

Programming BINP16 Home Exam, 2025-10-29, 10 points

Coding rules

- You should document and comment on your code, use descriptive functions and variable names, and follow the code style recommendations that were made during the course; all of this will be taken into account when grading the exam.
- Your script is meant to be used by other people. Therefore, the user should be informed about how the script should be run by printing usage information when the script is called with the right parameters.
- You are only allowed to use modules that are part of the Python standard library, like sys, os, re, math, random, argparse and string.
- You can also use the standard modules subprocess, itertools, and external packages such as Pandas and Biopython.
- You are NOT ALLOWED to use non-standard modules (e.g., modules from github).
- Remember to double-check your results using bash if possible!
- Submit the exam by: **31/10 13:00** into [Take home exam](#). Late submissions will be penalized by 1 point per minute (rounded).

Exam rules

You can use your private computer.

Any communication (including electronic communication) and sharing of code between students and between students and external persons are **strictly forbidden**.

Any access to email, Google Colab, or cell phone is **strictly forbidden**.

You are allowed to use the information on hard drives in your notes and on the internet.

You are NOT ALLOWED to use ChatGPT or other language-based models or Google the question.

For each question, one or more files should be submitted:

- 1) The programs should be named, named 1.sh, 2a.py, 2b.py etc., based on the question number.
- 2) The input file (if relevant), named 1_input_<name>.txt
- 3) The output file (if relevant), named 1_output_<name>.txt. If no output file is requested, state the answer in a text file (e.g., 1_answer.txt).

Compress all your solutions when you are done as a single zip file and **upload them to Canvas**.

The exam is available in the Canvas folder [Home Exam](#). We provided input data. If more data are required, you need to create them yourself. If you have any questions, please email the TA: Mirjam Müller mirjam.muller@med.lu.se

All the questions are mandatory.

Bash section – 3 points

Question 1 (3 pts)

- **Setup:** Students must run `manage_examples.py` first to populate the `examples/` directory with FASTA files.
 - **Scoring:** 3 tasks × 1 point each. A task earns 1 point only if both the logic and the output are correct (otherwise 0).
 - **Tools:** A short script or a one-liner is acceptable for each task.
1. **(1 pt):** List all `seq_*_chain*.fasta` files in the `examples/` directory, sorted by filename.
 2. **(1 pt):** Count the total number of FASTA files matching the pattern `seq_*_chain*.fasta`.
 3. **(1 pt):** Write a Bash script that loops over all `seq_*_chain*.fasta` files in `examples/`. For each file, extract all protein sequences (ignore headers starting with `>`), compute each sequence length (amino acids), and output the **combined total length across all files**.

Submit your answers as `MyNameQ1.sh`

Python section – 6 points

Question 2 (3 pts)

Write a Python function `gc_content(sequence, window_size)` that calculates the GC content (percentage of G and C nucleotides) within a sliding window (sliding 1 position each time) across a DNA sequence (read from an input file). The function should return a list of GC content percentages for each window.

Example:

```
sequence = "ATGCGCGCTATGCATCGTAGC"
window_size = 5
```

Output:

```
[60.0, 80.0, 60.0, 60.0, ...]
```

Question 3 (3 pt)

Read a DNA sequence from a file. Then, write a Python function `find_motif(sequence, motif)` that takes in the input DNA filename (`sequence`) and a short motif (`motif`) as input from the user and returns the starting positions (1-based indexing) of all occurrences of the motif within the sequence. Then, plot a simple graph showing the positions of the motif within the sequence

For example:

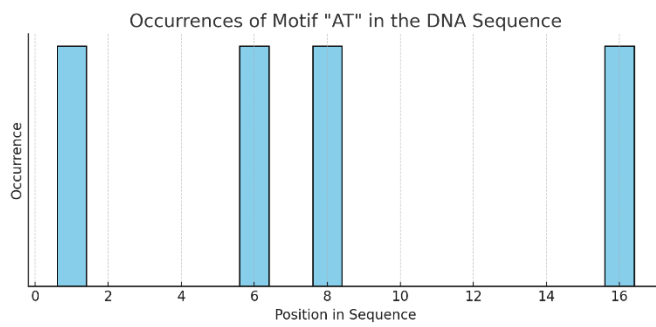
DNA sequence file would have: "**ATGCGATATCGTACGATCG**"

The user will input this motif = "at"

The expected output is:

[1, 7, 13, 17]

The graph is:



Good luck!

Eran