R2                          ACGCAGCATTTTGGGCC
R4                          ACGCAGCACTTTGGGCC

T → ? A, C  <*>
```

File formats – SAM

- Remember that an **aligner can report multiple alignments for the same read**!! Depending from the application, this could be an issue to be fixed.

- Take a look here https://samtools.github.io/hts-specs/SAMv1.pdf for **all details about this file format, paying particular attention to section 1.4 about mandatory fields** (page 6).

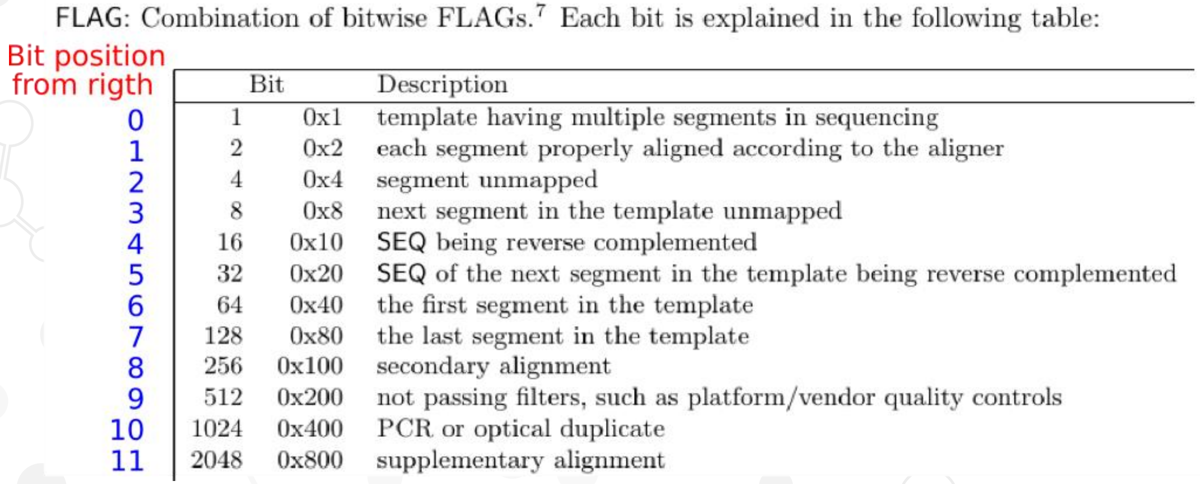
- **N.B. Genome positions in SAM files are in 1-based coordinate system**



Col	Field	Type	Regexp/Range	Brief description
1	QNAME	String	`[!-?A-~]{1,254}`	Query template NAME
2	FLAG	Int	$[0, 2^{16} - 1]$	bitwise FLAG
3	RNAME	String	`\*|[:rname:^*=][:rname:]*`	Reference sequence NAME[9]
4	POS	Int	$[0, 2^{31} - 1]$	1-based leftmost mapping POSition
5	MAPQ	Int	$[0, 2^{8} - 1]$	MAPping Quality
6	CIGAR	String	`\*|([0-9]+[MIDNSHPX=])+`	CIGAR string
7	RNEXT	String	`\*|=|[:rname:^*=][:rname:]*`	Reference name of the mate/next read
8	PNEXT	Int	$[0, 2^{31} - 1]$	Position of the mate/next read
9	TLEN	Int	$[-2^{31} + 1, 2^{31} - 1]$	observed Template LENgth
10	SEQ	String	`\*|[A-Za-z=.]+`	segment SEQuence
11	QUAL	String	`[!-~]+`	ASCII of Phred-scaled base QUALity+33

Flags in SAM file

Flags are used to keep track of alignment information in a compact way and uses 12 bits.



FLAG: Combination of bitwise FLAGs.[7] Each bit is explained in the following table:

Bit position from rigth	Bit		Description
0	1	0x1	template having multiple segments in sequencing
1	2	0x2	each segment properly aligned according to the aligner
2	4	0x4	segment unmapped
3	8	0x8	next segment in the template unmapped
4	16	0x10	SEQ being reverse complemented
5	32	0x20	SEQ of the next segment in the template being reverse complemented
6	64	0x40	the first segment in the template
7	128	0x80	the last segment in the template
8	256	0x100	secondary alignment
9	512	0x200	not passing filters, such as platform/vendor quality controls
10	1024	0x400	PCR or optical duplicate
11	2048	0x800	supplementary alignment

Some examples with FLAGS:

000000000001 --> $2^0 = 1$ --> template having multiple segments in sequencing
000000000010 --> $2^1 = 2$ --> each segment properly aligned according to the aligner
000000000100 --> $2^2 = 4$ --> segment unmapped
000100000000 --> $2^8 = 256$ --> secondary alignment

Bits can be combined:

000000001100 --> $2^2 + 2^3 = 4 + 8 = 12$ --> segment unmapped **and** next segment in the template unmapped

File formats – BAM

- BAM is the **binary version of a SAM file**. This means that BAM files are smaller than SAM files and this property is really helpful especially when we are working with huge files (e.g. a 30 GB SAM file can be compressed into a 17 GB BAM file). However, since BAM files are binary files, they are **not human readable**.

- N.B. Genome positions in BAM files are **in 0-based coordinate system**.

File formats – GTF

The Gene transfer format (GTF) is a file format used to **hold information about gene structure of a reference genome**. It is a tab-delimited text format based on the general feature format (GFF). This file format is really helpful when you want to know which biological feature (gene, exon,CDS, ...) is present in which genome positions. Genome positions in GTF files are in **1-based coordinate system**

```
1       havana  gene    11869   14409   .       +       .       gene_id "ENSG00000223972";
gene_version "5"; gene_name "DDX11L1"; gene_source "havana"; gene_biotype
"transcribed_unprocessed_pseudogene"; havana_gene "OTTHUMG00000000961"; havana_gene_version "2";
1       havana  transcript      11869   14409   .       +       .       gene_id "ENSG00000223972";
gene_version "5"; transcript_id "ENST00000456328"; transcript_version "2"; gene_name "DDX11L1";
gene_source "havana"; gene_biotype "transcribed_unprocessed_pseudogene"; havana_gene
"OTTHUMG00000000961"; havana_gene_version "2"; transcript_name "DDX11L1-002"; transcript_source
"havana"; transcript_biotype "processed_transcript"; havana_transcript "OTTHUMT00000362751";
havana_transcript_version "1"; tag "basic"; transcript_support_level "1";
1       havana  exon    11869   12227   .       +       .       gene_id "ENSG00000223972";
gene_version "5"; transcript_id "ENST00000456328"; transcript_version "2"; exon_number "1"; gene_name
"DDX11L1"; gene_source "havana"; gene_biotype "transcribed_unprocessed_pseudogene"; havana_gene
"OTTHUMG00000000961"; havana_gene_version "2"; transcript_name "DDX11L1-002"; transcript_source
"havana"; transcript_biotype "processed_transcript"; havana_transcript "OTTHUMT00000362751";
havana_transcript_version "1"; exon_id "ENSE00002234944"; exon_version "1"; tag "basic";
transcript_support_level "1";
"1";
```

File formats – VCF

VCF is a text file format (most likely stored in a compressed manner) **and it holds information about reads**. It contains meta-information lines, a header line, and then data lines each containing information about a position in the genome. The format also has the ability to contain genotype information on samples for each position.

```
#CHROM    POS     ID     REF    ALT    QUAL    FILTER    INFO    FORMAT    sorted.bam
10     48636     .     T     <*>     0     .     DP=1;I16=0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0;QS=0,0;MQ0F=0     PL     0,0,0
10     48637     .     G     <*>     0     .     DP=2;I16=0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0;QS=0,0;MQ0F=0     PL     0,0,0
10     48638     .     A     <*>     0     .     DP=2;I16=1,0,0,0,70,4900,0,0,60,3600,0,0,2,4,0,0;QS=1,0;MQ0F=0     PL
0,3,60
10     48639     .     A     <*>     0     .     DP=2;I16=1,0,0,0,74,5476,0,0,60,3600,0,0,3,9,0,0;QS=1,0;MQ0F=0     PL
0,3,60
10     48640     .     G     <*>     0     .     DP=2;I16=1,0,0,0,70,4900,0,0,60,3600,0,0,4,16,0,0;QS=1,0;MQ0F=0     PL
0,3,60
10     48641     .     A     <*>     0     .     DP=2;I16=1,0,0,0,70,4900,0,0,60,3600,0,0,5,25,0,0;QS=1,0;MQ0F=0     PL
0,3,60
10     48642     .     C     T,<*>     0     .     DP=2;I16=0,0,1,0,0,0,70,4900,0,0,60,3600,0,0,6,36;QS=0,1,0;SGB=-
0.379885;MQ0F=0     PL     60,3,0,60,3,60
```

Tools and data

INSTALL SAMTOOLS, BCFTOOLS and download human gtf file

```
conda activate Bioinfo_labs (or source activate Bioinfo_labs)
conda install -c bioconda samtools
conda install -c bioconda bcftools
```

Download **Homo_sapiens.GRCh38.95.gtf.gz** file from ftp://ftp.ensembl.org/pub/release-95/gtf/homo_sapiens/Homo_sapiens.GRCh38.95.gtf.gz , move to the correct folder and extract it:

```
gunzip –d Homo_sapiens.GRCh38.95.gtf.gz
```

Samtools, really basic usage

SAM Tools provide various utilities for manipulating alignments in the SAM format, including sorting, merging, indexing and generating alignments in a per-position format.

SAM/BAM conversions

As previously pointed out, BAM format is the binary version of a SAM file. The conversion can be performed using samtools:

```
samtools view -S -b my.sam > my.bam
```

BAM sorting

When multiple selections have to be performed onto a huge file, it is convenient to sort that file according to certain criteria (e.g. genomic region) in order to search for the required information in a faster way. For BAM sorting you can use:

```
samtools sort my.bam > my-sorted.bam
```

Samtools to filter SAM files using FLAGS

Among the many potentials of samtools view there is that of filtering the reads using the FLAG field in SAM files. To have an overall idea of how samtools view works open your terminal, activate Bioinfo_labs environment and type

```
samtools view
```

to get its manual page.

Take a look at -f and -F options. These options allow us to filter reads in a SAM file following a combination of criteria relying on alignment flags.

-f INT only include reads with **all of the FLAGs in INT present**

-F INT only include reads with **none of the FLAGS in INT present**

E.g. to obtain a SAM file with no unmapped reads and no secondary alignments we can exploit bit in position 2 (2^2=4) and bit in position 8 (2^8=256).

```
samtools view –F 260 bwa_out.sam > unique_aligned.sam
```

`-F 260` ($2^2 + 2^8$) means that in the final SAM file will be printed only reads for which bit number 2 or bit number 8 is not set to 1.

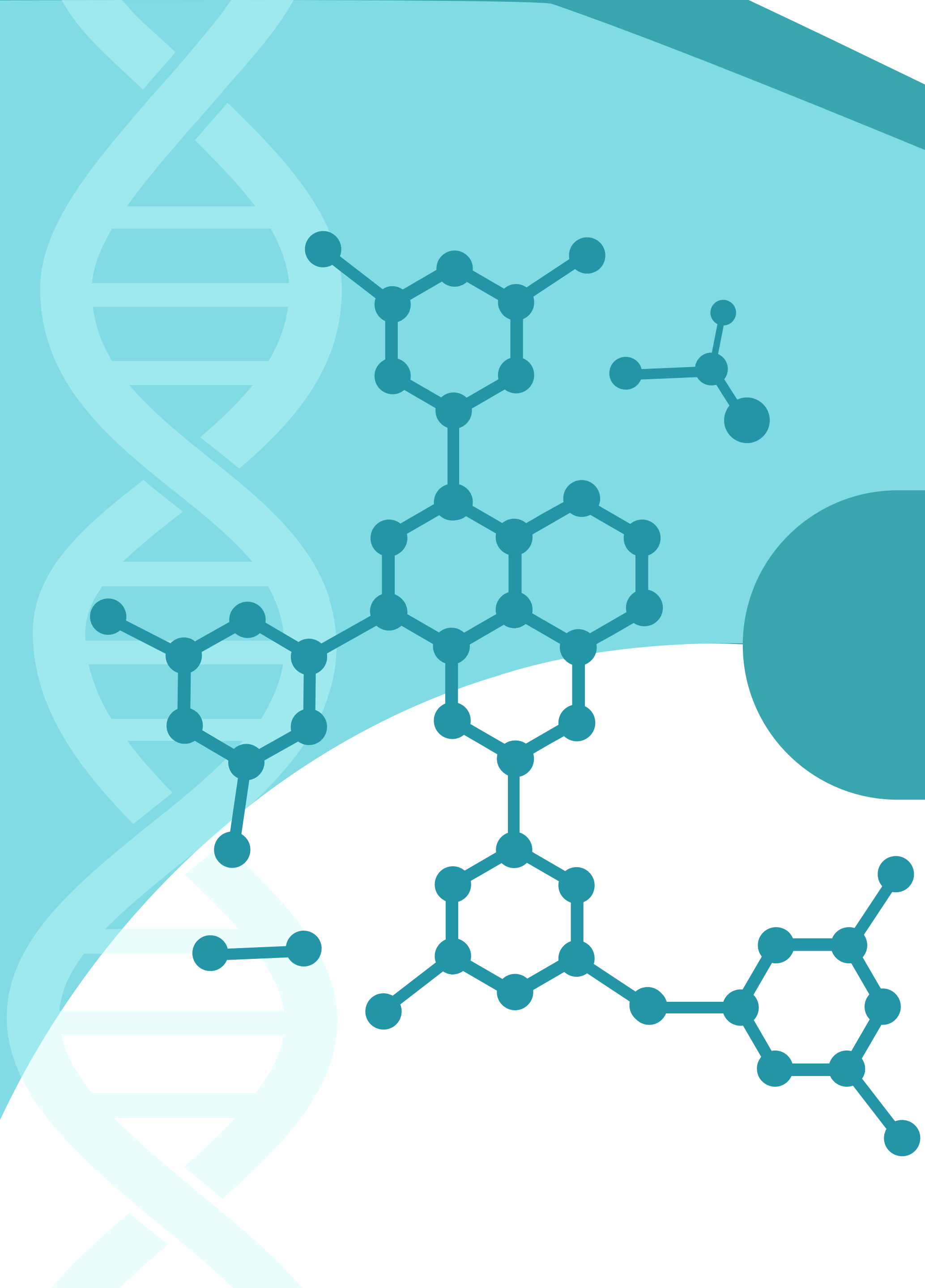
Bcftools

BCFtools is a set of utilities that manipulate variant calls in the Variant Call Format (VCF) and its binary counterpart BCF.

BCFTOOLS to create VCF file

Use bcftool to convert a sorted BAM file into VCF:

```
bcftools mpileup --fasta-ref reference_chr10_chr18.fa sorted.bam > sorted.vcf
```



LAB 3 - Assignments

LAB3 - Assignments

- Search for SNP and deletions in a sample
- Raw read count for protein coding genes

Questions?

Remember:
no question is
stupid

LET’S START PRACTICING!!