

LABORATORY REPORT #6

GEORGESCU Mihai-Antonio, 1242EA
Bioinformatics, 4th year 1st semester, 2025-2026

lab6_1.py

The screenshot shows the Spyder Python IDE interface. The code editor window displays `lab6_1.py` with the following content:

```
1  # -*- coding: utf-8 -*-
2
3  # Created on Thu Nov  6 09:56:54 2025
4
5  # @author: Antonio
6
7
8  import random
9  import matplotlib.pyplot as plt
10 import numpy as np
11 import re
12
13 # --- Step 1: Parse the FASTA file ---
14 # Assumes the file 'covid.fasta' is in the same directory
15 file_path = "covid.fasta"
16
17 sequence_lines = []
18 full_genome = ""
19 genome_length = 0
20
21 try:
22     with open(file_path, 'r') as f:
23         for line in f:
24             # Skip the header line
25             if not line.startswith('>'):
26                 sequence_lines.append(line.strip())
27
28     # Join all sequence lines into one large string
29     full_genome = "\n".join(sequence_lines)
30     genome_length = len(full_genome)
31
32     if genome_length == 0:
33         print("Warning: Genome sequence is empty. Check FASTA file.")
34         # Use a fallback length if parsing failed
35         genome_length = 30000
36
37 except Exception as e:
38     print(f"Error reading file: {e}. Using a dummy length.")
39     # Fallback length in case of file read error
40     genome_length = 30000
```

The Variable Explorer window on the right shows the following variables:

Name	Type	Size	Value
ax	axes._axes.Axes	1	Axes object of matplotlib.axes._axes module
band_size	int	1	500
f	TextIOWrapper	1	TextIOWrapper object of _io module
fig	Figure.Figure	1	Figure object of matplotlib.figure module
file_path	str	11	covid.fasta
frag_len	int	1	151
fragment_lengths	list	10	[2184, 2058, 1684, 1488, 1172, 1110, 541, 329, 211, 151]
full_genome	str	29903	ATTTAAAGGTTTACCTCCAGGTAAACAAACCAACTTCGATCTTGATGATCTCTCTCAA...
genome_length	int	1	29903
label_text	str	8	500 bp -
ladder_bands	list	14	[10000, 8000, 6000, 5000, 4000, 3000, 2500, 2000, 1500, 1000, ...]
ladder_labels	dict	3	{3000: '3000 bp -', 1500: '1500 bp -', 500: '500 bp -'}

The IPython console shows the command run and its output:

```
In [1]: %runfile C:/Users/Antonio/Desktop/bioinfo/lab6_1.py --wdir
Parsed genome. Total length: 29903 bp
Generated 10 random fragment lengths (bp): [2184, 2058, 1684, 1488, 1172, 1110, 541, 329, 211, 151]
Generated image: gel_electrophoresis_simulation.png
```

The system tray at the bottom indicates the date as 11/6/2025.

National University of Science and Technology POLITEHNICA of Bucharest
Faculty of Engineering in Foreign Languages

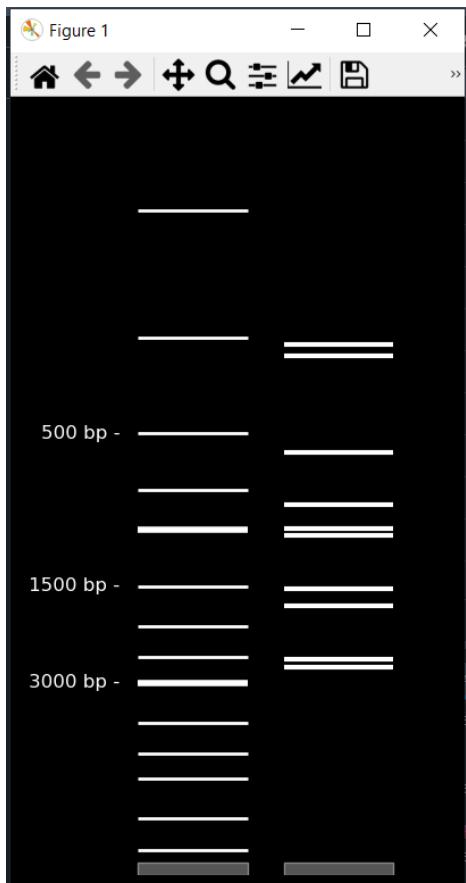
The screenshot shows the Spyder Python IDE interface. On the left, the code editor displays `lab6_1.py` with the following content:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Nov  6 09:56:54 2025
4 @author: Antonio
5 """
6
7 import random
8 import matplotlib.pyplot as plt
9 import numpy as np
10 import re
11
12 # --- Step 1: Parse the FASTA file ---
13 # Assumes the file 'covid.fasta' is in the same directory
14 file_path = 'covid.fasta'
15
16 sequence_lines = []
17 full_genome = ""
18 genome_length = 0
19
20 try:
21     with open(file_path, 'r') as f:
22         for line in f:
23             # Skip the header line
24             if not line.startswith('>'):
25                 sequence_lines.append(line.strip())
26
27     # Join all sequence lines into one large string
28     full_genome = ''.join(sequence_lines)
29     genome_length = len(full_genome)
30
31     if genome_length == 0:
32         print("Warning: Genome sequence is empty. Check FASTA file.")
33     # Use fallback length if parsing failed
34     genome_length = 30000
35
36 except Exception as e:
37     print(f"Error reading file: {e}. Using a dummy length.")
38     # Fallback length in case of file read error
39     genome_length = 30000
40
```

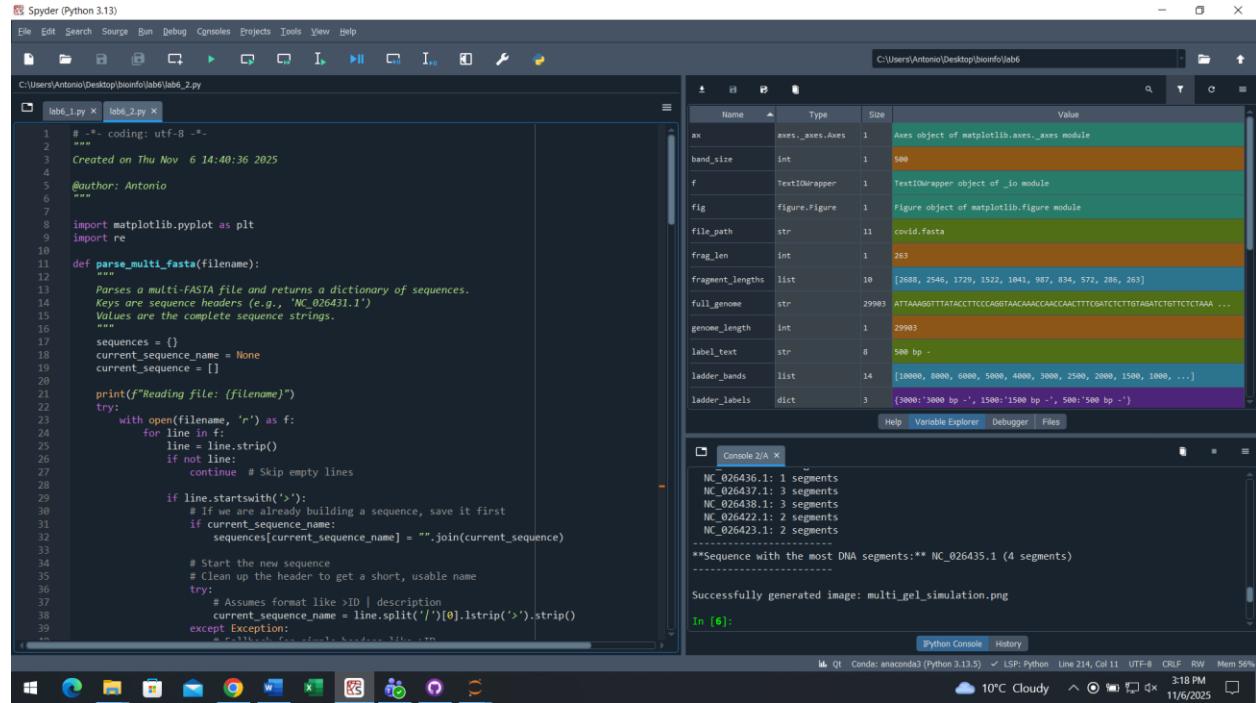
On the right, the Variable Explorer shows the following data:

Type	Size	Value
axes_	Axes	1 Axes object of matplotlib.axes.Axes module
int	1	500
TextIOWrapper	1	TextIOWrapper object of _io module
Figure_Figure	1	Figure object of matplotlib.figure module
str	11	covid.fasta
int	1	263
list	10	[2688, 2546, 1729, 1522, 1041, 987, 834, 572, 286, 263]
str	29903	ATAAAGGTTTATACCTCCAGATAACAAACCAACTTCGATCTCTGTAGATCTGTCCTCAA...
int	1	29903
str	8	500 bp -
list	14	[18000, 8000, 6000, 5000, 4000, 3000, 2500, 2000, 1500, 1000, ...]
dict	3	{3000:'3000 bp -', 1500:'1500 bp -', 500:'500 bp -'}

The central area displays a gel electrophoresis simulation plot titled "Figure 1". The plot shows horizontal bands of varying intensities across three lanes. The lanes are labeled "500 bp", "1500 bp", and "3000 bp" from top to bottom. The plot has a black background with white bands.



lab6_2.py



```

# -*- coding: utf-8 -*-
Created on Thu Nov  6 14:40:36 2025
@author: Antonio
"""

import matplotlib.pyplot as plt
import re

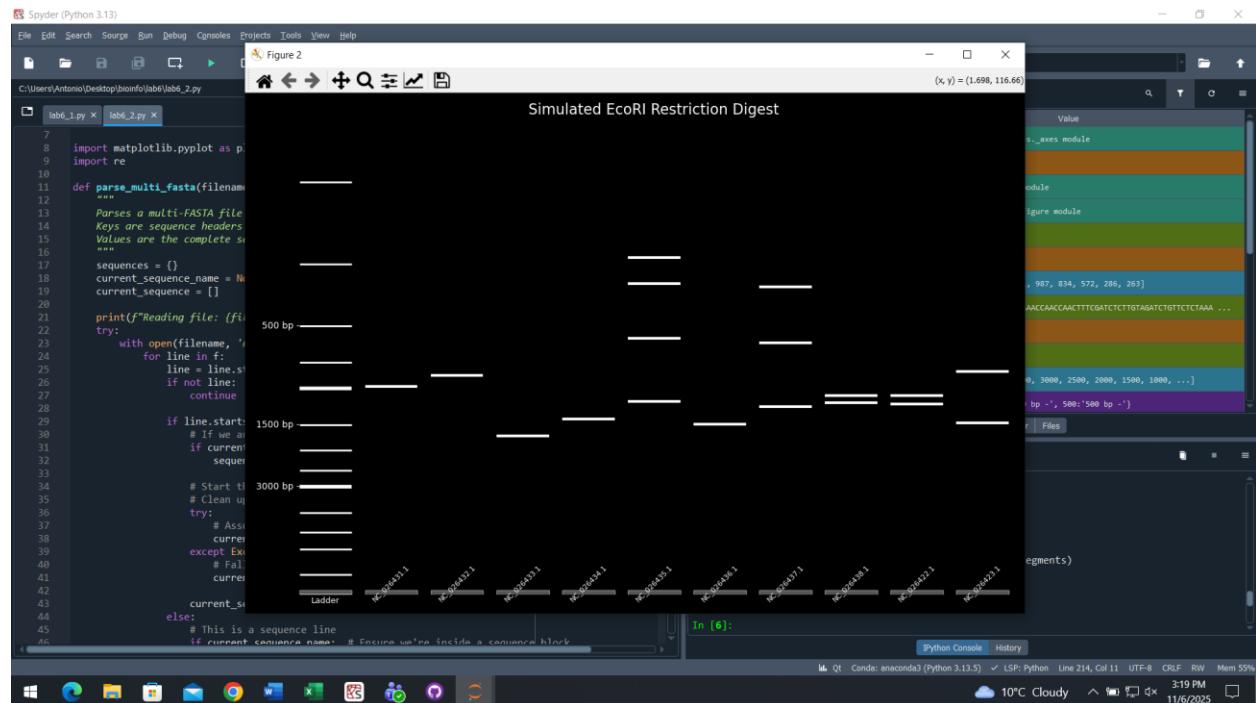
def parse_multi_fasta(filename):
    """
    Parses a multi-FASTA file and returns a dictionary of sequences.
    Keys are sequence headers (e.g., 'NC_026431.1')
    Values are the complete sequence strings.
    """
    sequences = {}
    current_sequence_name = None
    current_sequence = []
    print(f"Reading file: {filename}")
    try:
        with open(filename, 'r') as f:
            for line in f:
                line = line.strip()
                if not line:
                    continue # Skip empty lines

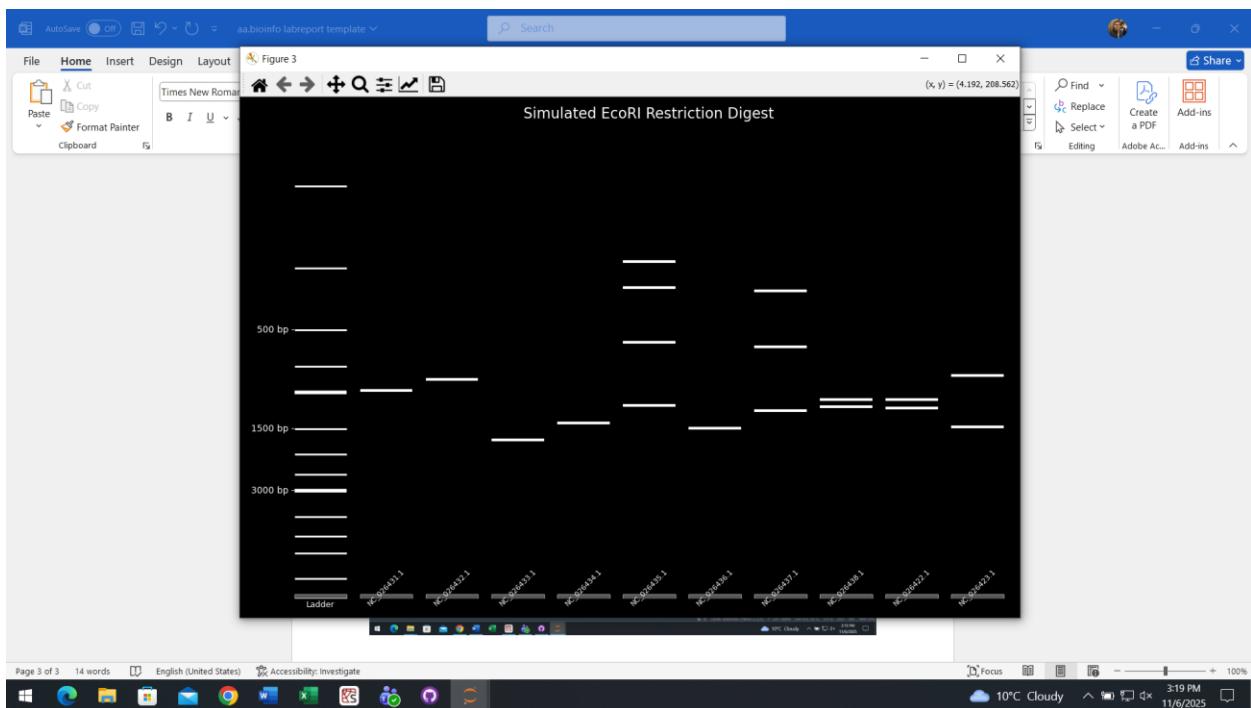
                if line.startswith('>'):
                    # If we are already building a sequence, save it first
                    if current_sequence_name:
                        sequences[current_sequence_name] = ''.join(current_sequence)

                    # Start the new sequence
                    # Clean up the header to get a short, usable name
                    try:
                        # Assumes format like >ID | description
                        current_sequence_name = line.split('/')[-1].lstrip('>').strip()
                    except Exception:
                        pass
                else:
                    # This is a sequence line
                    if current_sequence_name:
                        # Ensure we're inside a sequence block
                        current_sequence.append(line)
                    else:
                        raise ValueError("No sequence header found")
    except Exception as e:
        print(f"Error reading file: {e}")

    return sequences

```





NC_026435.1: 3 bands

NC_026437.1: 2 bands

NC_026438.1: 2 bands

NC_026422.1: 2 bands

NC_026423.1: 2 bands

NC_026431.1: 1 band

NC_026432.1: 1 band

NC_026433.1: 1 band

NC_026434.1: 1 band

NC_026436.1: 1 band