General Assignment Information

- Submit an electronic copy of your assignment via LEARN. If you run out of your allotted late submission time during the term (72 hours), your submission will incur a 5% penalty for every rounded-up hour past the deadline.
  For example, an assignment submitted 5 hours and 15 minutes late will receive a penalty of ceiling(5.25)*5% = 30%.
- Submissions are not accepted after 72 hours past the due date.
- For each question, submit a `.pdf` report describing your solution and the files necessary to run your program.
- You will lose marks if your code is unreadable, sloppy, or inefficient.
- Submit all your files in a single compressed file (.zip, .tar etc.)
- You **must** include the specific Python version used to compile their program (e.g. output of `python --version` command). You should assume that the system used for testing will not run any package installation software.
- The filename should include your username and/or student ID.

[10] 1. **Edit distance calculation**. Given two strings v and w, of lengths n and m, respectively. The edit distance $\mathrm{d_E}(v, w)$ is defined as the minimum number of edit operations (substitution, insertion, or deletion) of single symbols needed to transform v into w. Design an algorithm and write a program to calculate the edit distance between two DNA strings.

Input: A single FASTA file with two nucleotide strings v and w of at most 1000 nucleotides each.

```
>seq1
ACGTGCGTCGCA
>seq2
ACTGCCGCGCA
```

Output: The edit distance $\mathrm{d_E}(v, w)$.

```
python edit_distance.py --input=sample_fasta.py
3
```

[20] 2. **The fitting alignment problem**. Given two strings v and w, the fit alignment problem asks to find a substring v' of v, such that the global sequence alignment score between v' and w is maximized. Construct a highest-scoring fitting alignment between two strings.

Input: A FASTA format with two DNA strings v and v, where v has a length of at most 10 kbp and w has a length of at most 1 kbp and $\text{len}(w) \leq \text{len}(v)$.

```
>seq1
GTAGGCTTAAGGTTA

>seq2
TAGATA
```

Output: The maximum score of a fitting alignment of v and w, followed by a fitting alignment achieving this maximum score. Matches count +1 towards the overall score and both the mismatch and indel penalties are equal to 1. **If multiple fitting alignments achieving the maximum score exist, you may return any one.**

```
python fit_alignment.py --input=sample_fasta.fna
2

TAGGCTTA
TAGA--TA
```

[20] 3. **Identifying viral strands.** You will be given a FASTA file with known full viral genomes which might be related to a viral variant detected in a patient. In this question, you will select and use the alignment algorithm that is more appropriate to find which variant it is likely from. **Note: You can assume that the size of the viral genomes will be under 32 kbp and the size of the patient sample will be under 1 kbp.**

Input: You will be given one FASTA file `candidates.fna` containing the candidate viral genomes. You will also be provided with a FASTA file `patient.fna` containing the new partial sequence obtained from a patient.
Output: The sequence ID corresponding to the candidate sequence that best aligns to the patient sequence.

```
python search_variant.py --db=candidates.fna --query=patient.fna
KY112480.1
```