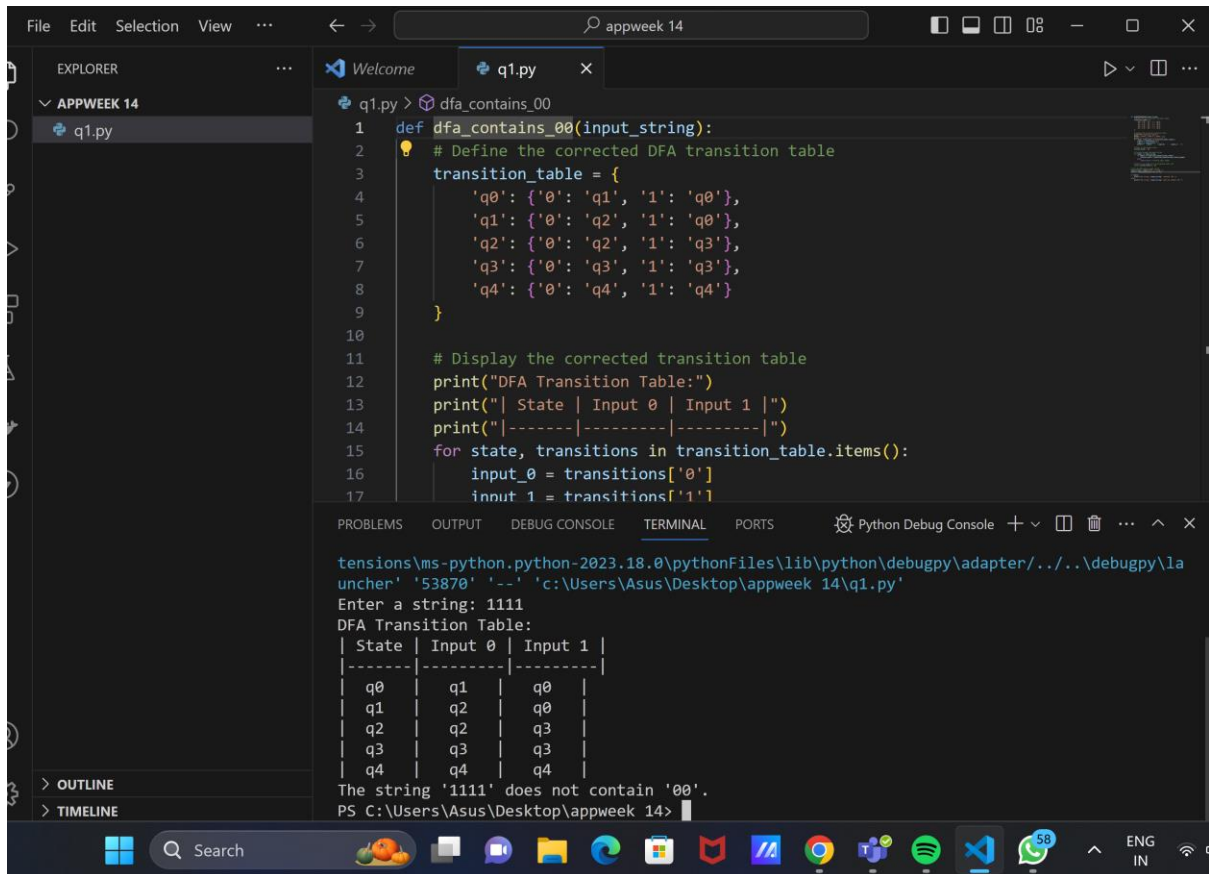


APP WEEK 14

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



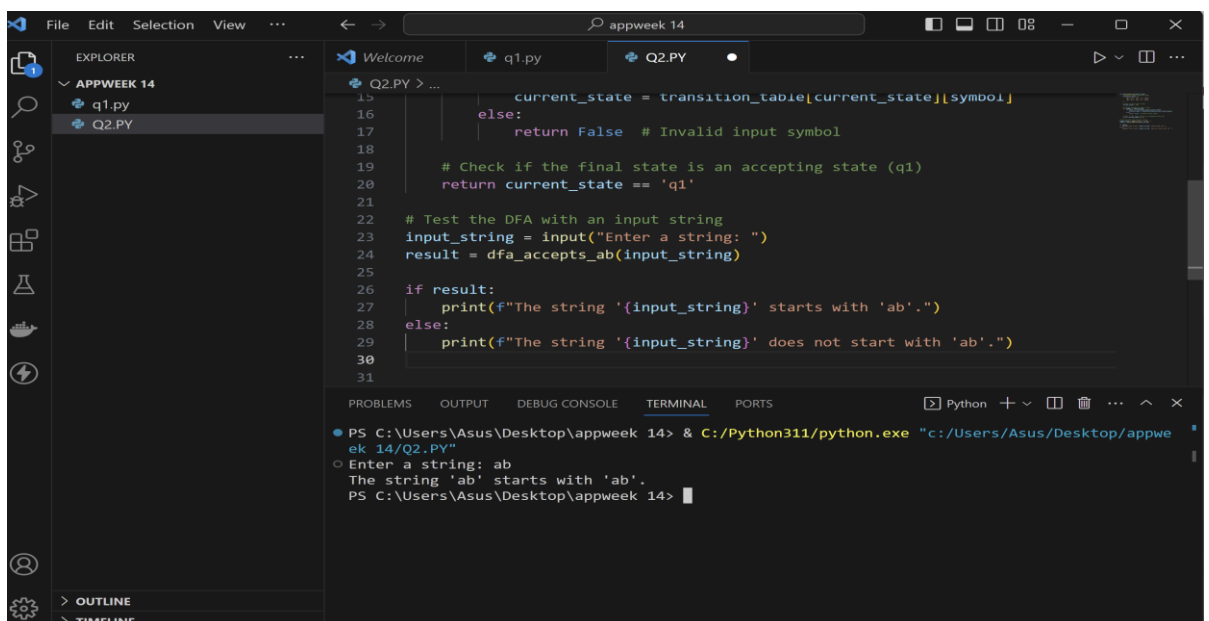
The screenshot shows a VS Code editor with a file named `q1.py`. The code defines a function `dfa_contains_00` that takes an `input_string` and returns a boolean. It defines a transition table for a DFA with 5 states (q0 to q4). The terminal output shows the transition table and the result for the input string '1111'.

```
1 def dfa_contains_00(input_string):
2     # Define the corrected DFA transition table
3     transition_table = {
4         'q0': {'0': 'q1', '1': 'q0'},
5         'q1': {'0': 'q2', '1': 'q0'},
6         'q2': {'0': 'q2', '1': 'q3'},
7         'q3': {'0': 'q3', '1': 'q3'},
8         'q4': {'0': 'q4', '1': 'q4'}
9     }
10
11     # Display the corrected transition table
12     print("DFA Transition Table:")
13     print("| State | Input 0 | Input 1 |")
14     print("|-----|-----|-----|")
15     for state, transitions in transition_table.items():
16         input_0 = transitions['0']
17         input_1 = transitions['1']
```

Terminal Output:

```
tensions\ms-python.python-2023.18.0\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher '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 transition table for the language accepting strings starting with 'ab' over input alphabets $\Sigma = \{a, b\}$.



The screenshot shows a VS Code editor with two files: `q1.py` and `Q2.PY`. The code in `Q2.PY` defines a function `dfa_accepts_ab` that takes an `input_string` and returns a boolean. It defines a transition table for a DFA with 3 states (q0 to q2). The terminal output shows the result for the input string 'ab'.

```
15     current_state = transition_table[current_state][symbol]
16     else:
17         return False # Invalid input symbol
18
19     # Check if the final state is an accepting state (q1)
20     return current_state == 'q1'
21
22 # Test the DFA with an input string
23 input_string = input("Enter a string: ")
24 result = dfa_accepts_ab(input_string)
25
26 if result:
27     print(f"The string '{input_string}' starts with 'ab'.")
28 else:
29     print(f"The string '{input_string}' does not start with 'ab'.")
30
31
```

Terminal Output:

```
PS C:\Users\Asus\Desktop\appweek 14> & C:/Python311/python.exe "c:\Users\Asus\Desktop\appweek 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 $\Sigma = \{a, b\}$ accepts the only input "aaab".--

The screenshot shows a VS Code editor with a file named 'q3.py' open. The code defines a DFA with states 'q0', 'q1', 'q2', 'q3', and 'q4'. The initial state is 'q0' and the final state is 'q4'. The transitions are defined as follows:

- q0 to q1 on 'a'
- q1 to q2 on 'a'
- q2 to q3 on 'a'
- q3 to q4 on 'b'
- q3 to q0 on 'a'
- q4 to q0 on 'a'
- q4 to q0 on 'b'

The script tests the DFA with the input string "aaab" and prints "The input 'aaab' is accepted by the DFA." The terminal output shows the command executed and the result.

```

10 current_state = 'q0'
11 elif current_state == 'q3' and symbol == 'b':
12     current_state = 'q4'
13 else:
14     return False
15
16 return current_state == 'q4'
17
18 # Input string to test
19 input_string = "aaab"
20
21 if is_accepted(input_string):
22     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/appweek 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 transition table for the language $L(M) = a + aa^*b$. 5.

Draw a deterministic finite automata which accept 00 and 11 at the end of a string containing 0, 1.

The screenshot shows a VS Code editor with a file named 'q4.py' open. The code defines a DFA with states 'q0', 'q1', 'q2', 'q3', 'q4', 'q5', and 'q6'. The initial state is 'q0' and the final states are 'q4' and 'q6'. The transitions are defined as follows:

- q0 to q1 on '0'
- q0 to q2 on '1'
- q1 to q3 on '0'
- q1 to q4 on '1'
- q2 to q5 on '0'
- q2 to q6 on '1'
- q3 to q1 on '0'
- q3 to q4 on '1'
- q4 to q0 on '0'
- q4 to q0 on '1'
- q5 to q2 on '0'
- q5 to q6 on '1'
- q6 to q0 on '0'
- q6 to q0 on '1'

The script tests the DFA with some example strings and prints the results. The terminal output shows the command executed and the results.

```

31 # Function to check if a string is accepted by the DFA
32 def is_accepted(input_string):
33     current_state = start_state
34     for symbol in input_string:
35         if symbol not in alphabet:
36             return False # Reject if the symbol is not in the alphabet
37         current_state = transition_table[current_state][symbol]
38     return current_state in accepting_states
39
40 # Test the DFA with some example strings
41 test_strings = ["", "0", "1", "00", "11", "010", "110", "101", "001", "000", "111"]
42 for test_string in test_strings:
43     if is_accepted(test_string):

```

```

PS C:\Users\Asus\Desktop\appweek 14> & 'C:\Python311\python.exe' 'c:\Users\Asus\.vscode\extensions\ms-python.python-2023.18.0\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '54092' '--' 'c:\Users\Asus\Desktop\appweek 14\q4.py'
Transition Table:
State | Input '0' | Input '1'
-----|-----|-----
q0    | q1        | q2
q1    | q3        | q4
q2    | q5        | q6
q3    | q1        | q4
q4    | q0        | q0
q5    | q2        | q6
q6    | q0        | q0

```

The screenshot shows a VS Code editor with a file named `q4.py` open. The code defines a function `is_accepted` to check if a string is accepted by a DFA. It also includes a list of test strings and a loop to print the results. The terminal output shows the results of the tests for various strings.

```

31 # function to check if a string is accepted by the DFA
32 def is_accepted(input_string):
33     current_state = start_state
34     for symbol in input_string:
35         if symbol not in alphabet:
36             return False # Reject if the symbol is not in the alphabet
37         current_state = transition_table[current_state][symbol]
38     return current_state in accepting_states
39
40 # Test the DFA with some example strings
41 test_strings = ["", "0", "1", "00", "11", "010", "110", "101", "001", "000", "111", "1111"]
42 for test_string in test_strings:
43     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)

The screenshot shows a VS Code editor with a file named `q5.py` open. The code defines the states, alphabet, accepting states, and transitions for an NFA. The terminal output shows the results of the tests for various strings.

```

1 # Define the set of states
2 states = {'q0', 'q1', 'q2', 'q3', 'q4'}
3
4 # Define the alphabet
5 alphabet = {'0', '1'}
6
7 # Define the set of accepting states
8 accepting_states = {'q2'}
9
10 # Define the NFA transitions as a dictionary of dictionaries
11 transitions = {
12     'q0': {'ε': {'q1'}},
13     '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\extensions\ms-python.python-2023.18.0\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '55987' '--' 'c:\Users\Asus\Desktop\appweek 14\q6.py'
'' is rejected
'0' is rejected
'00' is rejected

```

Q6)

```

17
18 # Print the transition table
19 print("Transition Table:")
20 print("State | Input 'a' | Input 'b'")
21 print("-----")
22 for state in states:
23     a_transitions = transitions[state].get('a', set())
24     b_transitions = transitions[state].get('b', set())
25     print(f"{state:<5} | {' '.join(sorted(a_transitions)):<11} | {' '.join(sorted(b_transitions)):<11}")
26
27 # Define a function to check if the NFA accepts the input string
28 def is_accepted(input_string):
29     current_states = {'q0'} # Start in the initial state

```

PS C:\Users\Asus\Desktop\appweek 14> & C:/Python311/python.exe "c:/Users/Asus/Desktop/appweek 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
 'aabb' is accepted
 'aa' is accepted
 'abbb' is accepted
 'aabb' is accepted

Q7)

```

23 # Print the transition table
24 print("Transition Table:")
25 print("State | Input 'a' | Input 'b'")
26 print("-----")
27 for state in states:
28     a_transitions = transitions[state].get('a', set())
29     b_transitions = transitions[state].get('b', set())
30     print(f"{state:<5} | {' '.join(sorted(a_transitions)):<11} | {' '.join(sorted(b_transitions)):<11}")
31
32 # Define a function to check if the NFA accepts the input string
33 def is_accepted(input_string):
34     current_states = {'q0'} # Start in the initial state

```

PS C:\Users\Asus\Desktop\appweek 14> & C:/Python311/python.exe "c:/Users/Asus/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)

The screenshot shows a VS Code editor with a project named 'appweek 14'. The file explorer on the left shows a directory 'APPWEEK 14' containing files q1.py, Q2.PY, q3.py, q4.py, q5.py, q6.py, q7.py, and q8.py. The editor is open to q8.py, which contains the following Python code:

```

36     current_state = next_state.pop() if next_state else None
37     else:
38         return False # Reject if there is no transition for the current s
39     return current_state in accepting_states
40
41 # Test the DFA with various strings

```

The terminal output shows the execution of the program:

```

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    | q1        | q0
q0    | q1        | q2
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

```

Q9)

File Edit Selection View ...

appweek 14

EXPLORER

APPWEEK 14

- q1.py
- Q2.PY
- q3.py
- q4.py
- q5.py
- q6.py
- q7.py
- q8.py
- q9.py

q9.py > ...

```
28 # Function to check if a string is accepted by the NFA
29 def is_accepted(input_string):
30     current_state = start_state
31     for symbol in input_string:
32         if symbol not in alphabet:
33             return False # Reject if the symbol is not in the alphabet
34         if current_state in transitions and symbol in transitions[current_state]:
35             current_state = transitions[current_state][symbol].pop()
36         else:
37             return False # Reject if there's no valid transition
38     return current_state in accepting_states
39
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Python + v

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

> OUTLINE
> TIMELINE