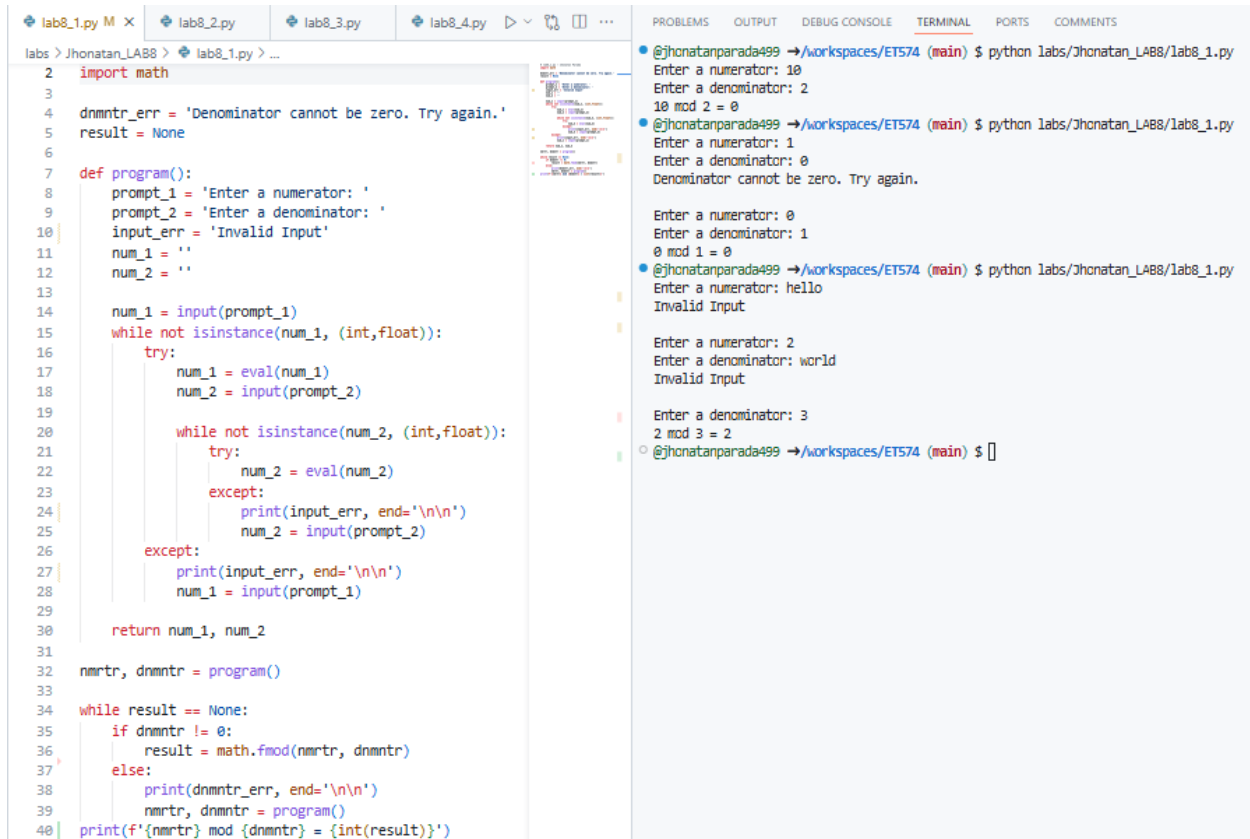


Lab 8 Summary



The image shows a code editor with a Python script on the left and its terminal output on the right. The script defines a function `program()` that prompts the user for a numerator and denominator, validates the input, and calculates the modulo. The terminal shows the execution of the script with various inputs and the resulting output.

```
2 import math
3
4 dnmntr_err = 'Denominator cannot be zero. Try again.'
5 result = None
6
7 def program():
8     prompt_1 = 'Enter a numerator: '
9     prompt_2 = 'Enter a denominator: '
10    input_err = 'Invalid Input'
11    num_1 = ''
12    num_2 = ''
13
14    num_1 = input(prompt_1)
15    while not isinstance(num_1, (int,float)):
16        try:
17            num_1 = eval(num_1)
18            num_2 = input(prompt_2)
19
20            while not isinstance(num_2, (int,float)):
21                try:
22                    num_2 = eval(num_2)
23                except:
24                    print(input_err, end='\n\n')
25                    num_2 = input(prompt_2)
26            except:
27                print(input_err, end='\n\n')
28                num_1 = input(prompt_1)
29
30    return num_1, num_2
31
32 nmtr, dnmtr = program()
33
34 while result == None:
35     if dnmtr != 0:
36         result = math.fmod(nmtr, dnmtr)
37     else:
38         print(dnmtr_err, end='\n\n')
39         nmtr, dnmtr = program()
40 print(f'{nmtr} mod {dnmtr} = {int(result)}')
```

Terminal Output:

```
@jhonatanparada499 →/workspaces/ETS74 (main) $ python labs/jhonatan_LAB8/lab8_1.py
Enter a numerator: 10
Enter a denominator: 2
10 mod 2 = 0
@jhonatanparada499 →/workspaces/ETS74 (main) $ python labs/jhonatan_LAB8/lab8_1.py
Enter a numerator: 1
Enter a denominator: 0
Denominator cannot be zero. Try again.
Enter a numerator: 0
Enter a denominator: 1
0 mod 1 = 0
@jhonatanparada499 →/workspaces/ETS74 (main) $ python labs/jhonatan_LAB8/lab8_1.py
Enter a numerator: hello
Invalid Input
Enter a numerator: 2
Enter a denominator: world
Invalid Input
Enter a denominator: 3
2 mod 3 = 2
@jhonatanparada499 →/workspaces/ETS74 (main) $ []
```

lab8_1.py Mlab8_2.py M ×lab8_3.py lab8_4.py

labs > Jhonatan_LAB8 > lab8_2.py > ...
1 # lab8_2.py - Jhonatan Parada
2 import random
3 import math
4
5 result_msg = 'Square root of'
6 start, stop = 1, 100
7
8 num = random.randint(start, stop)
9 result = math.isqrt(num)
10
11 print(f'{result_msg} {num} = {result}')

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS
• @jhonatanparada499 →/workspaces/ET574 (main) \$ python labs/Jhonatan_LAB8/lab8_2.py
Square root of 82 = 9
• @jhonatanparada499 →/workspaces/ET574 (main) \$ python labs/Jhonatan_LAB8/lab8_2.py
Square root of 16 = 4
• @jhonatanparada499 →/workspaces/ET574 (main) \$ python labs/Jhonatan_LAB8/lab8_2.py
Square root of 56 = 7
• @jhonatanparada499 →/workspaces/ET574 (main) \$ python labs/Jhonatan_LAB8/lab8_2.py
Square root of 56 = 7
• @jhonatanparada499 →/workspaces/ET574 (main) \$ python labs/Jhonatan_LAB8/lab8_2.py
Square root of 44 = 6
• @jhonatanparada499 →/workspaces/ET574 (main) \$ python labs/Jhonatan_LAB8/lab8_2.py
Square root of 51 = 7
• @jhonatanparada499 →/workspaces/ET574 (main) \$ python labs/Jhonatan_LAB8/lab8_2.py
Square root of 62 = 7
○ @jhonatanparada499 →/workspaces/ET574 (main) \$

lab8_1.py Mlab8_2.py M ×lab8_3.py lab8_4.py

labs > Jhonatan_LAB8 > lab8_3.py > ...
1 # lab8_3.py - Jhonatan Parada
2
3 def hello():
4 | print('Hello World')
5
6 hello()

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS
• @jhonatanparada499 →/workspaces/ET574 (main) \$ python labs/Jhonatan_LAB8/lab8_3.py
Hello World
○ @jhonatanparada499 →/workspaces/ET574 (main) \$

lab8_1.py Mlab8_2.py Mlab8_3.py**lab8_4.py**X

labs > Jhonatan_LAB8 > lab8_4.py > ...
1 # lab8_4.py - Jhonatan Parada
2
3 # This could be done:
4 # [import lab8_3] or [from lab8_3 import hello],
5 # but the function call [hello()] in lab8_3
6 # should be removed.
7
8 def hello(): print('Hello World')
9
10 def helloNum(n):
11 par_err = '[n] parameter must be and integer'
12 if not isinstance(n, int):
13 print(par_err)
14 return
15
16 for i in range(n):
17 hello()
18
19 helloNum(3)

PROBLEMSOUTPUTDEBUG CONSOLE**TERMINAL**PORTSCOMMENTS
• @jhonatanparada499 →/workspaces/ET574 (main) \$ python labs/Jhonatan_LAB8/lab8_4.py
Hello World
Hello World
Hello World
○ @jhonatanparada499 →/workspaces/ET574 (main) \$ []

lab8_3.pylab8_4.py**lab8_5.py**Xlab8_6.py

labs > Jhonatan_LAB8 > lab8_5.py > ...
1 # lab8_5.py - Jhonatan Parada
2 from random import randint
3
4 def main():
5 prompt = 'Enter a text: '
6 text = input(prompt).title()
7 n = randint(1, 10)
8
9 print(f'text = {text}')
10 print(f'n = {n}')
11 print(f'message(text, n) will print the following:')
12 message(text, n)
13
14 def message(p1,p2):
15 for i in range(p2):
16 print(p1)
17
18 main()

PROBLEMSOUTPUTDEBUG CONSOLE**TERMINAL**PORTSCOMMENTS
• @jhonatanparada499 →/workspaces/ET574 (main) \$ python labs/Jhonatan_LAB8/lab8_5.py
Enter a text: looking good ahead to spring
text = Looking Good Ahead To Spring
n = 3
message(text, n) will print the following:
Looking Good Ahead To Spring
Looking Good Ahead To Spring
Looking Good Ahead To Spring
Looking Good Ahead To Spring
○ @jhonatanparada499 →/workspaces/ET574 (main) \$ []

```
lab8_3.py lab8_4.py lab8_5.py lab8_6.py M X
labs > Jhonatan_LAB8 > lab8_6.py > middle
1 # lab8_6.py - Jhonatan Parada
2 from random import randint
3
4 def main():
5     n = randint(1, 10)
6     numList = list(range(1, n + 1))
7     print(middle(numList))
8
9 def middle(l):
10     if not isinstance(l, list): return
11
12     msg_1 = 'No changes made to list.'
13     msg_2 = 'List length'
14     lst_len = len(l)
15     new_lst = l.copy()
16
17     if lst_len <= 1: print(msg_1)
18     else: new_lst = new_lst[1:-1]
19
20     print(f'{msg_2} = {lst_len}', 1, sep='\n')
21     return new_lst
22
23 main()
```

```
@jhonatanparada499 ->/workspaces/ETS74 (main) $ python labs/Jhonatan_LAB8/lab8_6.py
List length = 7
[1, 2, 3, 4, 5, 6, 7]
[2, 3, 4, 5, 6]
@jhonatanparada499 ->/workspaces/ETS74 (main) $ python labs/Jhonatan_LAB8/lab8_6.py
No changes made to list.
List length = 1
[1]
[1]
@jhonatanparada499 ->/workspaces/ETS74 (main) $ python labs/Jhonatan_LAB8/lab8_6.py
List length = 2
[1, 2]
[1]
@jhonatanparada499 ->/workspaces/ETS74 (main) $ []
```

2. On this lab, while doing lab8_1.py, I realized that using the function [eval] was a more elegant or readable form to convert a string to an int or float data type, unlike I did in the previous lab, where I used a nested try statement to check and convert the string to either integer or float.

On this lab, I also learned that one can reuse code from other files, such as functions. One of the advantages of this is that one can write more organized and readable code, but it might become harder to deal and manage too many dependencies, especially when those dependencies also have other dependencies, it all can become a recursive game all the sudden.