**Tokenization**

-The purpose of the tokenization module is to divide text into individual tokens, which are small units such as words or symbols. Its operations include the input volume into meaningful components for processing or analysis.

Test Cases:

1. Entry: "a, b"

Expected output: ['a', 'b']

2. Input: "q"

Expected output: The program can terminate.

3. Input: ""

Expected output: []

Actual hours:2 hours.

the environment:

Platform: Windows 10

Tools: Visual Studio, a custom tokenization module

Actual hours:50 minutes.

Test data:

Test case 1: "a b c d"

Test case 2: "q"

Test case 3: ""

Expected Results:

Test case 1: ["a b c d"]

Test case 2: The program can terminate.

Test case 3: []

Test execution:

Test result 1: ["a b c d"]

Result 2: Tests the program that is expected to terminate.

The result of the test item 3: []

Actual hours:65 minutes.

**Manipulating**

-The purpose of the manipulation module is to perform string manipulation operations such as concatenating, parsing or modifying strings. Its functionality consists of taking input strings, applying certain operations and returning the result.

Test items:

1. Input: "a"

Expected output: "a"

2. Input: "q"

Expected output: The program terminates.

3. Input: "", "test"

Expected output: "test"

Actual hours:1 hour.

Test environment:

Platform: windows10

Tools: Visual Studio, a custom manipulation module

Actual hours:25 minutes.

Test data:

Test case 1: "hello", "world"

Test case 2: "q"

test case 3: "", "test"

Expected Results:

Test Case 1: "Hello World"

Test case 2: The program terminates.

Test Case 3: "Test"

Test execution:

Test result case 1: "a b"

Test result 2: The program terminates as expected.

Test result case 3: "Test"

Actual hours:80 minutes.

**Conversion**

-The purpose of the conversion module is to handle conversions between different data types, specifically converting numeric strings to integers. Its functionality consists of taking input strings, attempting conversions, and returning results or handling errors.

Test items:

1. Input: "123", "456"

Expected output: 123,456

2. Input: "q"

Expected output: The program terminates.

3. Input: "How are yo?"

Expected output: error/exception handling behavior.

Actual hours: 50mins

Test environment:

Platform: windows11

Tools: Visual Studio, a custom conversion module

Actual hours: 30 minutes

Test data:

Test case 1: "123", "456"

Test case 2: "q"

Test Case 3: "How are you?"

Expected Results: How are you?

Test Case 1: 123, 456

Test Case 2: The program terminates as expected.

Test Case 3: Error/Exception Handling Behavior.

Test execution:

Test result case 1: 123, 456

Test result 2: The program terminates as expected.

Test Result 3: Observed error/exception handling behavior.

Actual hours: 80 minutes

**Fundamental**

-The main purpose of the Fundamental module is to provide essential functions related to string management and character indexing. Its functions include processing non-empty strings, performing character indexing, handling empty strings carefully, and allowing the program to terminate when ``q'' is entered.

Test items:

1. Input: "Canada, USA"

Expected output:

- Character at index 0: C

- Character in index 5: a

2. Input: "q"

Expected output: The program terminates.

3. Input: ""

Expected output: empty string handle.

Actual hours: 25 minutes

Test environment:

Platform: Windows 11

Tools: Visual Studio, a custom Fundamental module

Actual hours:50 minutes

Test data:

Test case 1: "Canada, USA"

Test case 2: "q"

Test case 3: ""

Expected Results:

Test case 1:

- Character at index 0: C

-

- Character in index 5: a

Test Case 2: The program terminates as expected.

Test case 3: Handle empty string.

Test execution:

The result of test case 1:

- Character at index 0: C

- Character in index 5: a

Test result 2: The program terminates as expected.

Test Result 3: Handle empty string encountered.

Actual hours: 30 mins