Submission Guidelines

Please submit your solution to all five challenges via email to geral@lpb.pt. The solution to each challenge should be contained in a `.zip` file (so, your submission email should contain 5 `.zip` files).

Each `.zip` file must be named using the following format:

```
challenge_<challenge_number>_<team_name>.zip
```

For example, a submission for challenge 1 by "Team Genome Gurus" should be named:

```
challenge_1_Team_Genome_Gurus.zip
```

Each `.zip` file should include the following:

- The script containing your solution, or the source code to compile a binary that solves your problem (which can *optionally* be inside its own directory).
- A `README.md` file with the following format:
    - **First line**: Your team's name.
    - **Second line**: Leave blank.
    - **Third line**: Instructions on how to install any required dependencies and/or compile your project.
    - **Fourth line**: Instructions on how to run your program from the command line.

Example `README.md` file:

```
Team_Genome_Gurus

python3 -m pip install -r requirements.txt
python3 src/my_script.py "Homo sapiens" output_genes.txt
```

Output Format

Your script **must** save the output to a file. The output file should follow the naming format:

```
output_<challenge_number>.txt
```

For example, the output file for challenge I should be:

```
output_I.txt
```

The rest of the `README.md` file documentation is entirely up to you.

Evaluation Criteria:

1. **Compliance**: Adherence to input/output guidelines.
2. **Functionality**: Correct output.
3. **Efficiency**: Reasonable query handling and execution time.
4. **Code Quality**: Clarity, structure, and documentation.

GLHF!

Portuguese League of Bioinformatics 2025

Challenge I: Mutation Rate Estimation with Bootstrapping
Barbara McClintock, a young population geneticist is studying the mutation rate of a new bacterial species over multiple generations. The number of mutations observed in a sample of individuals has been recorded, but due to sampling variability, a single estimate of the mutation rate may not be reliable. To quantify uncertainty, Barbara decided to apply bootstrap resampling to generate confidence intervals for the mutation rate.

The mutation rate per site per generation (MR) is calculated using:
MR = Total Mutations Observed / (Genome Length × Total Number of Generations)
Input:
A file containing:

- A vector "mut" of comma separated integers where each value corresponds to the total number of mutations observed in each sampled individual (independent observations).
- An integer "l", representing the genome length in base pairs.
- An integer "g", representing the total number of generations.
- An integer "b", representing the number of bootstrap replicates requested.

Output:
A file containing:

- A floating-point number representing the observed mean mutation rate (rounded to two decimal places).
- A 95% confidence interval for the mutation rate (corresponding to the 2.5th and 97.5th percentiles from the distribution of bootstrap mean rates), given as two floating-point numbers in square brackets, with one decimal place in scientific format.

Example input file:

```
mut = 5,7,4,6,8,5,3,9,4,6
l = 5000
g = 10
b = 1000
```

Example output file:

```
Mean mutation rate observed: 1.14e-03
95% Confidence Interval: [9.2e-04, 1.4e-03]
```

Portuguese League of Bioinformatics 2025

Challenge II: Gene List Extraction from NCBI Entrez API
In this challenge, your team will develop a program that queries the NCBI Entrez API to retrieve a list of gene names for a given organism.
The challenge consists in creating a program that queries the NCBI Entrez database to list all gene names for a provided organism. The program should accept two command-line arguments:

- Organism name
- Output file path

Results should be written to an output file, one gene per line.

Input:
Two command line arguments:

- A species name
- An output file path

Output:
A file containing:

- Genes names, one per line

Example Input command:

```
./my_program "Homo sapiens" output_II.txt
```

Example Output file:

```
Cytb
COXI
NAD4
...etc...
```

Portuguese League of Bioinformatics 2025

Challenge III:
You are given a list of UniProt proteins and need to find their molecular functions, biological processes and associated cellular components using a gene annotation file based on the Gene Ontology (https://geneontology.org/).

1. Get the HUMAN GO annotation file goa_human.gaf.gz available at: ftp://ftp.ebi.ac.uk/pub/databases/GO/goa/ in the HUMAN folder.
2. Read the README file in the HUMAN folder to understand the file structure
3. Unzip the GAF file.
4. Process the file to extract for each UniProt ID the corresponding GO terms. Ignore annotations that have the evidence code IEA. Remove duplicates.
5. Find the top 10 GO terms for each GO branch (molecular function, biological process and cellular component) ranked by the number of UniProt proteins from the set that are associated to each of the GO terms.
6. Print them following the example of output.

Input:
A file containing a list of UniProt accession numbers (i.e., ids)
Output:
A file containing the top 10 GO terms for each GO branch

Input file example:

```
P35968
P07900
P31749
Q504U0
P04637
P01116
P15056
P38398
P21860
P07949
```

Output file example:

```
UniProtID: P35968
Biological Process: GO:0006457, GO:0006237, GO:0001431, …
Molecular Function: GO:0006457, GO:0006237, GO:0001431, …
Cellular Component: GO:0006457, GO:0006237, GO:0001431, …
```

Portuguese League of Bioinformatics 2025

Challenge IV:
You are a bioinformatician studying a bacterial gene that is suspected to contribute to pathogenicity. You have collected pre-aligned protein sequences from multiple strains and categorized them into two groups: pathogenic strains and non-pathogenic strains. Your goal is to identify amino acid mutations that are unique to the pathogenic group, as these might be responsible for the virulence of the bacteria.
You are given pre-aligned protein sequences from different bacterial strains and you must develop a program to compare the sequences position-by-position and detect amino acid mutations that are exclusive to the pathogenic strains (i.e., not found in non-pathogenic strains) and report the mutation positions, the amino acid in pathogenic strains, and the amino acid in non-pathogenic strains.

Input:
A pre-aligned FASTA-formatted file in which each sequence has a header line with the label indicating whether the strain is "pathogenic" or "non_pathogenic". The sequence is a pre-aligned protein sequence.

Output:
A list of positions in the sequence where pathogenic and non-pathogenic strains differ. For each such position, the output should include the position number (1-based index), the amino acids found in pathogenic strains and in non-pathogenic strains at that position (one per line).

Example input file:

```
>pathogenic|strain_A
MKTAYIAKQRQISFVKSHFSRQDILD
>pathogenic|strain_B
MKTAYIAKQKQISFVKTHFSRQDILD
>non_pathogenic|strain_C
MKTAYIAKQGQISFVKSHFSRQDILD
>non_pathogenic|strain_D
MKTAYIAKQGQISFVKSHFSRQDILD
```

Example output file:

```
Position 10: Pathogenic -> {'R', 'K'}, Non-Pathogenic -> {'G'}
```

Portuguese League of Bioinformatics 2025

Challenge V:
A bioinformatics researcher wants to develop a program capable of analyzing and comparing aptamer sequences for potential use in environmental microorganism detection. The program should focus on identifying conserved motifs and calculating the GC content, which are important factors in aptamer stability and binding affinity.

Input:
A file containing two aptamer sequences, seq1 and seq2, of potentially different lengths.

Output:
A file containing:
1. The length of each sequence
2. The GC content (as a percentage) of each sequence
3. The longest common substring between the two sequences

Example Input file:

```
seq1 ATCGGGAATTCCCGATATCG
seq2 GGGAATTCCCTATCGGCTA
```

Example Output:

```
seq1 length: 20
seq2 length: 19
seq1 GC content: 50.0%
seq2 GC content: 52.6%
Longest common substring: GGGAATTCCC
```