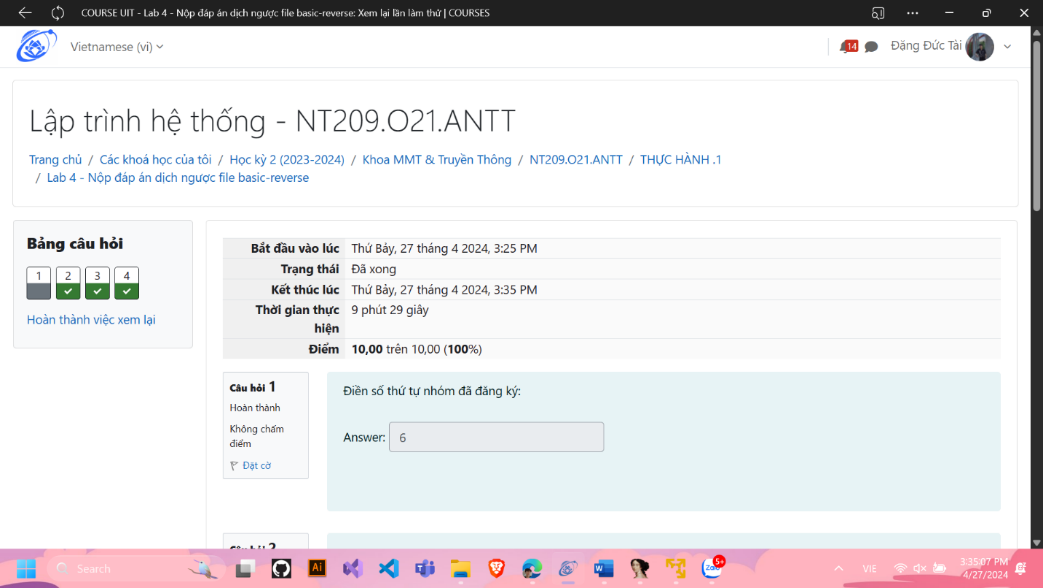
**LẬP TRÌNH HỆ THỐNG  
BÁO CÁO LAB 4**

**KỸ THUẬT DỊCH NGƯỢC CƠ BẢN**

|  |  |  |
| --- | --- | --- |
| **Họ và tên** | **MSSV** | **Lớp** |
| Lại Quan Thiên | 22521385 | NT209.O21.ANTT.1  Nhóm 6 |
| Đặng Đức Tài | 22521270 |



Hình 1: Kết quả tiến độ trả lời câu hỏi

# C2.1:

**\* Kết quả:**

A screenshot of a computer program

Description automatically generated

Hình 2: Kết quả khi chạy Câu 1

**\* Giải thích:**

Nếu đầu vào là *“Beauty is only skin deep*” thì gọi hàm ***success\_1().***

A computer code with green and blue text

Description automatically generated

Hình 3: Hàm hardCode của Câu 1

# C2.2:

**\* Kết quả:**

A screenshot of a computer program

Description automatically generated

Hình 4: Kết quả khi chạy Câu 2

**\* Giải thích:**

A screenshot of a computer code

Description automatically generated

Hình 5: Hàm otherhardCode của Câu 2

- Ở lệnh if đang so sánh nếu *v3 == 4* và *v0 == v2* thì gọi hàm success\_2() nên v3 và v0 là 2 số cần tìm.

- Chương trình gán ***v4 = 4*** nên số đầu tiên là **4**.

- Chương trình gán ***v0 = funny\_func(\*(&funny\_seq + 4), 4)***

- Tìm đến **funny\_seq** ta lấy kết quả của ô có địa chỉ là *(&funny\_seq + 4).*

A white background with blue text

Description automatically generated

Hình 6: Stack funny\_seq

- Tìm đến hàm **funny\_func()** ta thấy hàm ***funny\_func(\*(&funny\_seq + 4), 4)***trả về kết quả như sau: **(9 + 4 – 1) \* (9 + 4) = 156**

A white background with text and numbers

Description automatically generated

Hình 7: Hàm funny\_func

**=> Vậy hai số cần tìm là 4 và 156.**

# C2.3:

**\* Kết quả:**

A screenshot of a computer

Description automatically generated

Hình 8: Kết quả khi chạy Câi 3

\* **Giải thích:**

|  |  |
| --- | --- |
| A screenshot of a computer  Description automatically generated  Hình 9: Mã giả của hàm userpass() | Hình 10: Mã giả của hàm userpass() |

- Từ đoạn mã giả trên, ta có thể chuyển về code C/C++ để dễ hiểu hơn, cụ thể như sau:

|  |  |
| --- | --- |
| Hình 11: Code C/C++ được suy ra từ mã giả | Hình 12: Code C/C++ được suy ra từ mã giả |

A screenshot of a computer

Description automatically generated

Hình 13: Kết quả khi chạy chương trình C/C++ trên

- Phân tích:

**+ Hai hàm success\_3() và failed():**

* + Hàm ***success\_3()*** được gọi khi thông tin đăng nhập là chính xác.
  + Hàm ***failed()*** được gọi khi thông tin đăng nhập không chính xác.

**+ Hàm userpass():**

* + Khai báo một số biến cần thiết, bao gồm **v7** chứa một mảng các ký tự đã được khai báo sẵn (ở đây, các phần tử được khai báo trong mảng **v7** chính là mã ASCII), **s** chứa tên người dùng nhập vào (theo yêu cầu đề bài thì Nhóm 6 có Username là: **1270-1385**) và **v5** chứa mật khẩu nhập vào.
  + Sử dụng ***std::cin.getline()*** để lấy tên người dùng và mật khẩu từ người dùng.
  + Sau đó tiến hành kiểm tra xem độ dài của tên người dùng và mật khẩu có phù hợp không (9 ký tự).
  + Nếu phù hợp, tiếp tục xử lý:
    - Tạo một mảng **v4** và điền vào từng phần tử của nó dựa trên các ký tự trong tên người dùng (mảng **s**) hoặc trong mảng **v7**.
    - So sánh mỗi ký tự trong mật khẩu với giá trị tính toán từ **v4** và **s**. Nếu một trong các ký tự không khớp, thoát vòng lặp.
    - Nếu tất cả các ký tự khớp, trả về kết quả là thành công, gọi hàm ***success\_3()****.*
  + Nếu không phù hợp, trả về kết quả là thất bại, gọi hàm ***failed()****.*

**+ Hàm *main():*** Chỉ gọi hàm ***userpass()*** để bắt đầu quá trình xác thực.

**\* Thực hiện giải tay đoạn code trên để tìm ra mật khẩu:**

- Ta có:

+ Mảng char s[10] chứa username.

+ Mảng char v5[10] chứa password.

+ Mảng char v7[5] chứa 5 ký tự đã được khai báo sẵn, mỗi phần tử là mã ASCII.

+ Mảng char v4[9] dùng để điền vào từng phần tử của nó dựa trên các ký tự trong mảng s[10] và mảng v7[5] thông qua việc so sánh i

- Mảng **s[10]** **username** chứa 9 ký tự: **1270-1385** (ta tạm không tính ký tự \n), ta phân mảng này thành các ký tự tương ứng với mã ASCII như sau:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *i* | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| *s[i]* | **‘1’** | **‘2’** | **‘7’** | **‘0’** | **‘-’** | **‘1’** | **‘3’** | **‘8’** | **‘5’** |
| *ASCII* | **49** | **50** | **55** | **48** | **45** | **1** | **51** | **56** | **53** |

- Xét vòng lặp for đầu tiên: **for unsigned i = 0; i <= 8; ++i**, ta được giá trị của các phần tử mảng **v4** như sau (chú thích: hàng 3 là giá trị cụ thể của mỗi phần tử v4[i] bao gồm ký tự hiển thị và mã ASCII tương ứng)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *i* | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| *v4[i]* | **s[2]** | **s[3]** | **s[7]** | **s[8]** | **v7[0]** | **v7[1]** | **v7[2]** | **v7[3]** | **v7[4]** |
| *Giá trị v4[i]* | **‘7’ 55** | **‘0’ 48** | **‘8’ 56** | **‘5’ 53** | **‘{’ 123** | **‘<’ 60** | **‘i’ 105** | **‘s’ 115** | **‘i’ 105** |

- Xét vòng lặp for tiếp theo: **for unsigned i = 0; i <= strlen(s); ++i**, để không thoát khỏi vòng lặp cho đến khi nó lặp đủ thì phải thoả điều kiện **(s[i] + v4[i])/2 == v5[i],** do đó, ta có thể tìm được giá trị của các phần tử mảng **v5** như sau (chú thích: hàng 3 là giá trị của mỗi phần tử v5[i] với mã ASCII tương ứng)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *i* | 0 | 1 | 2 | 3 |
| *v5[i]* | **(s[0] + v4[0])/2** | **(s[1] + v4[1])/2** | **(s[2] + v4[2])/2** | **(s[3] + v4[3])/2** |
| *Giá trị v5[i]* | **(49 + 55)/2 = 52** | **(50 + 48)/2 = 49** | **(55 + 56)/2 = 55** | **(48 + 53)/2 = 50** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *i* | 4 | 5 | 6 | 7 |
| *v5[i]* | **(s[4] + v4[4])/2** | **(s[5] + v4[5])/2** | **(s[6] + v4[6])/2** | **(s[7] + v4[7])/2** |
| *Giá trị v5[i]* | **(45 + 123)/2 = 84** | **(49 + 60)/2 = 54** | **(51 + 105)/2 = 78** | **(56 + 115)/2 = 85** |

|  |  |
| --- | --- |
| *i* | 8 |
| *v5[i]* | **(s[8] + v4[8])/2** |
| *Giá trị v5[i]* | **(53 + 105)/2 = 79** |

- Từ kết quả 3 bảng trên, ta tổng hợp được mảng **v5** như sau:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *i* | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| *V5[i]* | **‘4’** | **‘1’** | **‘7’** | **‘2’** | **‘T’** | **‘6’** | **‘N’** | **‘U’** | **‘O’** |
| *ASCII* | **52** | **49** | **55** | **50** | **84** | **54** | **78** | **85** | **79** |

**=> Vậy password cần tìm là 4172T6NUO**