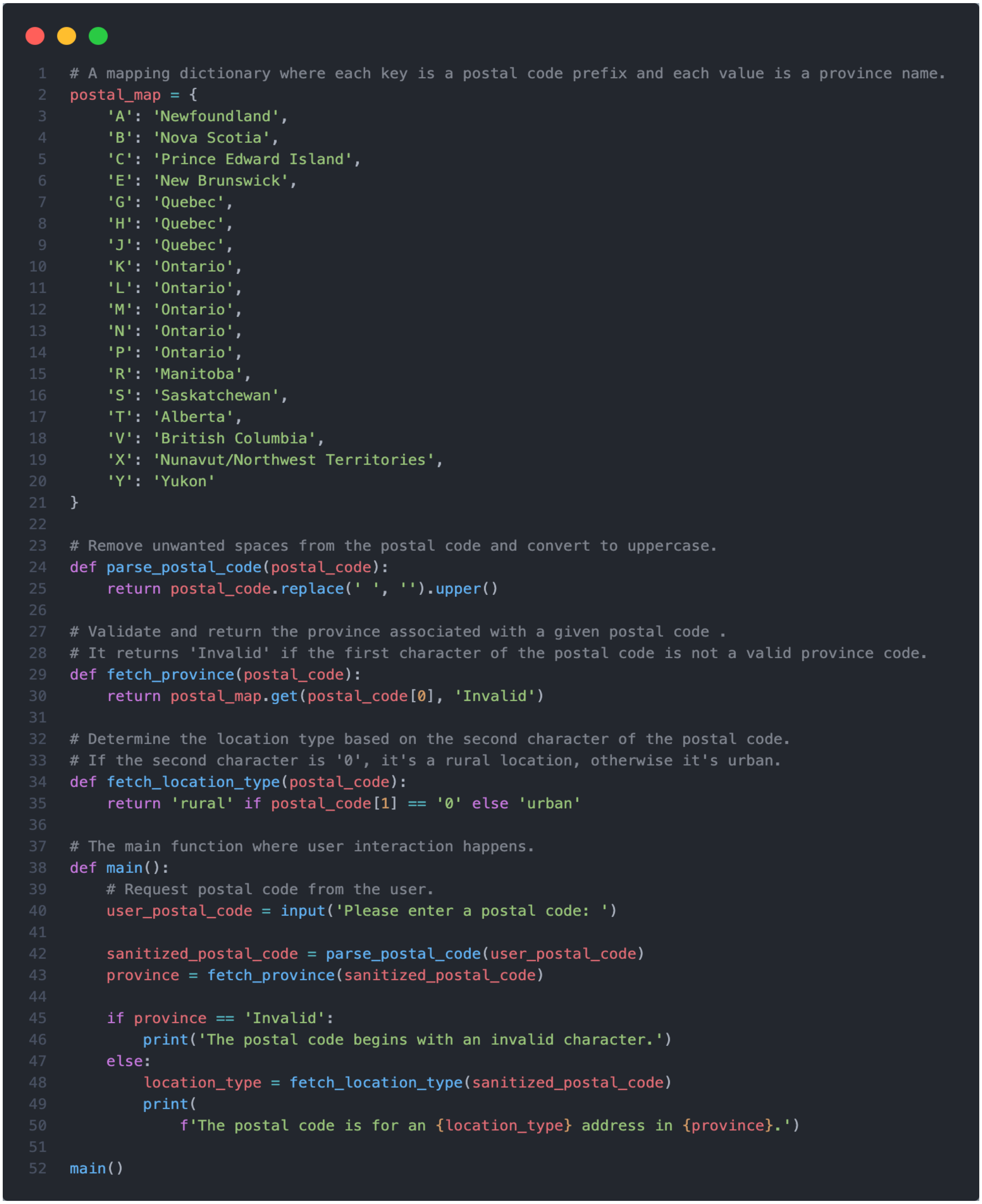
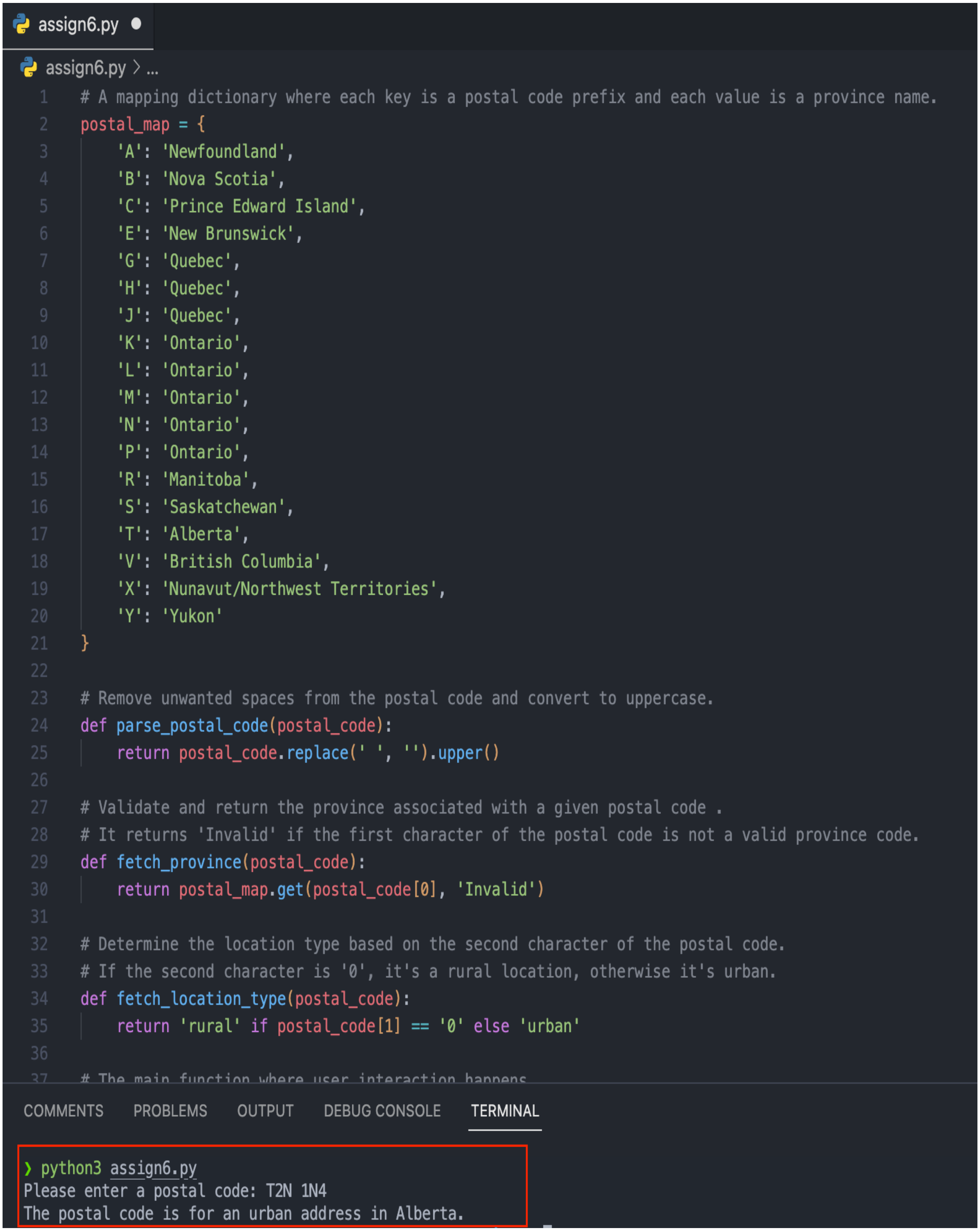
Assignment 6
Student ID: 901142
Student Name: Roshan Shrestha

Source code:



```python
# A mapping dictionary where each key is a postal code prefix and each value is a province name.
postal_map = {
    'A': 'Newfoundland',
    'B': 'Nova Scotia',
    'C': 'Prince Edward Island',
    'E': 'New Brunswick',
    'G': 'Quebec',
    'H': 'Quebec',
    'J': 'Quebec',
    'K': 'Ontario',
    'L': 'Ontario',
    'M': 'Ontario',
    'N': 'Ontario',
    'P': 'Ontario',
    'R': 'Manitoba',
    'S': 'Saskatchewan',
    'T': 'Alberta',
    'V': 'British Columbia',
    'X': 'Nunavut/Northwest Territories',
    'Y': 'Yukon'
}

# Remove unwanted spaces from the postal code and convert to uppercase.
def parse_postal_code(postal_code):
    return postal_code.replace(' ', '').upper()

# Validate and return the province associated with a given postal code .
# It returns 'Invalid' if the first character of the postal code is not a valid province code.
def fetch_province(postal_code):
    return postal_map.get(postal_code[0], 'Invalid')

# Determine the location type based on the second character of the postal code.
# If the second character is '0', it's a rural location, otherwise it's urban.
def fetch_location_type(postal_code):
    return 'rural' if postal_code[1] == '0' else 'urban'

# The main function where user interaction happens.
def main():
    # Request postal code from the user.
    user_postal_code = input('Please enter a postal code: ')

    sanitized_postal_code = parse_postal_code(user_postal_code)
    province = fetch_province(sanitized_postal_code)

    if province == 'Invalid':
        print('The postal code begins with an invalid character.')
    else:
        location_type = fetch_location_type(sanitized_postal_code)
        print(
            f'The postal code is for an {location_type} address in {province}.')

main()
```

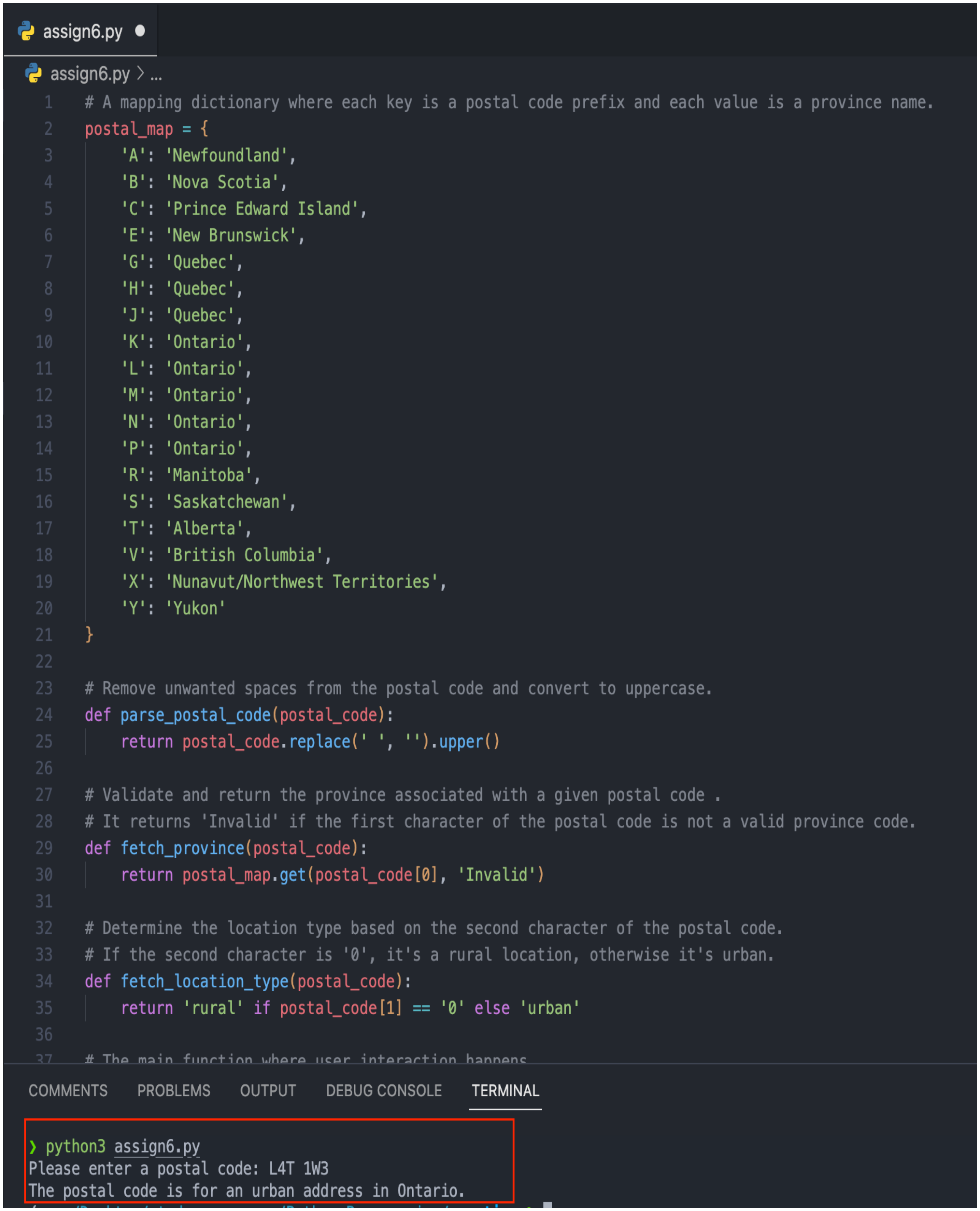
Test case 1 [T2N 1N4]:



```python
# A mapping dictionary where each key is a postal code prefix and each value is a province name.
postal_map = {
    'A': 'Newfoundland',
    'B': 'Nova Scotia',
    'C': 'Prince Edward Island',
    'E': 'New Brunswick',
    'G': 'Quebec',
    'H': 'Quebec',
    'J': 'Quebec',
    'K': 'Ontario',
    'L': 'Ontario',
    'M': 'Ontario',
    'N': 'Ontario',
    'P': 'Ontario',
    'R': 'Manitoba',
    'S': 'Saskatchewan',
    'T': 'Alberta',
    'V': 'British Columbia',
    'X': 'Nunavut/Northwest Territories',
    'Y': 'Yukon'
}

# Remove unwanted spaces from the postal code and convert to uppercase.
def parse_postal_code(postal_code):
    return postal_code.replace(' ', '').upper()

# Validate and return the province associated with a given postal code .
# It returns 'Invalid' if the first character of the postal code is not a valid province code.
def fetch_province(postal_code):
    return postal_map.get(postal_code[0], 'Invalid')

# Determine the location type based on the second character of the postal code.
# If the second character is '0', it's a rural location, otherwise it's urban.
def fetch_location_type(postal_code):
    return 'rural' if postal_code[1] == '0' else 'urban'

# The main function where user interaction happens
```

```
❯ python3 assign6.py
Please enter a postal code: T2N 1N4
The postal code is for an urban address in Alberta.
```

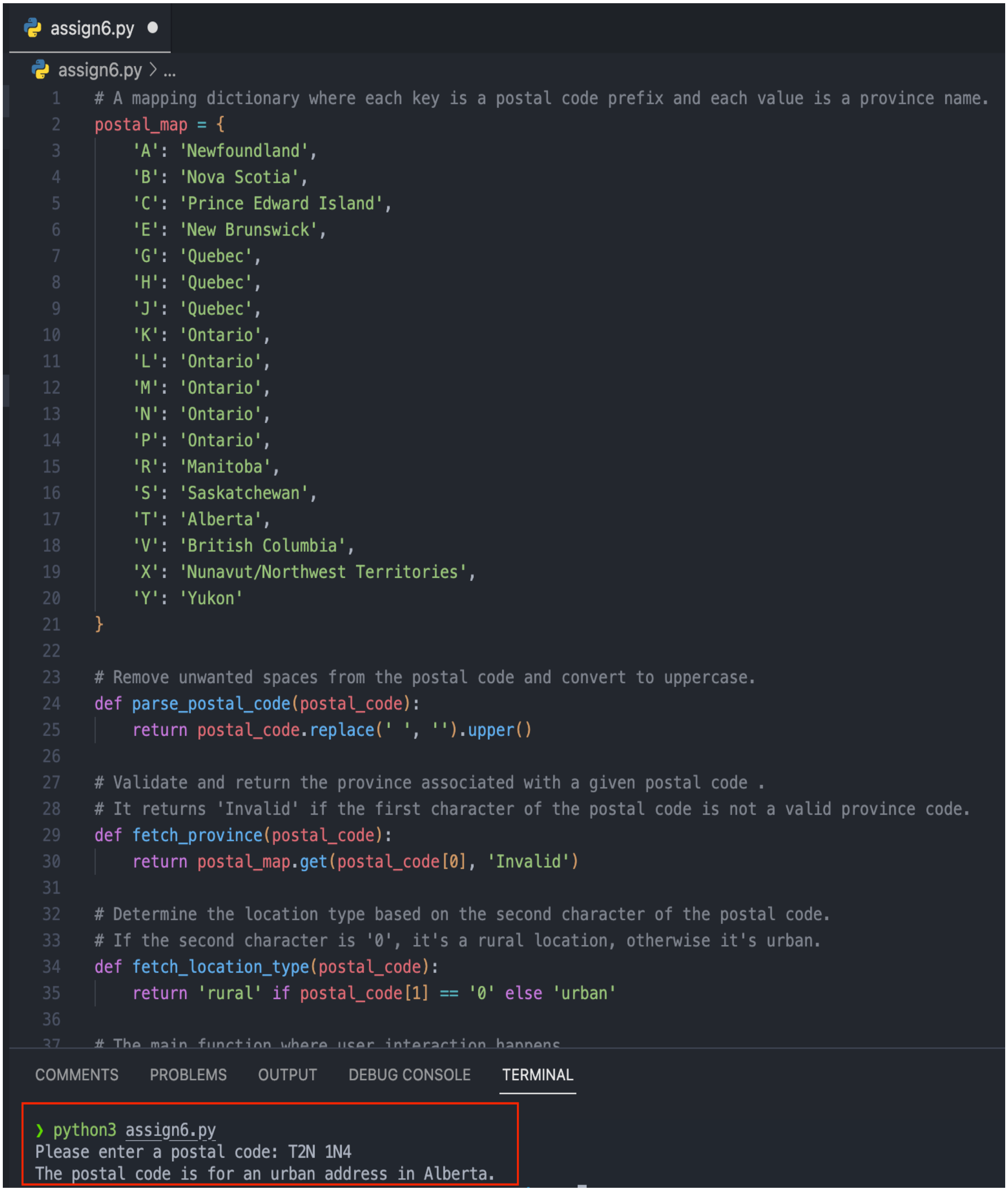
Test case 2 [L4T 1W3]:



```python
# A mapping dictionary where each key is a postal code prefix and each value is a province name.
postal_map = {
    'A': 'Newfoundland',
    'B': 'Nova Scotia',
    'C': 'Prince Edward Island',
    'E': 'New Brunswick',
    'G': 'Quebec',
    'H': 'Quebec',
    'J': 'Quebec',
    'K': 'Ontario',
    'L': 'Ontario',
    'M': 'Ontario',
    'N': 'Ontario',
    'P': 'Ontario',
    'R': 'Manitoba',
    'S': 'Saskatchewan',
    'T': 'Alberta',
    'V': 'British Columbia',
    'X': 'Nunavut/Northwest Territories',
    'Y': 'Yukon'
}

# Remove unwanted spaces from the postal code and convert to uppercase.
def parse_postal_code(postal_code):
    return postal_code.replace(' ', '').upper()

# Validate and return the province associated with a given postal code .
# It returns 'Invalid' if the first character of the postal code is not a valid province code.
def fetch_province(postal_code):
    return postal_map.get(postal_code[0], 'Invalid')

# Determine the location type based on the second character of the postal code.
# If the second character is '0', it's a rural location, otherwise it's urban.
def fetch_location_type(postal_code):
    return 'rural' if postal_code[1] == '0' else 'urban'

# The main function where user interaction happens
```

```
❯ python3 assign6.py
Please enter a postal code: L4T 1W3
The postal code is for an urban address in Ontario.
```

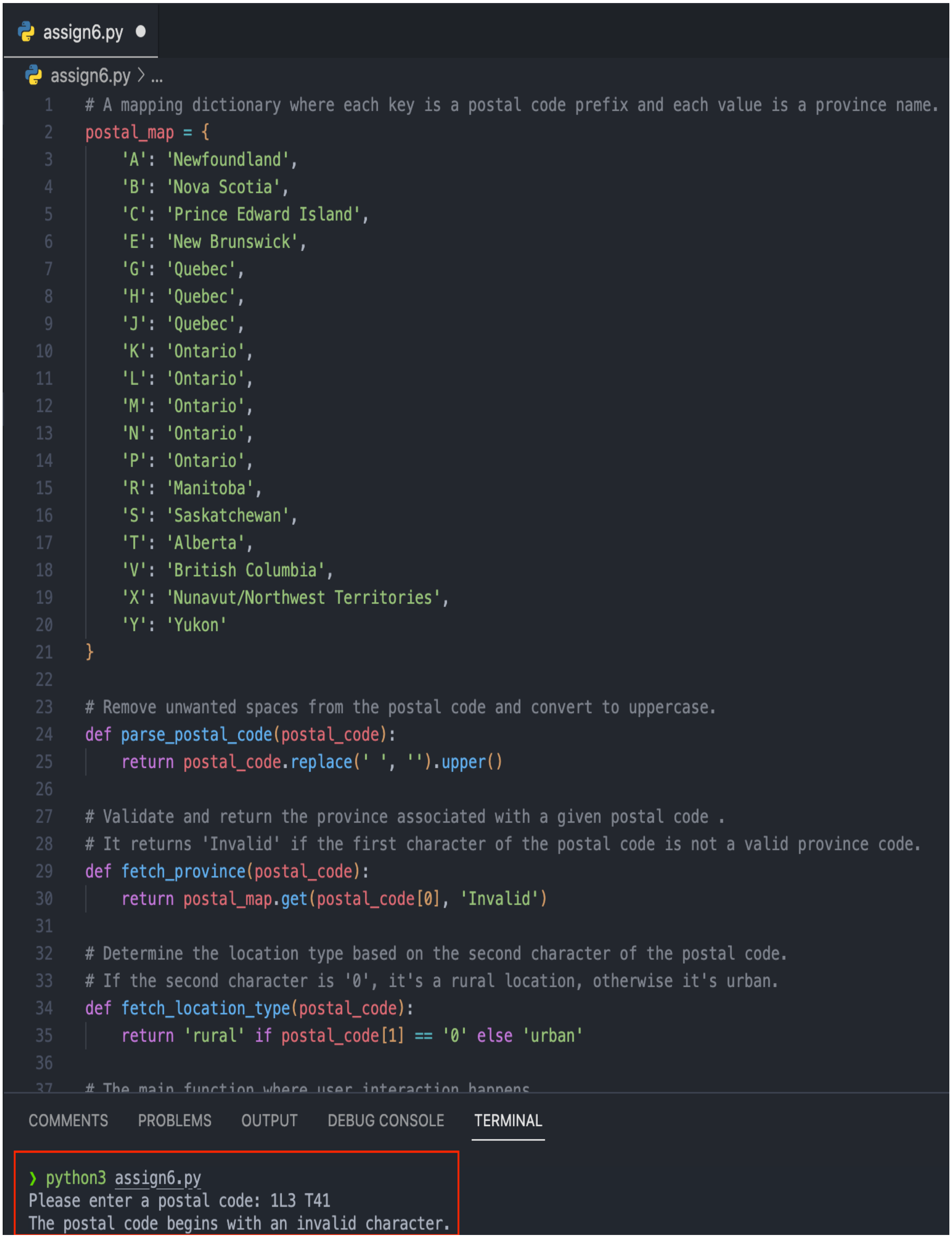
Test case 3 [T2N 1N4]:



```python
# A mapping dictionary where each key is a postal code prefix and each value is a province name.
postal_map = {
    'A': 'Newfoundland',
    'B': 'Nova Scotia',
    'C': 'Prince Edward Island',
    'E': 'New Brunswick',
    'G': 'Quebec',
    'H': 'Quebec',
    'J': 'Quebec',
    'K': 'Ontario',
    'L': 'Ontario',
    'M': 'Ontario',
    'N': 'Ontario',
    'P': 'Ontario',
    'R': 'Manitoba',
    'S': 'Saskatchewan',
    'T': 'Alberta',
    'V': 'British Columbia',
    'X': 'Nunavut/Northwest Territories',
    'Y': 'Yukon'
}

# Remove unwanted spaces from the postal code and convert to uppercase.
def parse_postal_code(postal_code):
    return postal_code.replace(' ', '').upper()

# Validate and return the province associated with a given postal code .
# It returns 'Invalid' if the first character of the postal code is not a valid province code.
def fetch_province(postal_code):
    return postal_map.get(postal_code[0], 'Invalid')

# Determine the location type based on the second character of the postal code.
# If the second character is '0', it's a rural location, otherwise it's urban.
def fetch_location_type(postal_code):
    return 'rural' if postal_code[1] == '0' else 'urban'

# The main function where user interaction happens
```

```
❯ python3 assign6.py
Please enter a postal code: T2N 1N4
The postal code is for an urban address in Alberta.
```

Test case 4 [1L3 T41]:



```python
# A mapping dictionary where each key is a postal code prefix and each value is a province name.
postal_map = {
    'A': 'Newfoundland',
    'B': 'Nova Scotia',
    'C': 'Prince Edward Island',
    'E': 'New Brunswick',
    'G': 'Quebec',
    'H': 'Quebec',
    'J': 'Quebec',
    'K': 'Ontario',
    'L': 'Ontario',
    'M': 'Ontario',
    'N': 'Ontario',
    'P': 'Ontario',
    'R': 'Manitoba',
    'S': 'Saskatchewan',
    'T': 'Alberta',
    'V': 'British Columbia',
    'X': 'Nunavut/Northwest Territories',
    'Y': 'Yukon'
}

# Remove unwanted spaces from the postal code and convert to uppercase.
def parse_postal_code(postal_code):
    return postal_code.replace(' ', '').upper()

# Validate and return the province associated with a given postal code .
# It returns 'Invalid' if the first character of the postal code is not a valid province code.
def fetch_province(postal_code):
    return postal_map.get(postal_code[0], 'Invalid')

# Determine the location type based on the second character of the postal code.
# If the second character is '0', it's a rural location, otherwise it's urban.
def fetch_location_type(postal_code):
    return 'rural' if postal_code[1] == '0' else 'urban'

# The main function where user interaction happens
```

```
❯ python3 assign6.py
Please enter a postal code: 1L3 T41
The postal code begins with an invalid character.
```