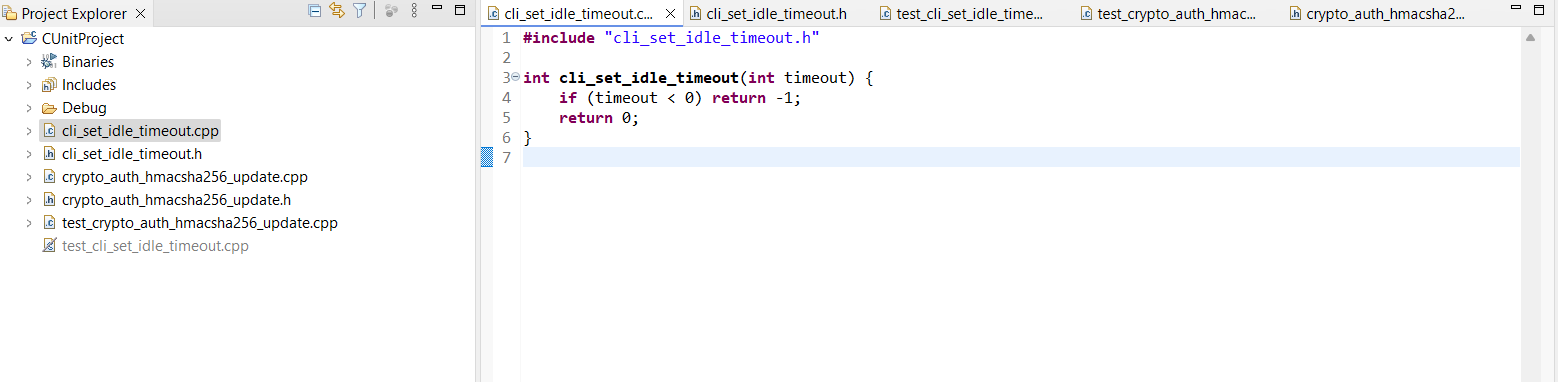
**Set 7: Function Testing**

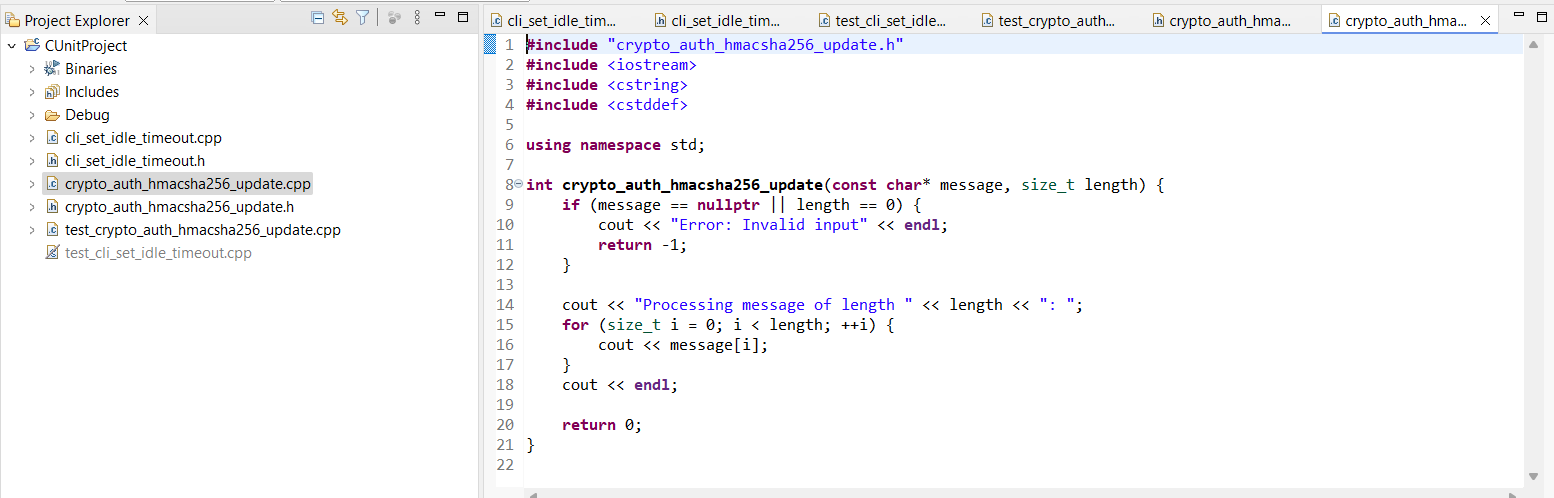
Functions under tests are these 2 :

• cli\_set\_idle\_timeout(int)  
• crypto\_auth\_hmacsha256\_update(const char\*, size\_t)

**✅ 1. Function Implementation**

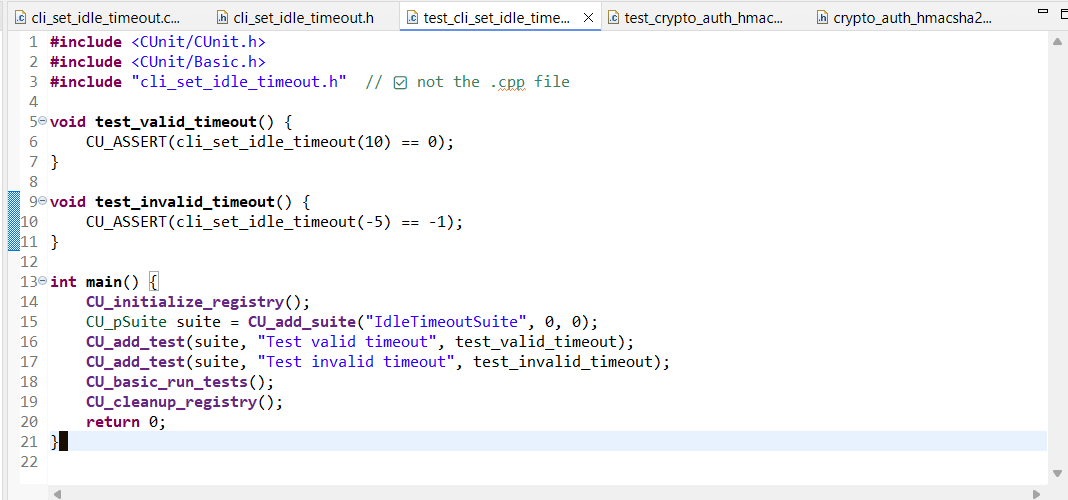
• cli\_set\_idle\_timeout(int)

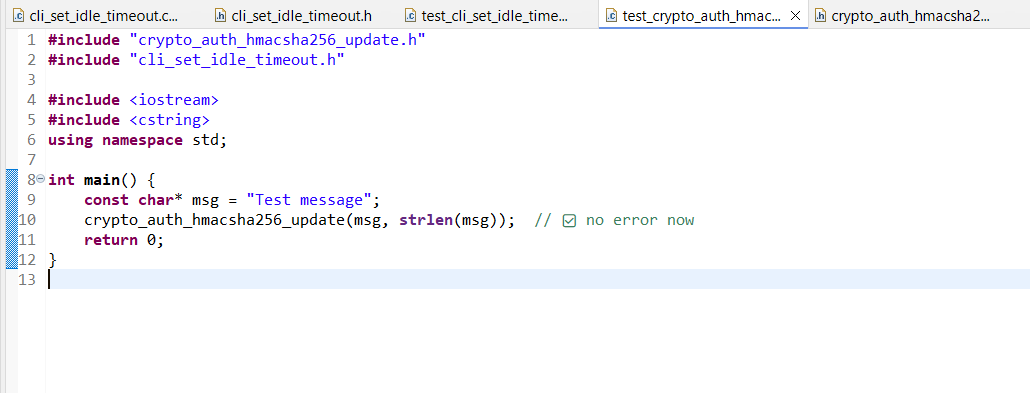
  
• crypto\_auth\_hmacsha256\_update(const char\*, size\_t)



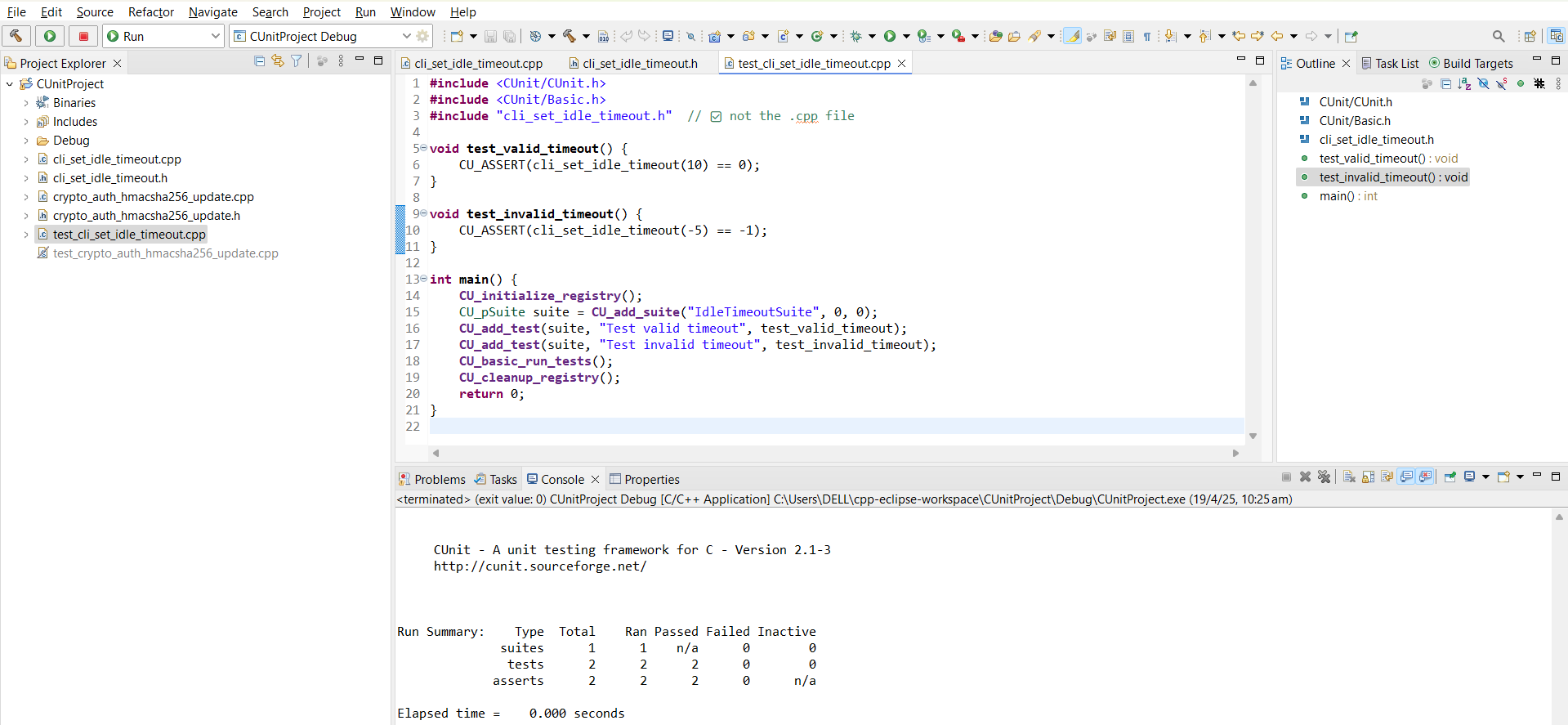
**✅ 2. Test Scripts**

• CUnit test cases were written and executed for cli\_set\_idle\_timeout().

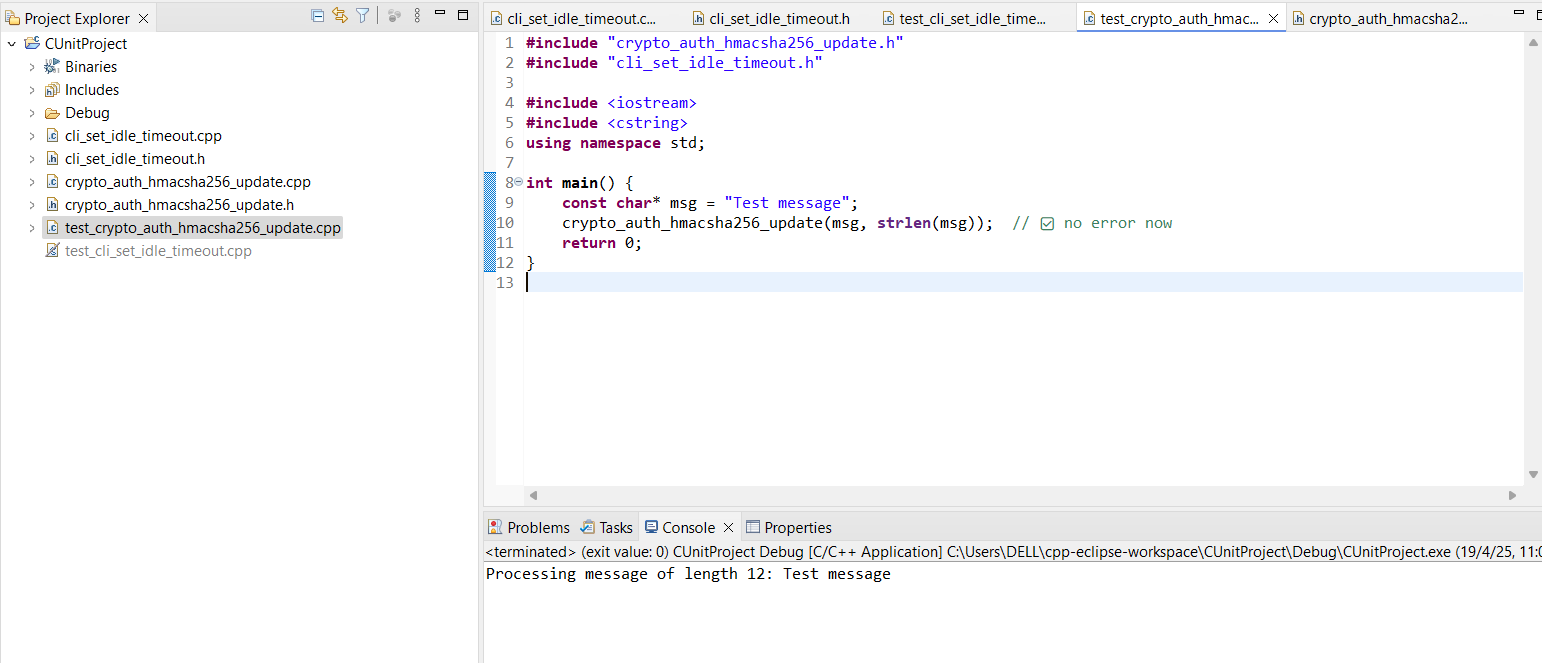
  
• Manual main-based test calls were created and verified for crypto\_auth\_hmacsha256\_update().



**✅ 3. Execution & Screenshot**

• CUnit test cases for cli\_set\_idle\_timeout executed successfully   


• Console test output for crypto\_auth\_hmacsha256\_update



**✅ 2. Input Space Partitioning For Both Functions**

**Function 1:** cli\_set\_idle\_timeout(int timeout)

**Input Parameter:**  
This function takes a single input:  
int timeout

**Input Characteristic Identified:**

* **Timeout Value Type** — the type of value being passed (negative, zero, positive)

| **Input Characteristic** | **Partition** | **Test Value** |  |
| --- | --- | --- | --- |
| Timeout Value | Negative | -5 |  |
|  | Zero | 0 |  |
|  | Positive | 10 |  |

* Any **negative value** should return -1 indicating invalid input.
* Any **zero or positive value** should return 0 indicating valid input.

--------------------------------------------------------------------------------------------

**Function 2:** crypto\_auth\_hmacsha256\_update(const char\* message, size\_t length)

**Two input parameters:**

1. message → a pointer to a C-style string
2. length → length of the message

**Identify Input Characteristics:**

Characteristic 1: Message Validity

* Is the pointer null?
* Is it an empty string?
* Is it a valid non-empty string?

Characteristic 2: Length Validity

* Is the length 0?
* Is the length matching the string size?
* Is it longer/shorter than actual string?

| **Input Characteristic** | **Partition** | **Test Value** |
| --- | --- | --- |
| Message | Null Pointer | nullptr |
|  | Empty String | "" |
|  | Valid String | "Test" |
| Length | Zero Length | 0 |
|  | Correct Length | 4 (for "Test") |
|  | Too Long (Edge Case) | 10 (for "Test") |
|  |  |  |

**✅ 3. Edge-Pair Coverage (EPC) + Control Flow Graph (CFG)**

**Function 1:** cli\_set\_idle\_timeout(int timeout)

**Code for CFG:**

int cli\_set\_idle\_timeout(int timeout) {

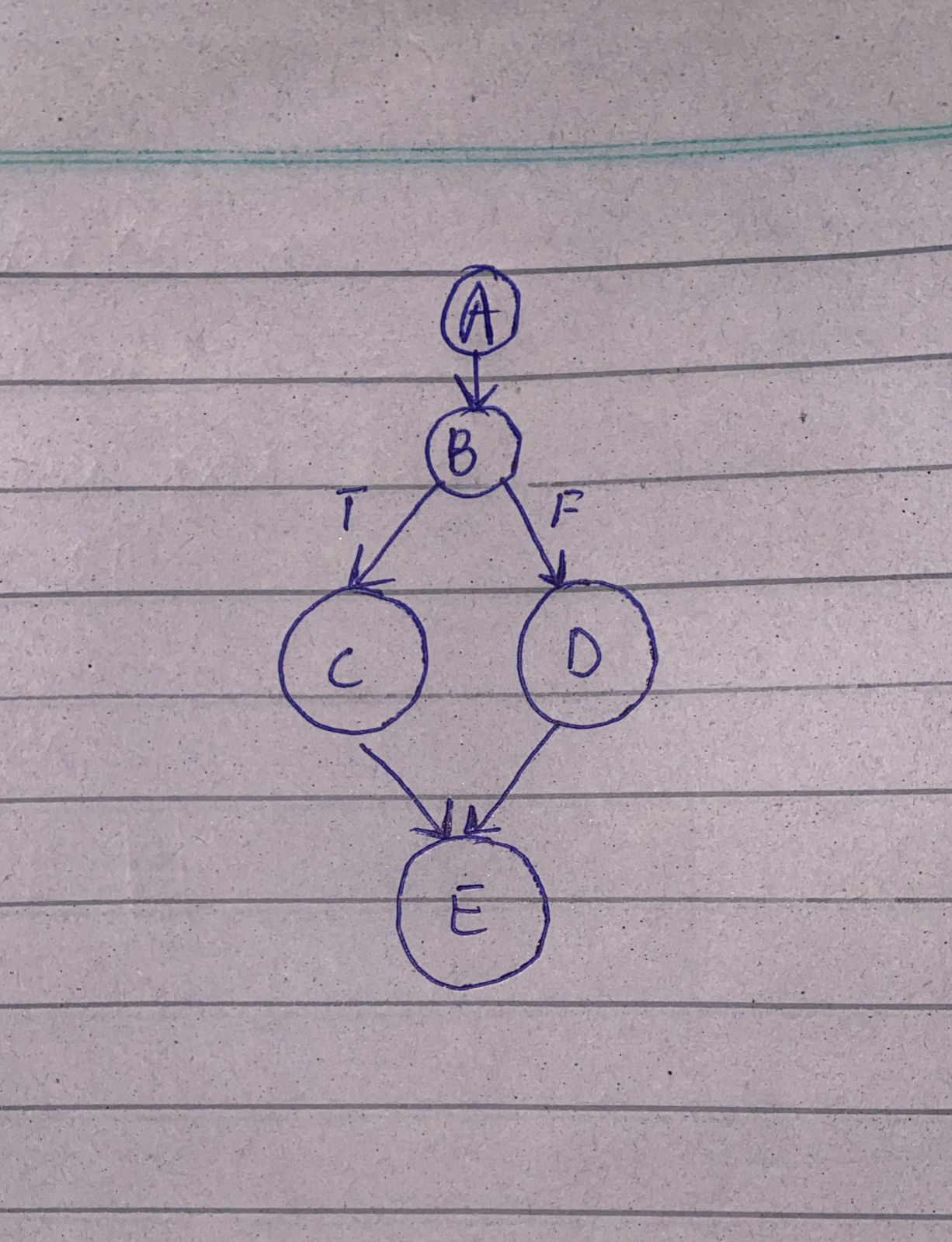
if (timeout < 0)

return -1;

return 0;

}

**Control Flow Graph (CFG):**



**Edge-Pairs:**

| **Edge-Pair** | **Path** |
| --- | --- |
| A → B → C | Negative timeout |
| A → B → D | Zero or positive |

**EPC Test Values:**

| **Edge-Pair** | **Test Value** | **Explanation** |
| --- | --- | --- |
| A → B → C → E | -5 | Negative → returns -1 |
| A → B → D → E | 10, 0 | Valid → returns 0 |

**Function 2:** crypto\_auth\_hmacsha256\_update(const char\* message, size\_t length)

**Code for CFG:**

int crypto\_auth\_hmacsha256\_update(const char\* message, size\_t length) {

if (message == nullptr || length == 0) {

return -1;

}

// Print message

for (size\_t i = 0; i < length; ++i) {

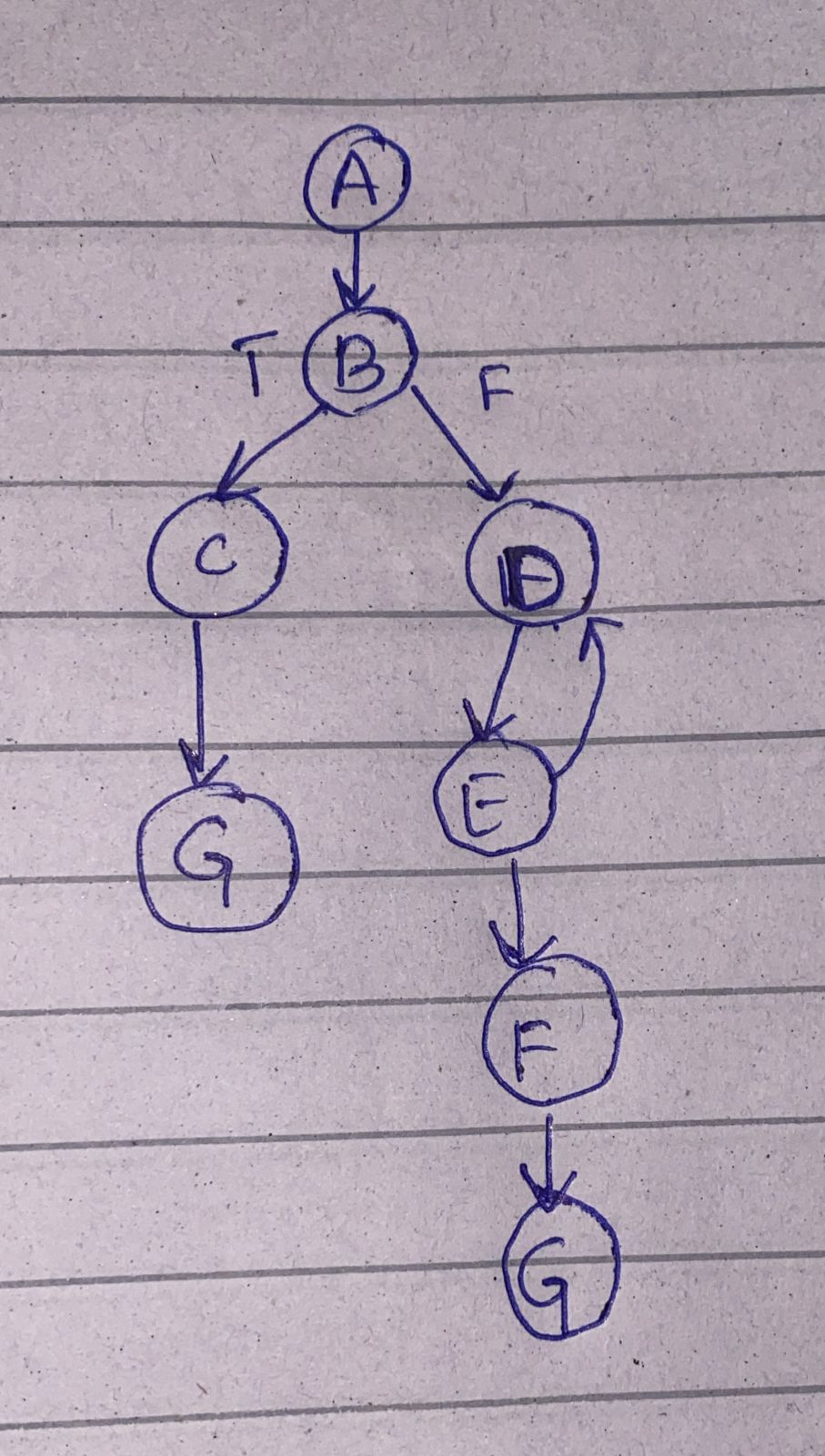
cout << message[i];

}

return 0;

}

**Control Flow Graph (CFG):**



**Edge Pairs:**

| **Edge-Pair** | **Description** |
| --- | --- |
| A → B → C | Message is null or length is 0 |
| A → B → D | Message valid, length > 0 |
| D → E → D | Loop executes (i < length) |
| D → F | Loop exit when i ≥ length |
| C → G | Return -1 on invalid input |
| F → G | Return 0 after printing |

**EPC Test Values:**

| **Edge Pair Path** | **Input Values** | **Expected Output** |
| --- | --- | --- |
| A → B → C → G | nullptr, 5 | -1 (invalid input) |
| A → B → C → G | "Hello", 0 | -1 (invalid input) |
| A → B → D → F → G | "Hi", 2 → loop runs twice | 0 + prints Hi |
| D → E → D (loop) | "Hi", 2 → tests loop condition | Intermediate printing |
| D → F → G | After i == length | Ends printing, return 0 |