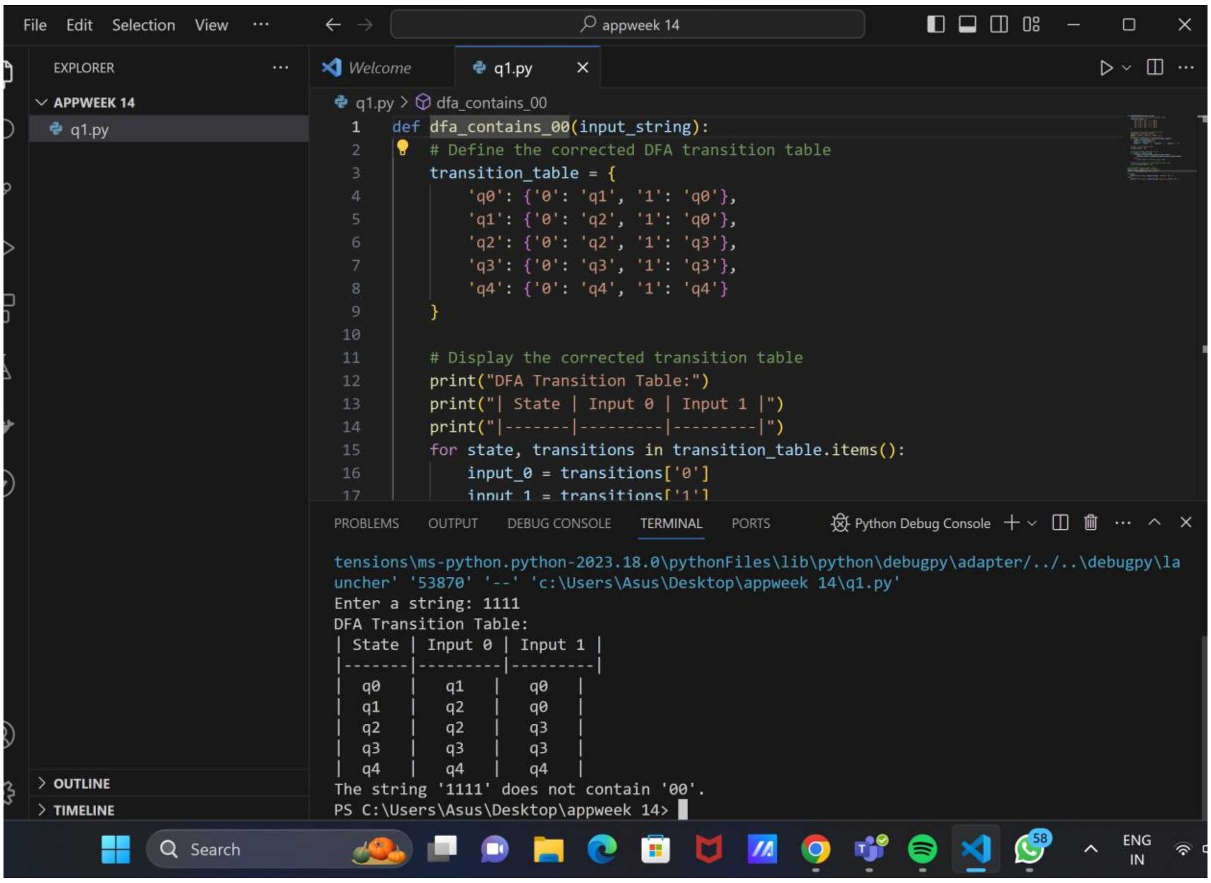
APP WEEK 14

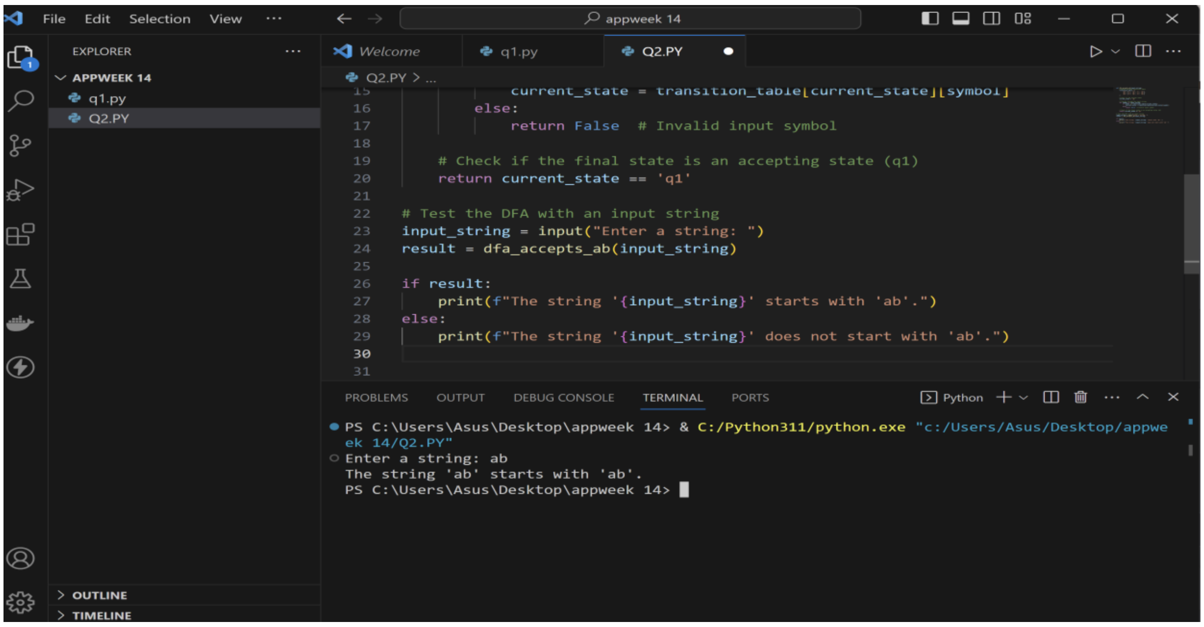
Q1) . Draw a DFA and give the transi�on table for the language that accepts all and only those strings that contain 00



```python
def dfa_contains_00(input_string):
    # Define the corrected DFA transition table
    transition_table = {
        'q0': {'0': 'q1', '1': 'q0'},
        'q1': {'0': 'q2', '1': 'q0'},
        'q2': {'0': 'q2', '1': 'q3'},
        'q3': {'0': 'q3', '1': 'q3'},
        'q4': {'0': 'q4', '1': 'q4'}
    }

    # Display the corrected transition table
    print("DFA Transition Table:")
    print("| State | Input 0 | Input 1 |")
    print("|-------|---------|---------|")
    for state, transitions in transition_table.items():
        input_0 = transitions['0']
        input_1 = transitions['1']
```

```
tensions\ms-python.python-2023.18.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\la
uncher' '53870' '--' 'c:\Users\Asus\Desktop\appweek 14\q1.py'
Enter a string: 1111
DFA Transition Table:
| State | Input 0 | Input 1 |
|-------|---------|---------|
|  q0   |   q1    |   q0    |
|  q1   |   q2    |   q0    |
|  q2   |   q2    |   q3    |
|  q3   |   q3    |   q3    |
|  q4   |   q4    |   q4    |
The string '1111' does not contain '00'.
PS C:\Users\Asus\Desktop\appweek 14>
```

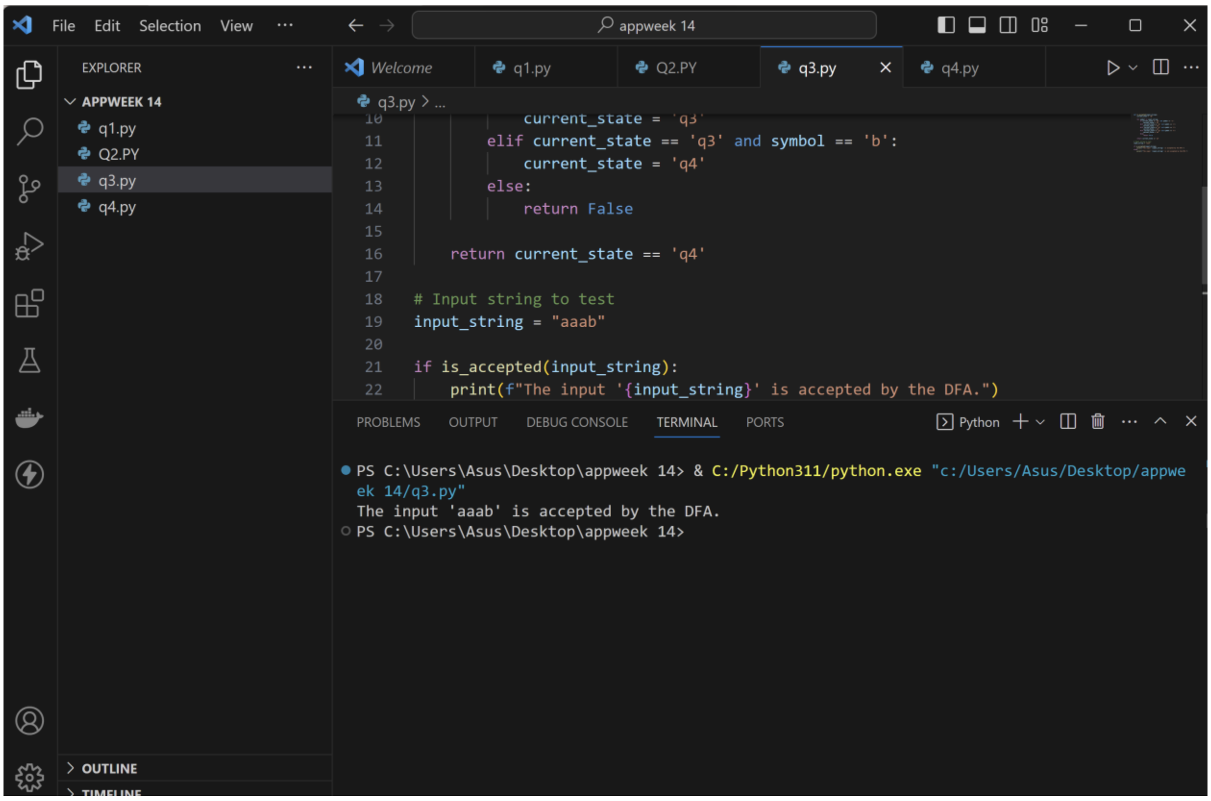
Q2) Draw a DFA and give the transi�on table for the language accep�ng strings star�ng with ‘ab’ over input alphabets ∑ = {a, b}.



```python
            current_state = transition_table[current_state][symbol]
        else:
            return False  # Invalid input symbol

    # Check if the final state is an accepting state (q1)
    return current_state == 'q1'

# Test the DFA with an input string
input_string = input("Enter a string: ")
result = dfa_accepts_ab(input_string)

if result:
    print(f"The string '{input_string}' starts with 'ab'.")
else:
    print(f"The string '{input_string}' does not start with 'ab'.")
```

```
PS C:\Users\Asus\Desktop\appweek 14> & C:/Python311/python.exe "c:/Users/Asus/Desktop/appwe
ek 14/Q2.PY"
Enter a string: ab
The string 'ab' starts with 'ab'.
PS C:\Users\Asus\Desktop\appweek 14>
```

Q3)Construct a DFA with ∑ = {a, b} accepts the only input “aaab”.--

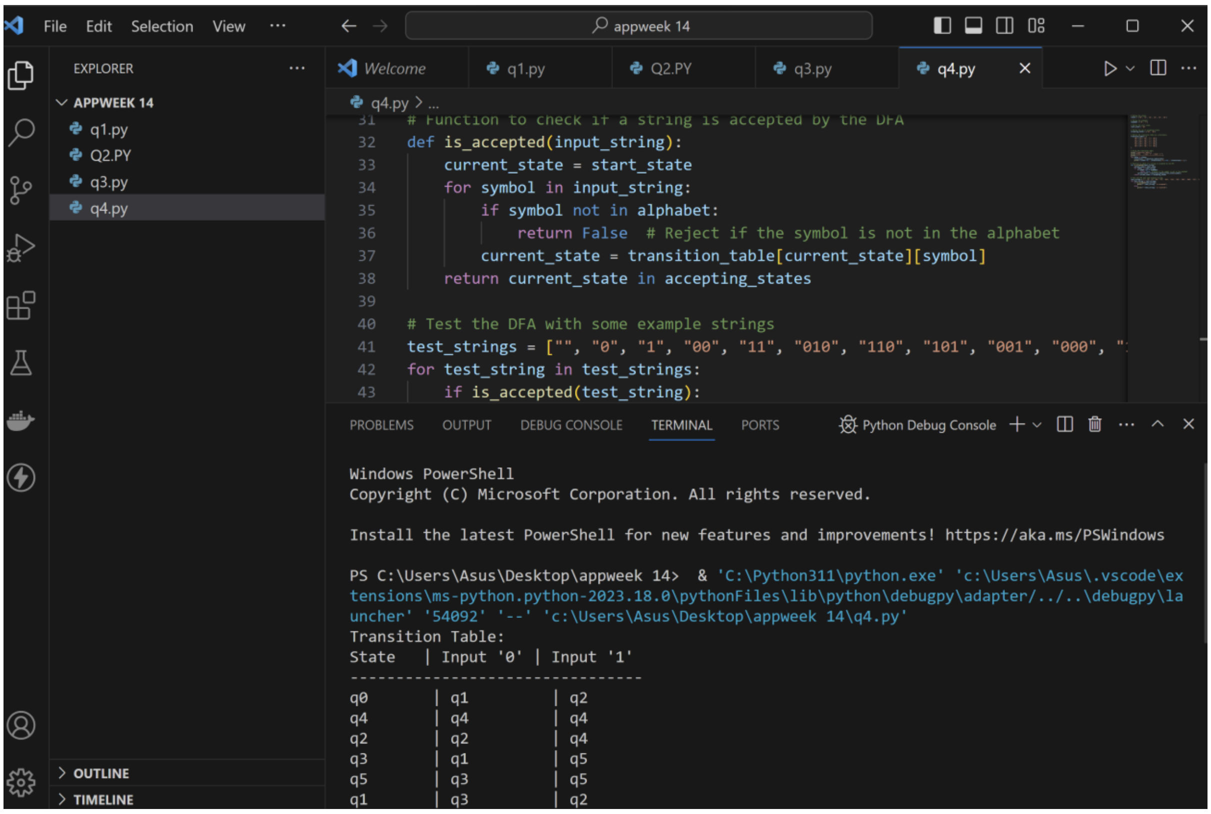


```python
            current_state = 'q3'
        elif current_state == 'q3' and symbol == 'b':
            current_state = 'q4'
        else:
            return False

    return current_state == 'q4'

# Input string to test
input_string = "aaab"

if is_accepted(input_string):
    print(f"The input '{input_string}' is accepted by the DFA.")
```

```
PS C:\Users\Asus\Desktop\appweek 14> & C:/Python311/python.exe "c:/Users/Asus/Desktop/appwe
ek 14/q3.py"
The input 'aaab' is accepted by the DFA.
PS C:\Users\Asus\Desktop\appweek 14>
```

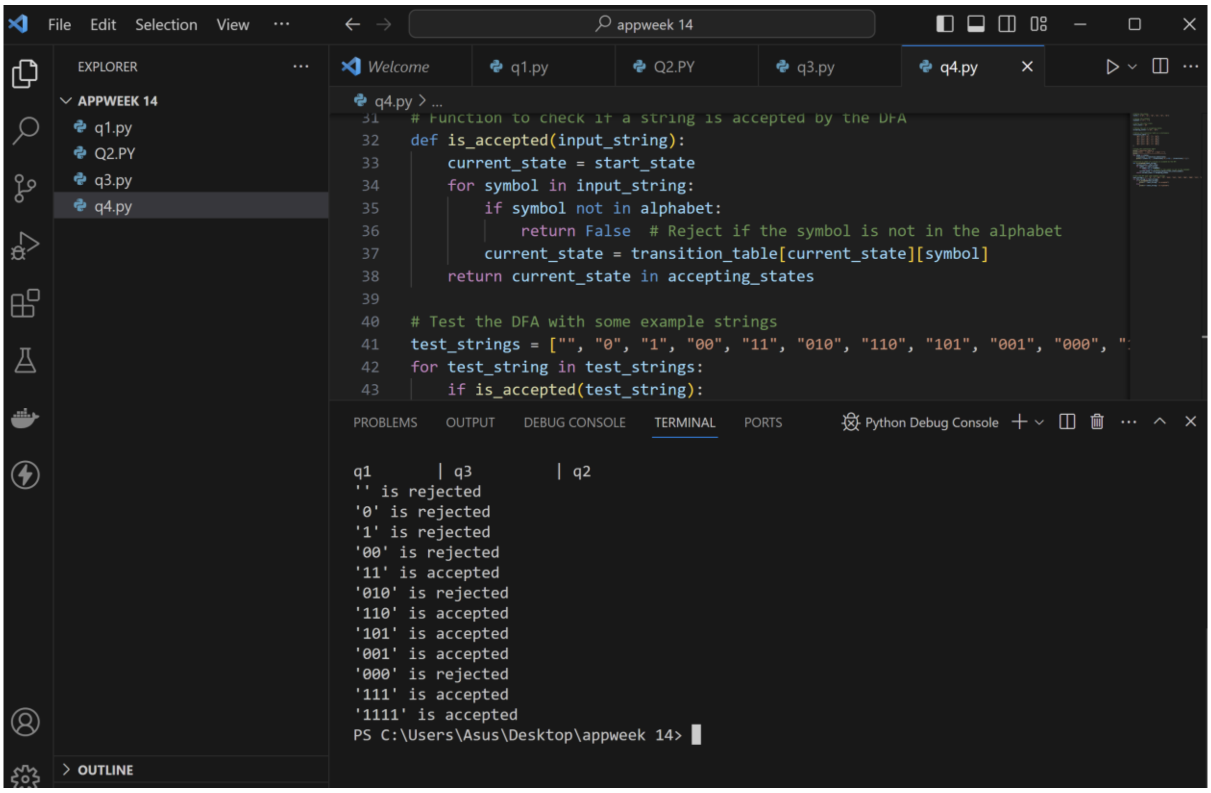
Q4) Draw a DFA and give the transi�on table for the language L(M)=a + aa*b. 5.

Draw a determinis�c finite automate which accept 00 and 11 at the end of a string containing 0, 1.



```python
# Function to check if a string is accepted by the DFA
def is_accepted(input_string):
    current_state = start_state
    for symbol in input_string:
        if symbol not in alphabet:
            return False  # Reject if the symbol is not in the alphabet
        current_state = transition_table[current_state][symbol]
    return current_state in accepting_states

# Test the DFA with some example strings
test_strings = ["", "0", "1", "00", "11", "010", "110", "101", "001", "000", "
for test_string in test_strings:
    if is_accepted(test_string):
```

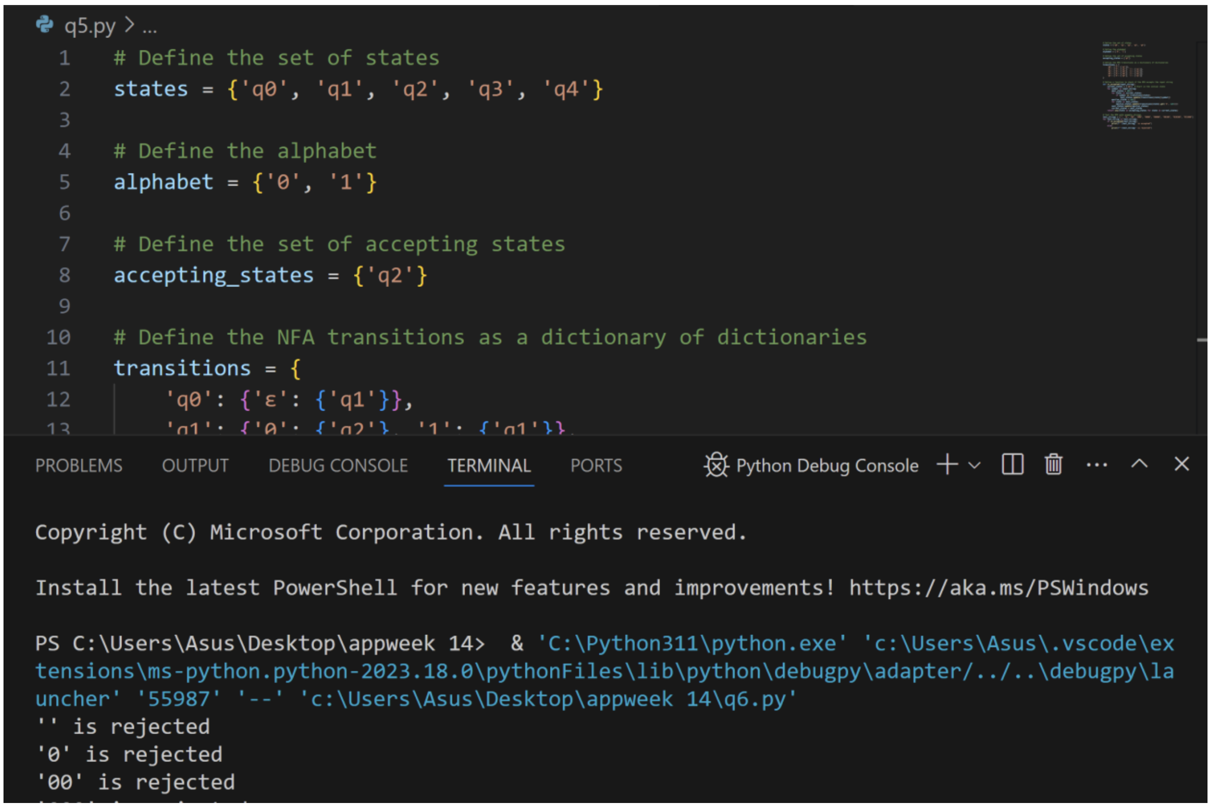
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Asus\Desktop\appweek 14>  & 'C:\Python311\python.exe' 'c:\Users\Asus\.vscode\ex
tensions\ms-python.python-2023.18.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\la
uncher' '54092' '--' 'c:\Users\Asus\Desktop\appweek 14\q4.py'
Transition Table:
State   | Input '0' | Input '1'
--------------------------------
q0       | q1         | q2
q4       | q4         | q4
q2       | q2         | q4
q3       | q1         | q5
q5       | q3         | q5
q1       | q3         | q2
```



```python
    # Function to check if a string is accepted by the DFA
    def is_accepted(input_string):
        current_state = start_state
        for symbol in input_string:
            if symbol not in alphabet:
                return False  # Reject if the symbol is not in the alphabet
            current_state = transition_table[current_state][symbol]
        return current_state in accepting_states

    # Test the DFA with some example strings
    test_strings = ["", "0", "1", "00", "11", "010", "110", "101", "001", "000", "
    for test_string in test_strings:
        if is_accepted(test_string):
```

```
q1       | q3          | q2
'' is rejected
'0' is rejected
'1' is rejected
'00' is rejected
'11' is accepted
'010' is rejected
'110' is accepted
'101' is accepted
'001' is accepted
'000' is rejected
'111' is accepted
'1111' is accepted
PS C:\Users\Asus\Desktop\appweek 14> 
```

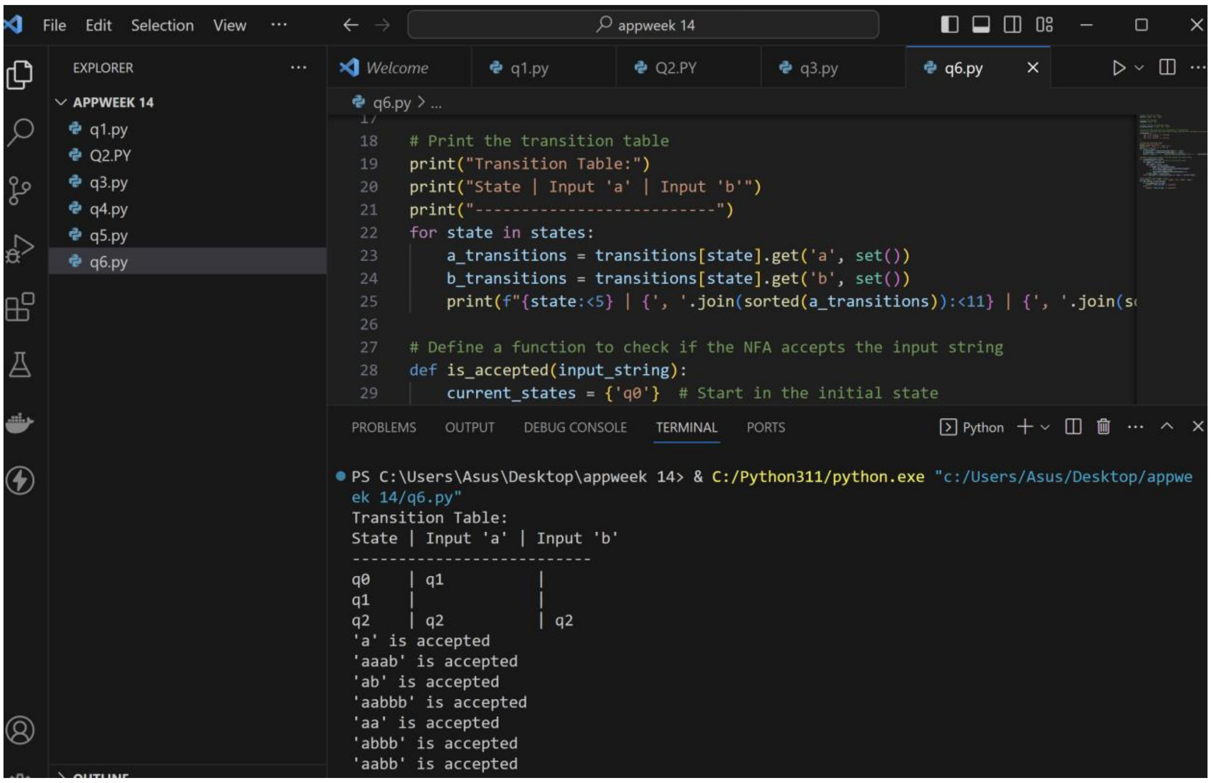
Q5)



```python
# Define the set of states
states = {'q0', 'q1', 'q2', 'q3', 'q4'}

# Define the alphabet
alphabet = {'0', '1'}

# Define the set of accepting states
accepting_states = {'q2'}

# Define the NFA transitions as a dictionary of dictionaries
transitions = {
    'q0': {'ε': {'q1'}},
    'q1': {'0': {'q2'}, '1': {'q1'}},
```

```
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Asus\Desktop\appweek 14>  & 'C:\Python311\python.exe' 'c:\Users\Asus\.vscode\ex
tensions\ms-python.python-2023.18.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\la
uncher' '55987' '--' 'c:\Users\Asus\Desktop\appweek 14\q6.py'
'' is rejected
'0' is rejected
'00' is rejected
```

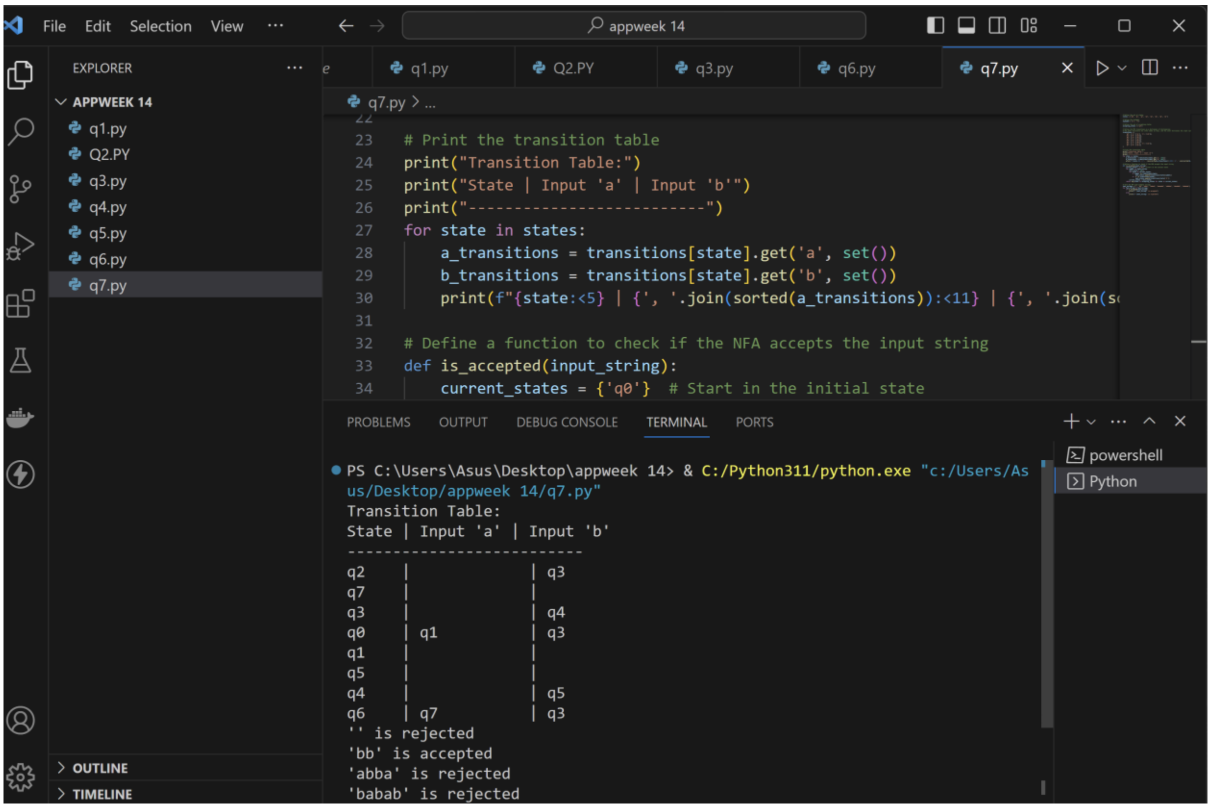
Q6)



```python
# Print the transition table
print("Transition Table:")
print("State | Input 'a' | Input 'b'")
print("-------------------------")
for state in states:
    a_transitions = transitions[state].get('a', set())
    b_transitions = transitions[state].get('b', set())
    print(f"{state:<5} | {', '.join(sorted(a_transitions)):<11} | {', '.join(s

# Define a function to check if the NFA accepts the input string
def is_accepted(input_string):
    current_states = {'q0'}  # Start in the initial state
```

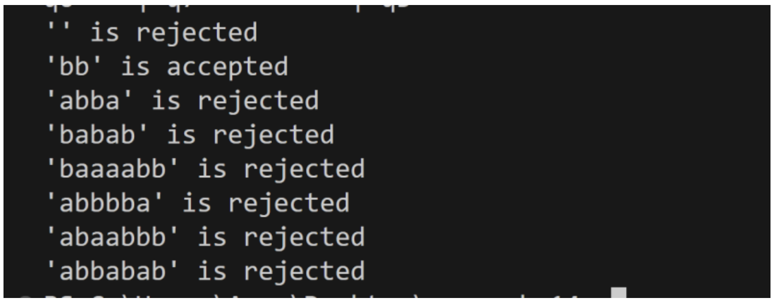
```
PS C:\Users\Asus\Desktop\appweek 14> & C:/Python311/python.exe "c:/Users/Asus/Desktop/appwe
ek 14/q6.py"
Transition Table:
State | Input 'a' | Input 'b'
-------------------------
q0    | q1          |
q1    |             |
q2    | q2          | q2
'a' is accepted
'aaab' is accepted
'ab' is accepted
'aabbb' is accepted
'aa' is accepted
'abbb' is accepted
'aabb' is accepted
```

Q7)

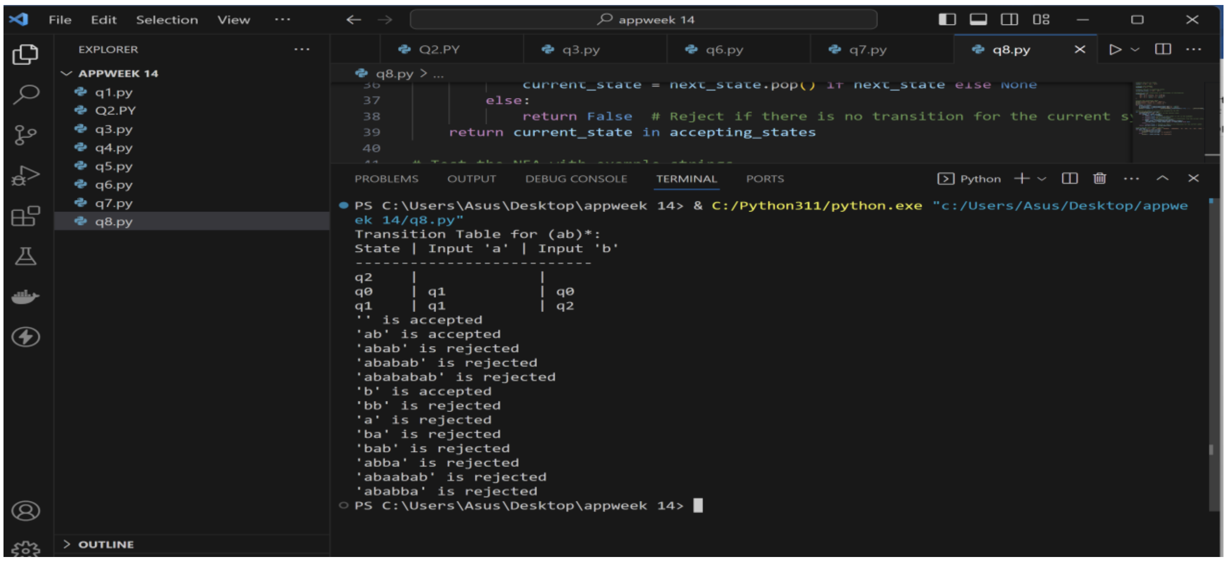


```python
# Print the transition table
print("Transition Table:")
print("State | Input 'a' | Input 'b'")
print("-------------------------")
for state in states:
    a_transitions = transitions[state].get('a', set())
    b_transitions = transitions[state].get('b', set())
    print(f"{state:<5} | {', '.join(sorted(a_transitions)):<11} | {', '.join(s

# Define a function to check if the NFA accepts the input string
def is_accepted(input_string):
    current_states = {'q0'}  # Start in the initial state
```

```
PS C:\Users\Asus\Desktop\appweek 14> & C:/Python311/python.exe "c:/Users/As
us/Desktop/appweek 14/q7.py"
Transition Table:
State | Input 'a' | Input 'b'
-------------------------
q2    |             | q3
q7    |             |
q3    |             | q4
q0    | q1          | q3
q1    |             |
q5    |             |
q4    |             | q5
q6    | q7          | q3
'' is rejected
'bb' is accepted
'abba' is rejected
'babab' is rejected
```


```
'' is rejected
'bb' is accepted
'abba' is rejected
'babab' is rejected
'baaaabb' is rejected
'abbbba' is rejected
'abaabbb' is rejected
'abbabab' is rejected
```

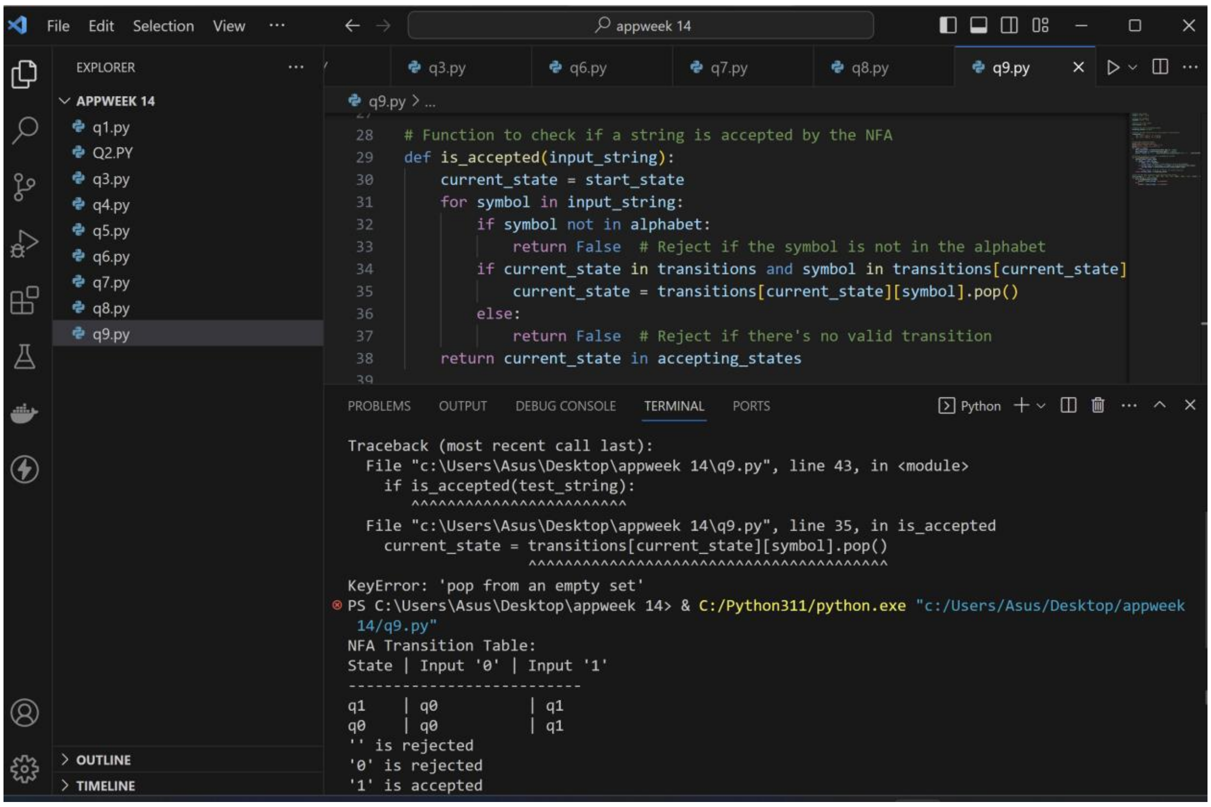
Q8)



```
            current_state = next_state.pop() if next_state else None
        else:
            return False  # Reject if there is no transition for the current s
    return current_state in accepting_states
```

```
PS C:\Users\Asus\Desktop\appweek 14> & C:/Python311/python.exe "c:/Users/Asus/Desktop/appwe
ek 14/q8.py"
Transition Table for (ab)*:
State | Input 'a' | Input 'b'
--------------------------
q2    |           |
q0    | q1        | q0
q1    | q1        | q2
'' is accepted
'ab' is accepted
'abab' is rejected
'ababab' is rejected
'abababab' is rejected
'b' is accepted
'bb' is rejected
'a' is rejected
'ba' is rejected
'bab' is rejected
'abba' is rejected
'abaabab' is rejected
'ababba' is rejected
PS C:\Users\Asus\Desktop\appweek 14>
```

Q9)



```python
# Function to check if a string is accepted by the NFA
def is_accepted(input_string):
    current_state = start_state
    for symbol in input_string:
        if symbol not in alphabet:
            return False  # Reject if the symbol is not in the alphabet
        if current_state in transitions and symbol in transitions[current_state]
            current_state = transitions[current_state][symbol].pop()
        else:
            return False  # Reject if there's no valid transition
    return current_state in accepting_states
```

```
Traceback (most recent call last):
  File "c:\Users\Asus\Desktop\appweek 14\q9.py", line 43, in <module>
    if is_accepted(test_string):
       ^^^^^^^^^^^^^^^^^^^^^^^^
  File "c:\Users\Asus\Desktop\appweek 14\q9.py", line 35, in is_accepted
    current_state = transitions[current_state][symbol].pop()
                    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
KeyError: 'pop from an empty set'
PS C:\Users\Asus\Desktop\appweek 14> & C:/Python311/python.exe "c:/Users/Asus/Desktop/appweek
 14/q9.py"
NFA Transition Table:
State | Input '0' | Input '1'
--------------------------
q1    | q0         | q1
q0    | q0         | q1
'' is rejected
'0' is rejected
'1' is accepted
```