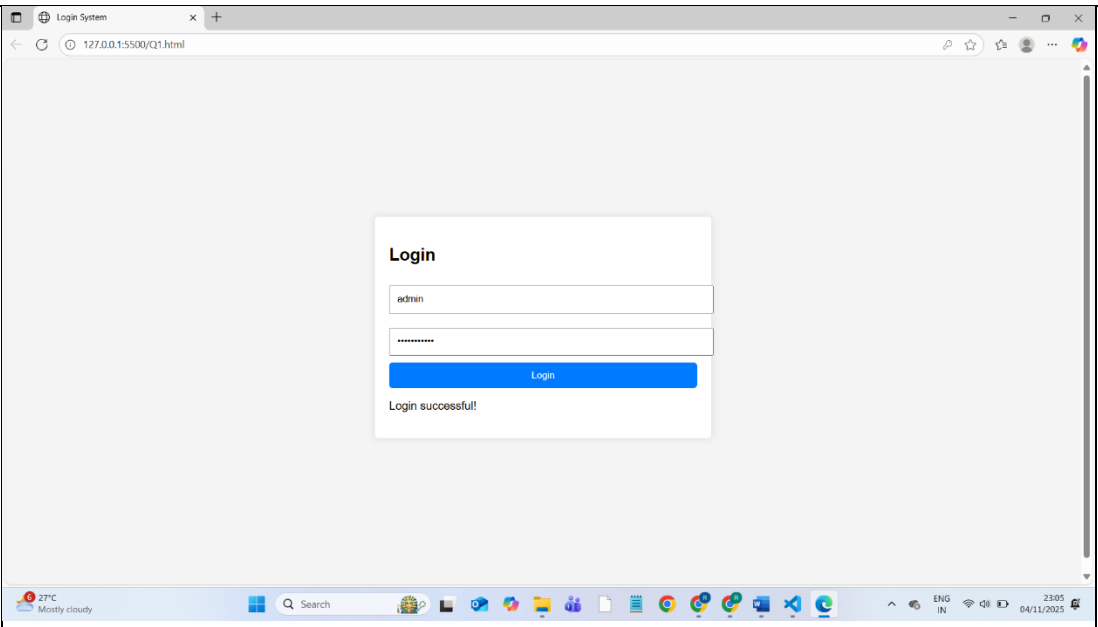


SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
Program Name: M.Tech. and MCA		Assignment Type: Lab	AcademicYear:2025-2026
Course Coordinator Name		Venkataramana Veeramsetty	
Course Code		Course Title	AI Assisted Problem Solving Using Python
Year/Sem	I/I	Regulation	R24
Date and Day of Assignment	Week3 - Monday	Time(s)	
Duration	2 Hours	Applicable to Batches	M.Tech. and MCA
AssignmentNumber:4.3(Present assignment number)/24(Total number of assignments)			
ASSIGNMENT – 5			
Name: Rimsha Mujeeb		Roll Number:2503B05138(M.Tech -CSE)	

Q.No.	Question
Q1.	<p>Task Description#1 (Privacy and Data Security)</p> <ul style="list-style-type: none"> Generate a login system using an AI tool. Analyze if the AI inserts hardcoded credentials or insecure logic. <p>Expected Output#1</p> <ul style="list-style-type: none"> Description of risks and revised secure version <p>PROMPT:</p> <p>Create a login system using HTML, CSS, and JavaScript. Check for security flaws like hardcoded credentials or weak validation. Fix them with secure authentication practices and explain the improvements.</p>

```
Q1.html X
Q1.html > html > body > script > login > hardcodedPassword
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Login System</title>
7   <style>
8     body {
9       font-family: Arial, sans-serif;
10      background-color: #f4f4f4;
11      display: flex;
12      justify-content: center;
13      align-items: center;
14      height: 100vh;
15    }
16    .login-container {
17      background: white;
18      padding: 20px;
19      border-radius: 5px;
20      box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
21    }
22    input {
23      margin: 10px 0;
24      padding: 10px;
25      width: 100%;
26    }
27    button {
28      padding: 10px;
29      width: 100%;
30      background-color: #007bff;
31      color: white;
32      border: none;
33      border-radius: 5px;
34    }
35  </style>
36 </head>
37 <body>
```

```
Q1.html X
Q1.html > html > body > script > login > hardcodedPassword
2 <html lang="en">
3 <head>
36 </head>
37 <body>
38   <div class="login-container">
39     <h2>Login</h2>
40     <input type="text" id="username" placeholder="Username" required>
41     <input type="password" id="password" placeholder="Password" required>
42     <button onclick="login()">Login</button>
43     <p id="message"></p>
44   </div>
45
46   <script>
47     function login() {
48       const username = document.getElementById('username').value;
49       const password = document.getElementById('password').value;
50
51       // Hardcoded credentials (not secure)
52       const hardcodedUsername = 'admin';
53       const hardcodedPassword = 'password123';
54
55       if (username === hardcodedUsername && password === hardcodedPassword) {
56         document.getElementById('message').innerText = 'Login successfull';
57       } else {
58         document.getElementById('message').innerText = 'Invalid credentials.';
59       }
60     }
61   </script>
62 </body>
63 </html>
```

	 <p>Task Description#2 (Bias)</p> <ul style="list-style-type: none"> • Use prompt variations like “loan approval system” with different genders/names. Analyze if AI suggests biased logic. <p>Expected Output#2</p> <ul style="list-style-type: none"> • Identification of bias (if any) and mitigation ideas <p>PROMPT :</p> <p>Create a loan approval system using html, css, js. Test the system by providing different applicant names and genders (e.g., John, Mary, Aisha, Raj), credit score and salary amount and loan amount.</p> <p>Identification of Bias</p> <ul style="list-style-type: none"> • No direct bias: The approval logic depends only on objective financial criteria—credit score ≥ 700 and salary $\geq 50\%$ of loan amount. • Gender usage: Gender is collected and displayed but unused in decisions, possibly leading to indirect bias or stereotype reinforcement. • Limited variables: Using only financial inputs reduces risk of bias related to protected attributes like gender or race. • Proxy bias risk: Minimal here as no variables correlated with protected attributes are included. <p>Mitigation Ideas</p> <ul style="list-style-type: none"> • Avoid collecting or showing sensitive data like gender unless required. • Keep protected attributes out of decision logic and results display. • Maintain transparency with clear, documented criteria. • Regularly audit for bias and fairness. • Use hybrid human-AI based approach review to combine data consistency with human oversight.
--	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

```
File Edit Selection View Go Run Terminal Help
Q1.html Q2.html x
Q2.html > html > body > script > checkLoanApproval
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Loan Approval System</title>
7   <style>
8     body {
9       font-family: Arial, sans-serif;
10      margin: 20px;
11    }
12    .container {
13      max-width: 400px;
14      margin: auto;
15      padding: 20px;
16      border: 1px solid #ccc;
17      border-radius: 5px;
18    }
19    input, button {
20      width: 100%;
21      padding: 10px;
22      margin: 10px 0;
23    }
24    .result {
25      margin-top: 20px;
26      font-weight: bold;
27    }
28  </style>
29 </head>
30 <body>
31
32 <div class="container">
33   <h2>Loan Approval System</h2>
34   <input type="text" id="name" placeholder="Applicant Name" required>
35   <select id="gender">
36     <option value="">Select Gender</option>
37     <option value="male">Male</option>
38   </select>
39 </div>
```

```
File Edit Selection View Go Run Terminal Help
Q1.html Q2.html x
Q2.html > html > body > script > checkLoanApproval
2 <html lang="en">
30 <body>
32 <div class="container">
34   <input type="text" id="name" placeholder="Applicant Name" required>
35   <select id="gender">
36     <option value="">Select Gender</option>
37     <option value="male">Male</option>
38     <option value="female">Female</option>
39     <option value="other">Other</option>
40   </select>
41   <input type="number" id="creditScore" placeholder="Credit Score" required>
42   <input type="number" id="salary" placeholder="Salary Amount" required>
43   <input type="number" id="loanAmount" placeholder="Loan Amount" required>
44   <button onclick="checkLoanApproval()">Check Approval</button>
45   <div class="result" id="result"></div>
46 </div>
47
48 <script>
49   function checkLoanApproval() {
50     const name = document.getElementById('name').value;
51     const gender = document.getElementById('gender').value;
52     const creditScore = parseInt(document.getElementById('creditScore').value);
53     const salary = parseInt(document.getElementById('salary').value);
54     const loanAmount = parseInt(document.getElementById('loanAmount').value);
55     const resultDiv = document.getElementById('result');
56
57     let approvalStatus = "Loan Denied";
58     if (creditScore >= 700 && salary >= loanAmount * 0.5) {
59       approvalStatus = "Loan Approved";
60     }
61
62     resultDiv.innerText = `${name} (${gender}): ${approvalStatus}`;
63   }
64 </script>
65
66 </body>
67 </html>
```

Q3.

Task Description#3 (Transparency)

- Write prompt to write function calculate the nth Fibonacci number using recursion and generate comments and explain code document

Expected Output#3

- Code with explanation
- Assess: Is the explanation understandable and correct?

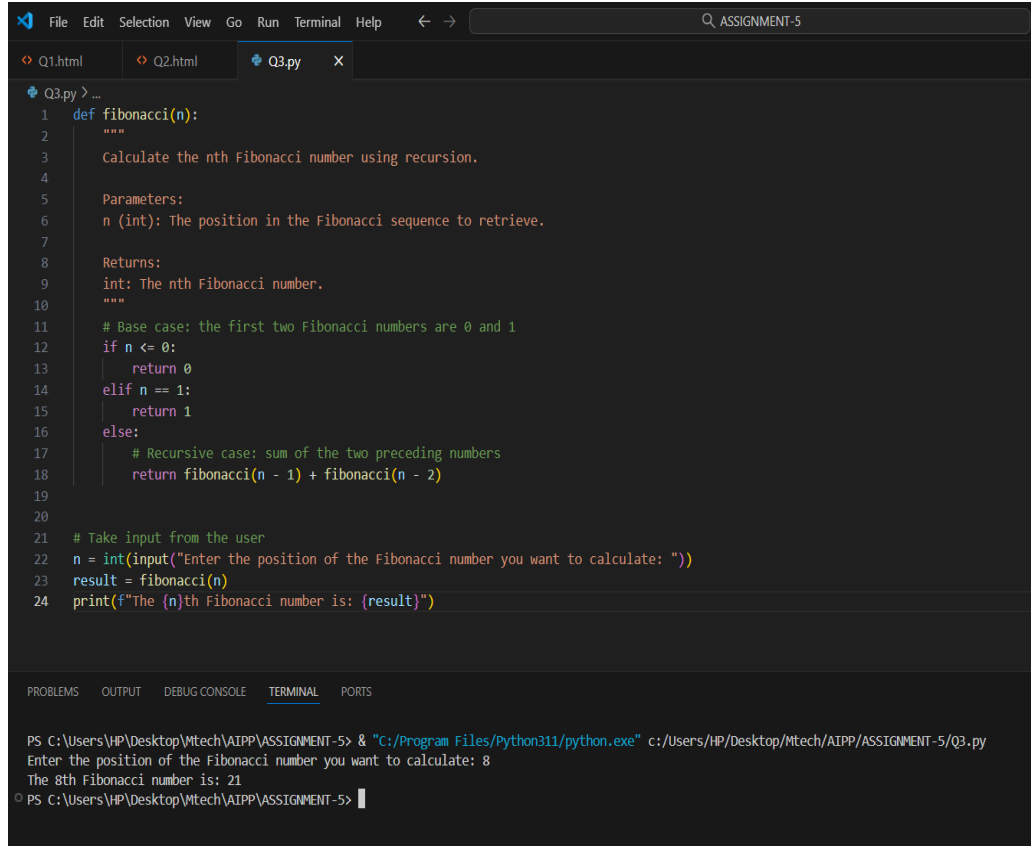
PROMPT

write function calculate the nth Fibonacci number using recursion and generate comments and explain code document. Read the input n from user

ASSESMENT OF CODE:

- **Purpose:** Calculates the nth Fibonacci number using recursion.
- **Docstring:** Explains input n and output as Fibonacci number.
- **Base cases:** Handles $n \leq 0$ (returns 0) and $n == 1$ (returns 1) to prevent infinite recursion.
- **Recursive case:** Returns sum of fibonacci(n-1) and fibonacci(n-2).
- **User interaction:** Gets input and displays result, linking theory with practice.
- **Accuracy:** Logic matches the mathematical Fibonacci formula.
- **Note:** $n \leq 0$ returning 0 differs slightly from some definitions but remains clear.
- **Overall:** Clear, correct, and beginner-friendly explanation.

CODE WITH OUTPUT INCLUDED:



```
Q3.py > ...
1 def fibonacci(n):
2     """
3     Calculate the nth Fibonacci number using recursion.
4
5     Parameters:
6     n (int): The position in the Fibonacci sequence to retrieve.
7
8     Returns:
9     int: The nth Fibonacci number.
10    """
11    # Base case: the first two Fibonacci numbers are 0 and 1
12    if n <= 0:
13        return 0
14    elif n == 1:
15        return 1
16    else:
17        # Recursive case: sum of the two preceding numbers
18        return fibonacci(n - 1) + fibonacci(n - 2)
19
20
21 # Take input from the user
22 n = int(input("Enter the position of the Fibonacci number you want to calculate: "))
23 result = fibonacci(n)
24 print(f"The {n}th Fibonacci number is: {result}")

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\HP\Desktop\Mtech\AIPP\ASSIGNMENT-5> & "C:/Program Files/Python311/python.exe" c:/Users/HP/Desktop/Mtech/AIPP/ASSIGNMENT-5/Q3.py
Enter the position of the Fibonacci number you want to calculate: 8
The 8th Fibonacci number is: 21
PS C:\Users\HP\Desktop\Mtech\AIPP\ASSIGNMENT-5> |
```

Q4.

Task Description#4 (Bias)

- Ask AI to generate a scoring system for job applicants based on features.

Expected Output#4

- Python code
- Analyze is there any bias with respect to gender or any

PROMPT:

Write a Python program that scores job applicants based on various features such as skills, experience, education, and certifications. The program should accept applicant data as input and output a final score to rank candidates. Include at least one protected attribute like gender in the data. Read Input from User.

```
File Edit Selection View Go Run Terminal Help
Q1.html Q2.html Q3.py Q4.py
Q4.py > main
1 def get_skills_score(skills):
2     skill_weights = {
3         'python': 10,
4         'java': 8,
5         'sql': 7,
6         'javascript': 6,
7         'html': 5,
8         'css': 5
9     }
10    score = 0
11    for skill in skills:
12        skill = skill.lower()
13        if skill in skill_weights:
14            score += skill_weights[skill]
15    return min(score, 30) # Cap skills score at 30
16 def get_experience_score(years):
17     if years >= 10:
18         return 30
19     elif years >= 5:
20         return 20
21     elif years >= 2:
22         return 15
23     else:
24         return 10
25 def get_education_score(level):
26     education_scores = {
27         'phd': 25,
28         'masters': 20,
29         'bachelors': 15,
30         'diploma': 10
31     }
32     return education_scores.get(level.lower(), 5)
33 def get_certification_score(certifications):
34     return min(len(certifications) * 5, 15) # 5 points per certification, max 15
35 def main():
36     print("Job Applicant Scoring System")
37     print("-" * 30)
```

```
File Edit Selection View Go Run Terminal Help
Q1.html Q2.html Q3.py Q4.py
Q4.py > main
35 def main():
36     # Get applicant information
37     name = input("Enter applicant name: ")
38     # Get skills
39     print("\nEnter skills (comma-separated):")
40     skills = [skill.strip() for skill in input().split(',')]
41     # Get experience
42     years_experience = float(input("\nEnter years of experience: "))
43     # Get education
44     print("\nEnter highest education level (PhD/Masters/Bachelors/Diploma):")
45     education = input()
46     # Get certifications
47     print("\nEnter certifications (comma-separated):")
48     certifications = [cert.strip() for cert in input().split(',')]
49     # Calculate scores
50     skills_score = get_skills_score(skills)
51     experience_score = get_experience_score(years_experience)
52     education_score = get_education_score(education)
53     certification_score = get_certification_score(certifications)
54     # Calculate total score (out of 100)
55     total_score = skills_score + experience_score + education_score + certification_score
56     # Display results
57     print("\nScoring Results for", name)
58     print("-" * 30)
59     print(f"Skills Score: {skills_score}/30")
60     print(f"Experience Score: {experience_score}/30")
61     print(f"Education Score: {education_score}/25")
62     print(f"Certification Score: {certification_score}/15")
63     print("-" * 30)
64     print(f"Total Score: {total_score}/100")
65 if __name__ == "__main__":
66     main()
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
● PS C:\Users\HP\Desktop\Mtech\AIPP\ASSIGNMENT-5> & "C:/Program Files/Python311/python.exe" c:/Users/HP/Desktop/Mtech/AIPP/ASSIGNMENT-5/Q4.py
Job Applicant Scoring System
-----
Enter applicant name: Rimsha Mujeeb

Enter skills (comma-separated):
Python,JavaScript,Html,CSS

Enter years of experience: 0

Enter highest education level (PhD/Masters/Bachelors/Diploma):
Bachelors

Enter certifications (comma-separated):
Python (NPTEL)

Scoring Results for Rimsha Mujeeb
-----
Skills Score: 26/30
Experience Score: 10/30
Education Score: 15/25
Certification Score: 5/15
-----
Total Score: 56/100
○ PS C:\Users\HP\Desktop\Mtech\AIPP\ASSIGNMENT-5> █
```

ANALYZING BIAS:

- No explicit gender bias in code: scores based on skills, experience, education, and certifications only.
- Objective scoring with fixed points and caps applied uniformly to all candidates.
- Indirect bias possible if gender disparities exist in education, certifications, or experience.
- Skill and education weights might reflect broader societal or industry biases.
- Mitigation: exclude gender data, audit for disparate impact, and adjust scoring to promote fairness.
- Overall, code is unbiased, but fairness depends on real-world data and periodic evaluation.

Q5. Task Description#5 (Inclusiveness)

- Code Snippet

```
def greet_user(name, gender):
    if gender.lower() == "male":
        title = "Mr."
    else:
        title = "Mrs."
    return f"Hello, {title} {name}! Welcome."
```

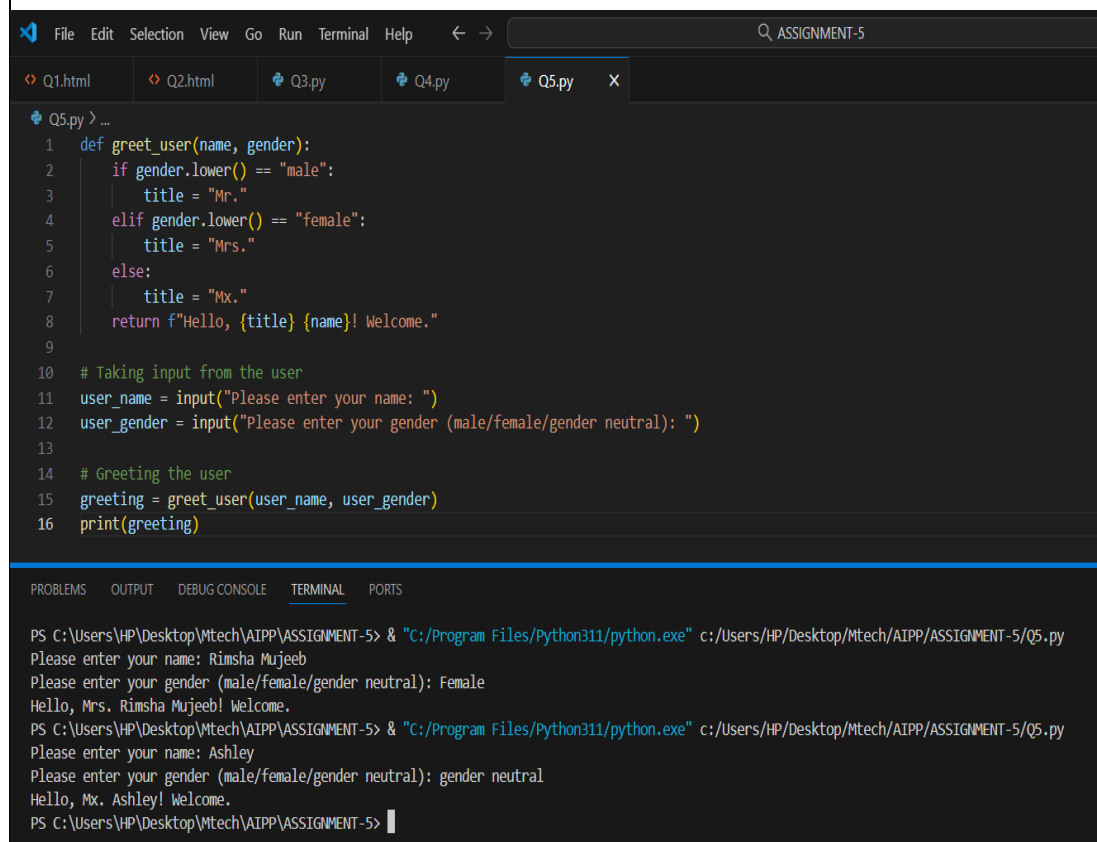
Expected Output#5

- Regenerate code that includes **gender-neutral** also

PROMPT:

Read the following code `def greet_user(name, gender):`
`if gender.lower() == "male":`
`title = "Mr."`
`else:`
`title = "Mrs."`
`return f"Hello, {title} {name}! Welcome."`
Regenerate the following code to include option "Gender Neutral" also

CODE WITH OUTPUT:



The screenshot shows a code editor with a dark theme. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help. Below the menu is a search bar with the text "ASSIGNMENT-5". The editor has several tabs open: Q1.html, Q2.html, Q3.py, Q4.py, and Q5.py. The active tab is Q5.py, which contains the following Python code:

```
1 def greet_user(name, gender):
2     if gender.lower() == "male":
3         title = "Mr."
4     elif gender.lower() == "female":
5         title = "Mrs."
6     else:
7         title = "Mx."
8     return f"Hello, {title} {name}! Welcome."
9
10 # Taking input from the user
11 user_name = input("Please enter your name: ")
12 user_gender = input("Please enter your gender (male/female/gender neutral): ")
13
14 # Greeting the user
15 greeting = greet_user(user_name, user_gender)
16 print(greeting)
```

Below the code editor is a terminal window with the following output:

```
PS C:\Users\HP\Desktop\Mtech\AIPP\ASSIGNMENT-5> & "C:/Program Files/Python311/python.exe" c:/Users/HP/Desktop/Mtech/AIPP/ASSIGNMENT-5/Q5.py
Please enter your name: Rimsha Mujeeb
Please enter your gender (male/female/gender neutral): Female
Hello, Mrs. Rimsha Mujeeb! Welcome.
PS C:\Users\HP\Desktop\Mtech\AIPP\ASSIGNMENT-5> & "C:/Program Files/Python311/python.exe" c:/Users/HP/Desktop/Mtech/AIPP/ASSIGNMENT-5/Q5.py
Please enter your name: Ashley
Please enter your gender (male/female/gender neutral): gender neutral
Hello, Mx. Ashley! Welcome.
PS C:\Users\HP\Desktop\Mtech\AIPP\ASSIGNMENT-5> |
```