DUNDALK
INSTITUTE OF TECHNOLOGY
INSTITIÚID TEICNEOLAÍOCHTA
DHÚN DEALGAN

Continuous Assessment
Cover Sheet

Student Name: Neo Sahadeo	**Student Number: D00264604**	
Programme: Computer Systems and Operations	**Stage: 1**	**Complete Student Checklist:** Re-read brief x References and Bibliography □ Proofread □
Module: Introduction to Programming		
Due Date: 20/10/2024	**No. Pages: 4**	
Lecturer(s) Name: Amanda Clancy		
Assignment No. and/or Description/Topic: **CA 1**		**Mode of Submission:** Softcopy x Hardcopy □

DECLARATION: **I declare that:**

- This work is entirely my own, and no part of it has been copied from any other person's words or ideas, except as specifically acknowledged through the use of inverted commas and in-text references;
- No part of this assignment has been written for me by any other person except where such collaboration has been authorised by the lecturer(s) concerned;
- I have not used generative artificial intelligence (AI) (e.g. ChatGPT) unless it has been permitted by the lecturer(s) concerned;
- I understand that I am bound by DkIT Academic Integrity Policy. I understand that I may be penalised if I have violated the policy in any way;
- This assignment has not been submitted for any other module at DkIT or any other institution, unless authorised by the relevant Lecturer(s);
- I have read and abided by all of the requirements set down for this assignment.

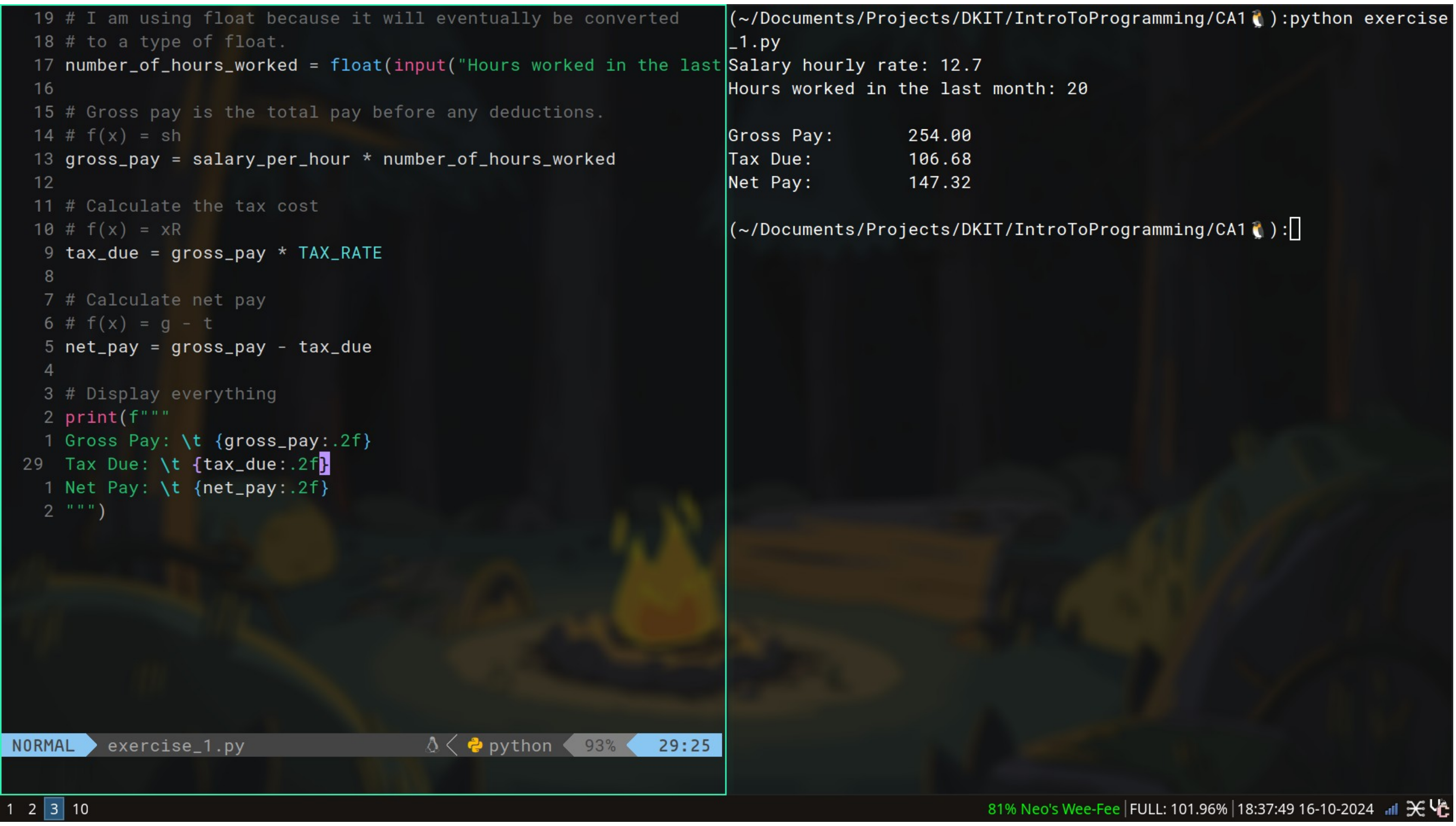
SIGNATURE..................................Neo Sahadeo... **DATE**...16/10/2024.............

Lecturer's Comments:

Provisional Mark : _____ **Lecturers Signature :** ________________________ **Date:** _____________

Work submitted late will be subject to penalties in accordance with the DkIT Continuous Assessment Policy

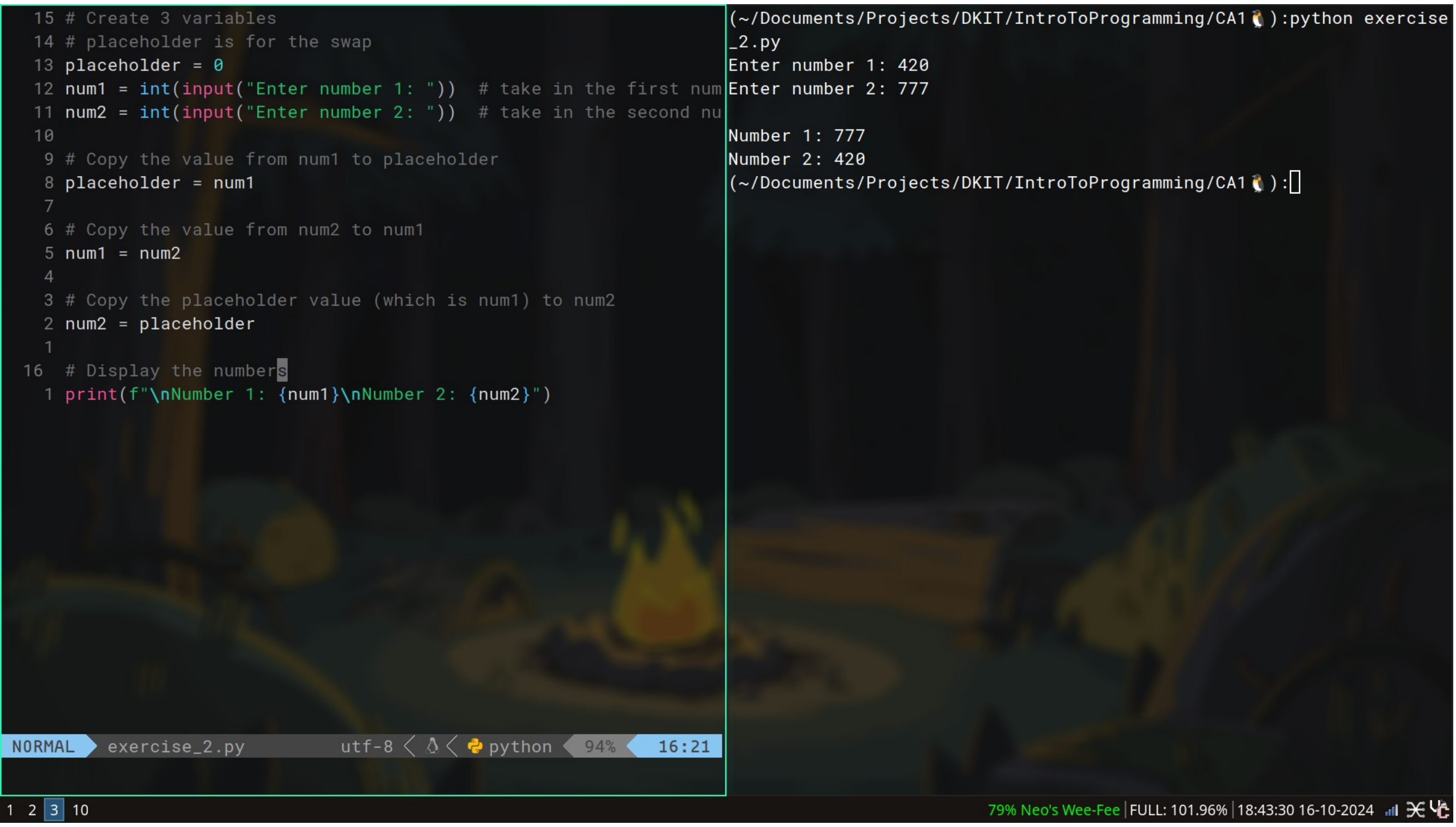
Exercise 1



```python
# I am using float because it will eventually be converted
# to a type of float.
number_of_hours_worked = float(input("Hours worked in the last

# Gross pay is the total pay before any deductions.
# f(x) = sh
gross_pay = salary_per_hour * number_of_hours_worked

# Calculate the tax cost
# f(x) = xR
tax_due = gross_pay * TAX_RATE

# Calculate net pay
# f(x) = g - t
net_pay = gross_pay - tax_due

# Display everything
print(f"""
Gross Pay: \t {gross_pay:.2f}
Tax Due: \t {tax_due:.2f}
Net Pay: \t {net_pay:.2f}
""")
```

```
(~/Documents/Projects/DKIT/IntroToProgramming/CA1🐧):python exercise
_1.py
Salary hourly rate: 12.7
Hours worked in the last month: 20

Gross Pay:       254.00
Tax Due:         106.68
Net Pay:         147.32

(~/Documents/Projects/DKIT/IntroToProgramming/CA1🐧):
```

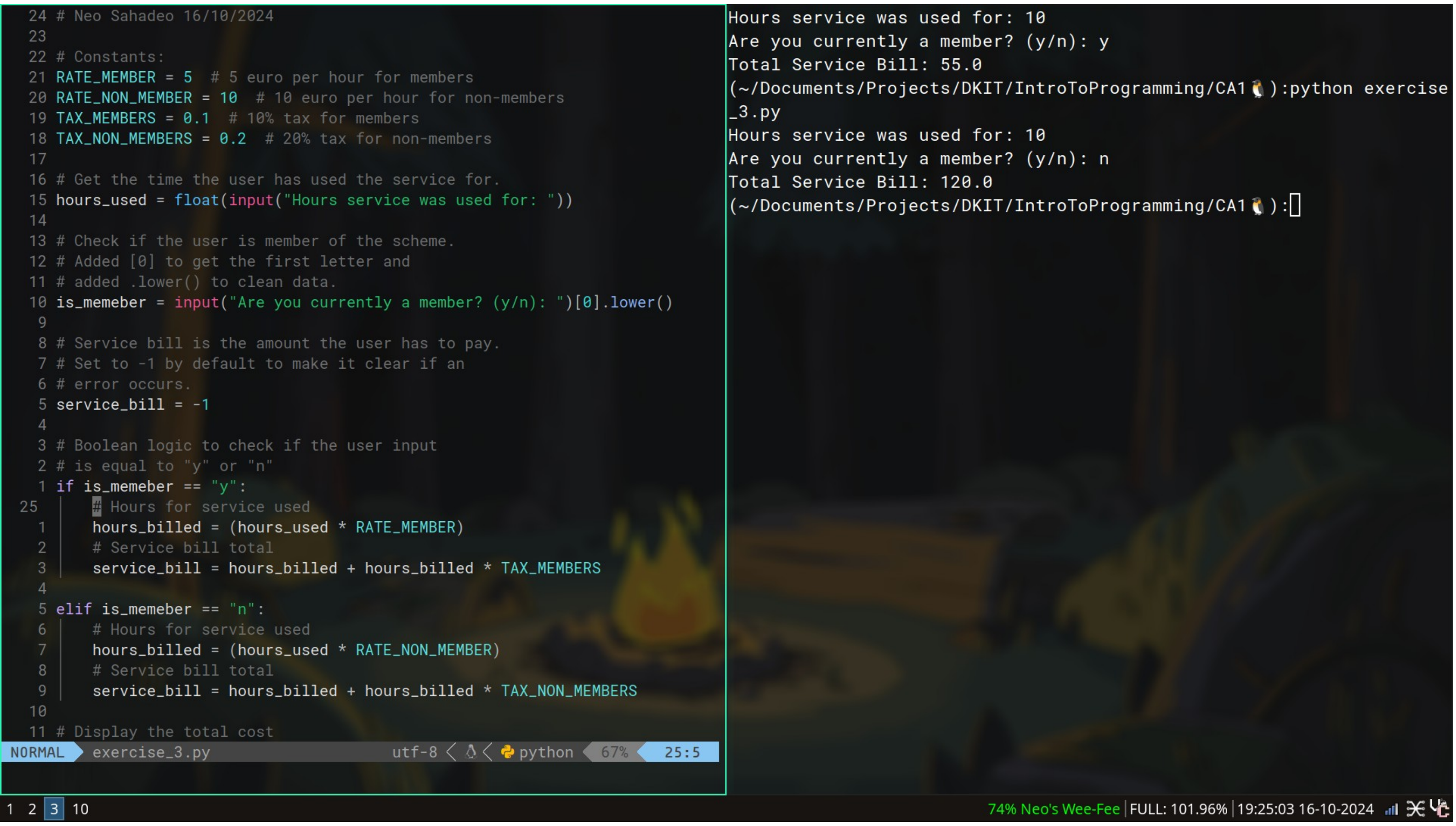
Exercise 2



```python
# Create 3 variables
# placeholder is for the swap
placeholder = 0
num1 = int(input("Enter number 1: "))  # take in the first num
num2 = int(input("Enter number 2: "))  # take in the second nu

# Copy the value from num1 to placeholder
placeholder = num1

# Copy the value from num2 to num1
num1 = num2

# Copy the placeholder value (which is num1) to num2
num2 = placeholder

# Display the numbers
print(f"\nNumber 1: {num1}\nNumber 2: {num2}")
```

```
(~/Documents/Projects/DKIT/IntroToProgramming/CA1🐧):python exercise
_2.py
Enter number 1: 420
Enter number 2: 777

Number 1: 777
Number 2: 420
(~/Documents/Projects/DKIT/IntroToProgramming/CA1🐧):
```

Exercise 3

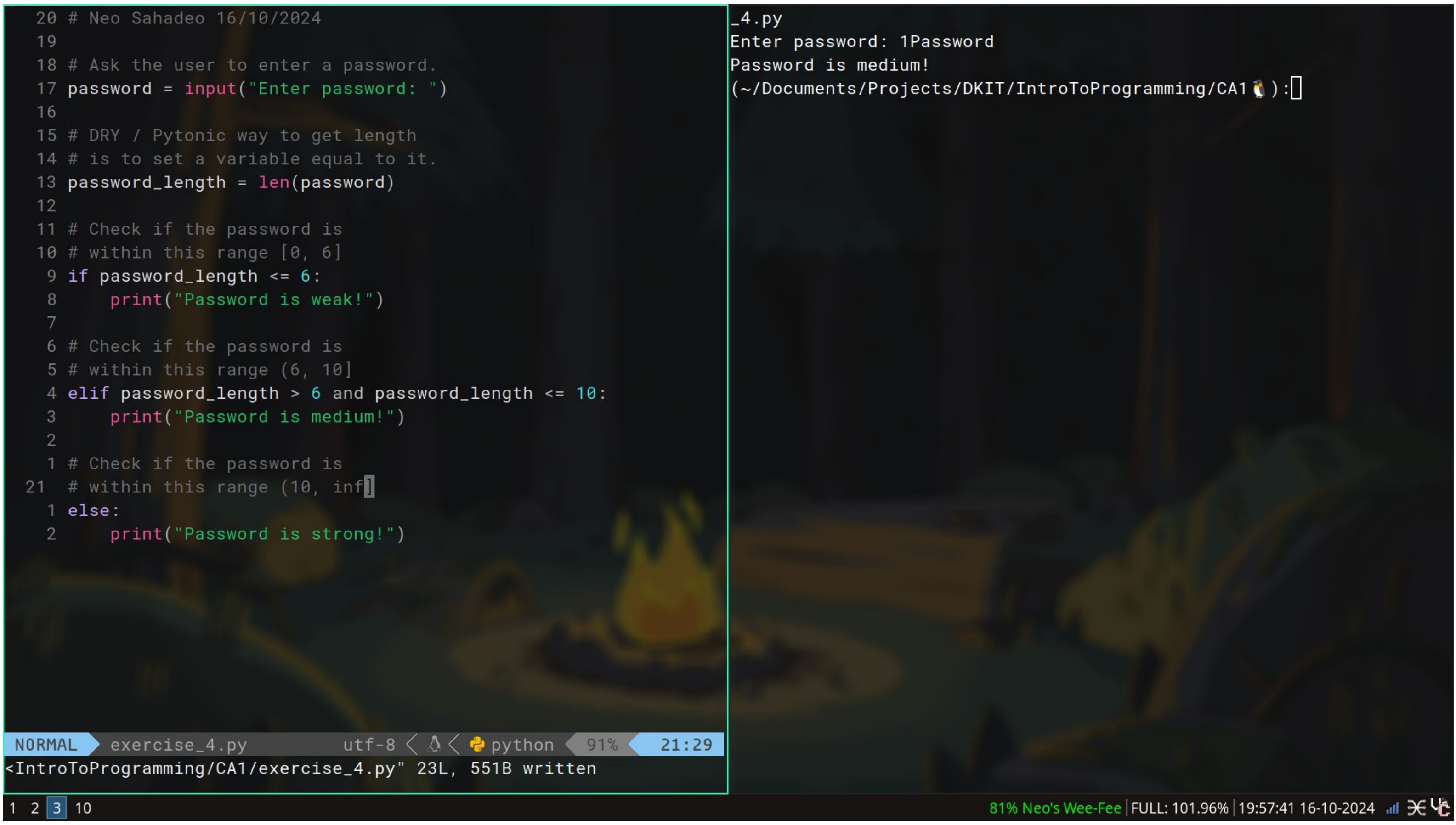


```
# Neo Sahadeo 16/10/2024

# Constants:
RATE_MEMBER = 5  # 5 euro per hour for members
RATE_NON_MEMBER = 10  # 10 euro per hour for non-members
TAX_MEMBERS = 0.1  # 10% tax for members
TAX_NON_MEMBERS = 0.2  # 20% tax for non-members

# Get the time the user has used the service for.
hours_used = float(input("Hours service was used for: "))

# Check if the user is member of the scheme.
# Added [0] to get the first letter and
# added .lower() to clean data.
is_memeber = input("Are you currently a member? (y/n): ")[0].lower()

# Service bill is the amount the user has to pay.
# Set to -1 by default to make it clear if an
# error occurs.
service_bill = -1

# Boolean logic to check if the user input
# is equal to "y" or "n"
if is_memeber == "y":
    # Hours for service used
    hours_billed = (hours_used * RATE_MEMBER)
    # Service bill total
    service_bill = hours_billed + hours_billed * TAX_MEMBERS

elif is_memeber == "n":
    # Hours for service used
    hours_billed = (hours_used * RATE_NON_MEMBER)
    # Service bill total
    service_bill = hours_billed + hours_billed * TAX_NON_MEMBERS

# Display the total cost
```

```
NORMAL  exercise_3.py      utf-8  python  67%  25:5
```

```
Hours service was used for: 10
Are you currently a member? (y/n): y
Total Service Bill: 55.0
(~/Documents/Projects/DKIT/IntroToProgramming/CA1 ):python exercise
_3.py
Hours service was used for: 10
Are you currently a member? (y/n): n
Total Service Bill: 120.0
(~/Documents/Projects/DKIT/IntroToProgramming/CA1 ):
```

Exercise 4

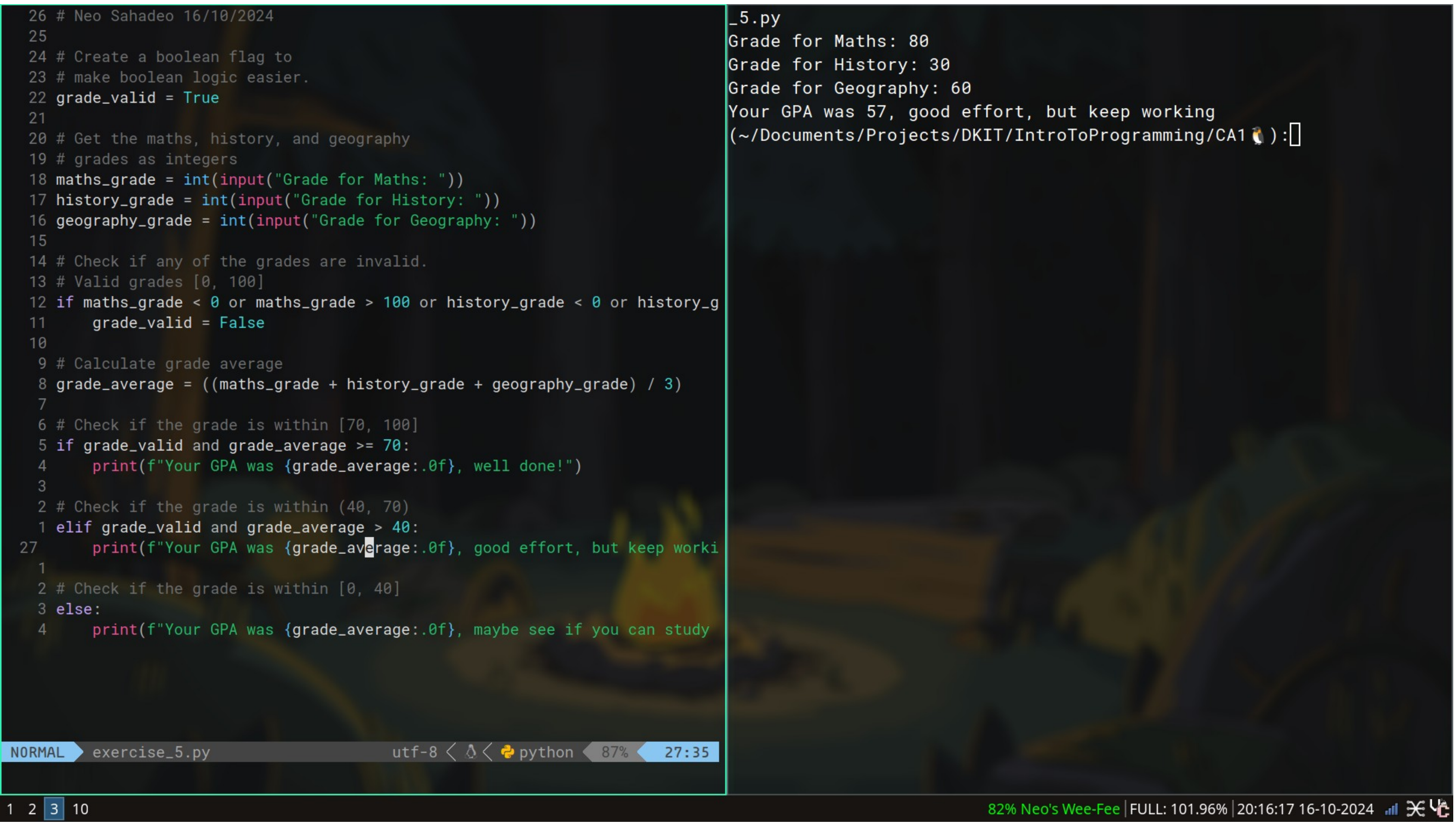


```
# Neo Sahadeo 16/10/2024

# Ask the user to enter a password.
password = input("Enter password: ")

# DRY / Pytonic way to get length
# is to set a variable equal to it.
password_length = len(password)

# Check if the password is
# within this range [0, 6]
if password_length <= 6:
    print("Password is weak!")

# Check if the password is
# within this range (6, 10]
elif password_length > 6 and password_length <= 10:
    print("Password is medium!")

# Check if the password is
# within this range (10, inf]
else:
    print("Password is strong!")
```

```
NORMAL  exercise_4.py      utf-8  python  91%  21:29
<IntroToProgramming/CA1/exercise_4.py" 23L, 551B written
```

```
_4.py
Enter password: 1Password
Password is medium!
(~/Documents/Projects/DKIT/IntroToProgramming/CA1 ):
```

Exercise 5



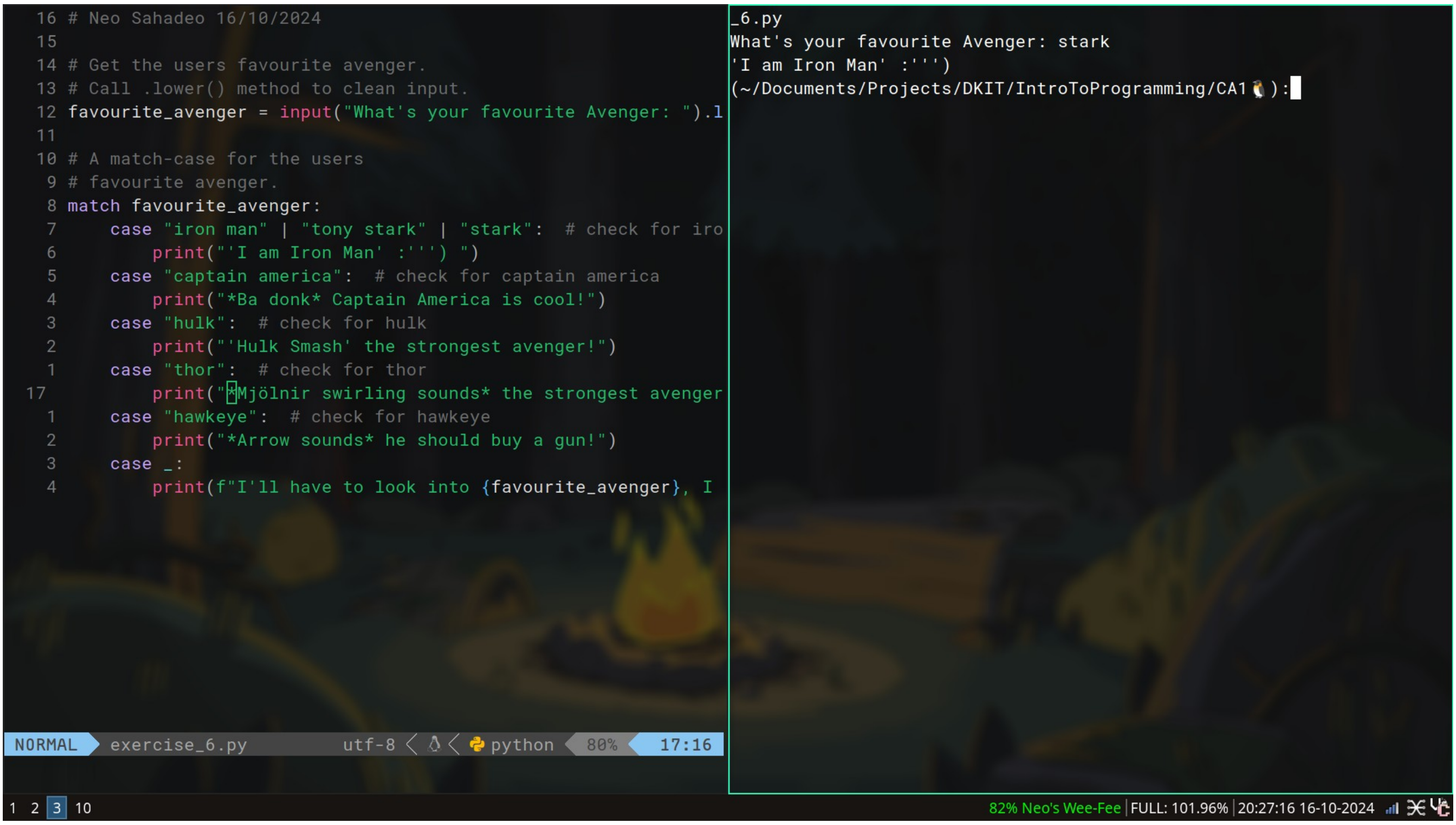
```
# Neo Sahadeo 16/10/2024

# Create a boolean flag to
# make boolean logic easier.
grade_valid = True

# Get the maths, history, and geography
# grades as integers
maths_grade = int(input("Grade for Maths: "))
history_grade = int(input("Grade for History: "))
geography_grade = int(input("Grade for Geography: "))

# Check if any of the grades are invalid.
# Valid grades [0, 100]
if maths_grade < 0 or maths_grade > 100 or history_grade < 0 or history_g
    grade_valid = False

# Calculate grade average
grade_average = ((maths_grade + history_grade + geography_grade) / 3)

# Check if the grade is within [70, 100]
if grade_valid and grade_average >= 70:
    print(f"Your GPA was {grade_average:.0f}, well done!")

# Check if the grade is within (40, 70)
elif grade_valid and grade_average > 40:
    print(f"Your GPA was {grade_average:.0f}, good effort, but keep worki

# Check if the grade is within [0, 40]
else:
    print(f"Your GPA was {grade_average:.0f}, maybe see if you can study
```

```
NORMAL  exercise_5.py                utf-8  python  87%  27:35
```

```
_5.py
Grade for Maths: 80
Grade for History: 30
Grade for Geography: 60
Your GPA was 57, good effort, but keep working
(~/Documents/Projects/DKIT/IntroToProgramming/CA1 ):
```

```
1 2 3 10                 82% Neo's Wee-Fee | FULL: 101.96% | 20:16:17 16-10-2024
```

Exercise 6



```
# Neo Sahadeo 16/10/2024

# Get the users favourite avenger.
# Call .lower() method to clean input.
favourite_avenger = input("What's your favourite Avenger: ").l

# A match-case for the users
# favourite avenger.
match favourite_avenger:
    case "iron man" | "tony stark" | "stark":  # check for iro
        print("'I am Iron Man' :''') ")
    case "captain america":  # check for captain america
        print("*Ba donk* Captain America is cool!")
    case "hulk":  # check for hulk
        print("'Hulk Smash' the strongest avenger!")
    case "thor":  # check for thor
        print("*Mjölnir swirling sounds* the strongest avenger
    case "hawkeye":  # check for hawkeye
        print("*Arrow sounds* he should buy a gun!")
    case _:
        print(f"I'll have to look into {favourite_avenger}, I
```

```
NORMAL  exercise_6.py          utf-8  python  80%  17:16
```

```
_6.py
What's your favourite Avenger: stark
'I am Iron Man' :''')
(~/Documents/Projects/DKIT/IntroToProgramming/CA1 ):
```

```
1 2 3 10                 82% Neo's Wee-Fee | FULL: 101.96% | 20:27:16 16-10-2024
```