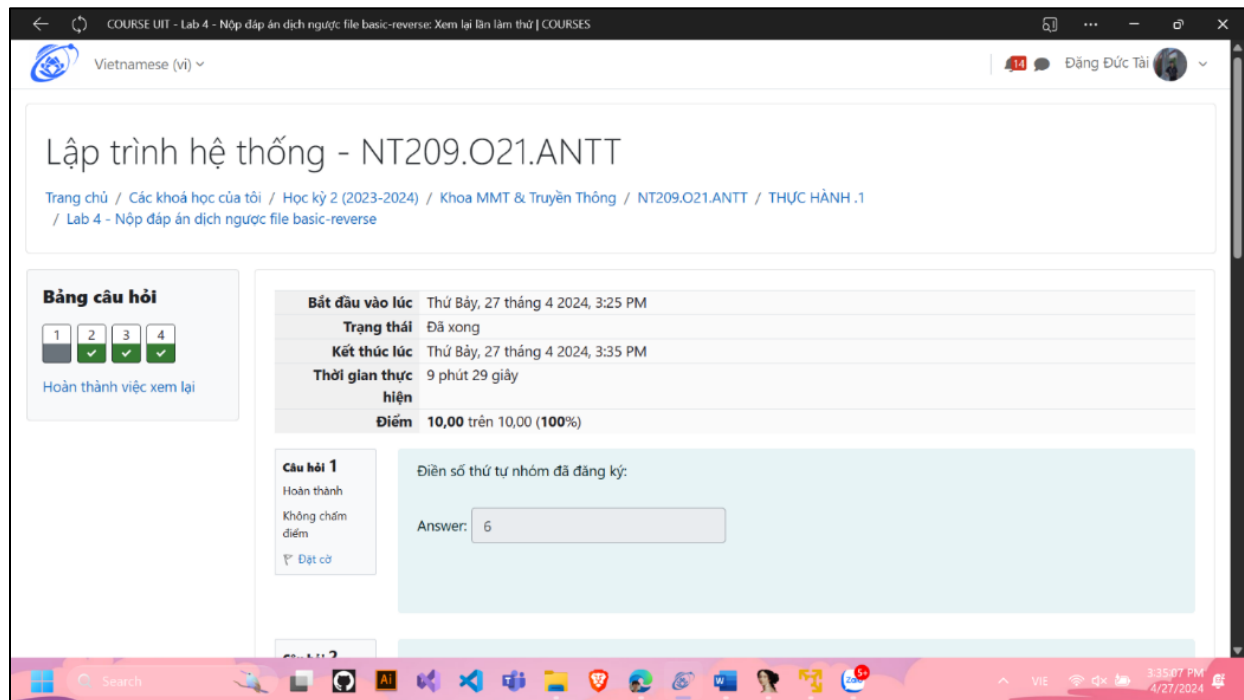


LẬP TRÌNH HỆ THỐNG

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:

```
dangductai@dangductai: ~/NT209
dangductai@dangductai:~/NT209$ ./basic-reverse
Supported authentication methods:
1. Hard-coded password
2. A pair of 2 numbers
3. Username/password
Enter your choice: 1
Enter the hard-coded password (option 1):
Beauty is only skin deep
Your input hard-coded password: Beauty is only skin deep
Congrats! You found the hard-coded secret, good job :).
dangductai@dangductai:~/NT209$
```

Hình 2: Kết quả khi chạy Câu 1

* Giải thích:

Nếu đầu vào là “Beauty is only skin deep” thì gọi hàm *success_1()*.

```
int hardCode()
{
    int result; // eax@2
    char s1; // [sp+8h] [bp-3F0h]@1

    getchar();
    puts("Enter the hard-coded password (option 1):");
    __isoc99_scanf("%[^\n]", &s1);
    printf("Your input hard-coded password: %s\n", &s1);
    if ( !strcmp(&s1, "Beauty is only skin deep") )
        result = success_1();
    else
        result = failed();
    return result;
}
```

Hình 3: Hàm hardCode của Câu 1

C2.2:

* Kết quả:

```
dangductai@dangductai: ~/NT209
dangductai@dangductai:~/NT209$ ./basic-reverse
Supported authentication methods:
1. Hard-coded password
2. A pair of 2 numbers
3. Username/password
Enter your choice: 2
Enter your 2 numbers (separated by space) (option 2):
4 156
Your input: 4 156
Congrats! You found a secret pair of numbers :).
dangductai@dangductai:~/NT209$
```

Hình 4: Kết quả khi chạy Câu 2

* Giải thích:

```
int otherhardCode()
{
    int v0; // edx@2
    int result; // eax@3
    int v2; // [sp+4h] [bp-14h]@1
    int v3; // [sp+8h] [bp-10h]@1
    int v4; // [sp+Ch] [bp-Ch]@1

    getchar();
    puts("Enter your 2 numbers (separated by space) (option 2):");
    __isoc99_scanf("%d %d", &v3, &v2);
    printf("Your input: %d %d\n", v3, v2);
    v4 = 4;
    if ( v3 == 4 )
    {
        v0 = funny_func((&funny_seq + 4), 4);
        if ( v0 == v2 )
            result = success_2();
        else
            result = failed();
    }
    else
    {
        result = failed();
    }
    return result;
}
```

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 $v0 = \text{funny_func}(*(\&\text{funny_seq} + 4), 4)$
- Tìm đến `funny_seq` ta lấy kết quả của ô có địa chỉ là $(\&\text{funny_seq} + 4)$.

.rodata:08048A60	public funny_seq	
.rodata:08048A60 funny_seq	dd 1	; DATA XREF: otherhardCode+60↑r
.rodata:08048A64	dd 3	
.rodata:08048A68	dd 5	
.rodata:08048A6C	dd 7	
.rodata:08048A70	dd 9	

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: $(9 + 4 - 1) * (9 + 4) = 156$

```
int __cdecl funny_func(int a1, int a2)
{
    return (a1 + a2 - 1) * (a1 + a2);
}
```

Hình 7: Hàm *funny_func*

=> Vậy hai số cần tìm là 4 và 156.

C2.3:

* Kết quả:

```
dangductai@dangductai: ~/NT209
dangductai@dangductai:~/NT209$ ./basic-reverse
Supported authentication methods:
1. Hard-coded password
2. A pair of 2 numbers
3. Username/password
Enter your choice: 3
Enter your username:
1270-1385
Enter your password:
4172T6NUO
Your input username: 1270-1385 and password: 4172T6NUO
Congrats! You found your own username/password pair :).
dangductai@dangductai:~/NT209$
```

Hình 8: Kết quả khi chạy Cài 3

* Giải thích:

```
1 int userpass()
2 {
3     size_t v0; // ebx@2
4     int result; // eax@3
5     size_t v2; // eax@15
6     size_t v3; // edx@16
7     char v4[9]; // [sp+Ah] [bp-2Eh]@6
8     char v5[10]; // [sp+13h] [bp-25h]@1
9     char s[10]; // [sp+1Dh] [bp-18h]@1
10    char v7[5]; // [sp+27h] [bp-11h]@1
11    unsigned int i; // [sp+2Ch] [bp-Ch]@4
12
13    v7[0] = 123;
14    v7[1] = 60;
15    v7[2] = 105;
16    v7[3] = 115;
17    v7[4] = 105;
18    getchar();
19    puts("Enter your username:");
20    __isoc99_scanf("%[^\n]", s);
21    getchar();
22    puts("Enter your password:");
23    __isoc99_scanf("%[^\n]", v5);
24    printf("Your input username: %s and password: %s\n", s, v5);
25    if ( strlen(s) == 9 && (v0 = strlen(s), v0 == strlen(v5)) )
26    {
27        for ( i = 0; (signed int)i <= 8; ++i )
28        {
29            if ( (signed int)i > 1 )
30            {
31                if ( (signed int)i > 3 )
32                {
33                    v4[i] = v7[i - 4];
34                }
35                else
36                {
37                    v4[i] = s[i + 5];
38                }
39            }
40        }
41    }
42    v2 = strlen(s);
43    if ( v2 <= 1 || (s[i] + v4[i]) / 2 != v5[i] )
44        break;
45    v3 = strlen(s);
46    if ( v3 == i )
47        result = success_3();
48    else
49        result = failed();
50    return result;
51 }
```

Hình 9: Mã giả của hàm userpass()

```
19 puts("Enter your username:");
20 __isoc99_scanf("%[^\n]", s);
21 getchar();
22 puts("Enter your password:");
23 __isoc99_scanf("%[^\n]", v5);
24 printf("Your input username: %s and password: %s\n", s, v5);
25 if ( strlen(s) == 9 && (v0 = strlen(s), v0 == strlen(v5)) )
26 {
27     for ( i = 0; (signed int)i <= 8; ++i )
28     {
29         if ( (signed int)i > 1 )
30         {
31             if ( (signed int)i > 3 )
32             {
33                 v4[i] = v7[i - 4];
34             }
35             else
36             {
37                 v4[i] = s[i + 5];
38             }
39         }
40     }
41     for ( i = 0; ; ++i )
42     {
43         v2 = strlen(s);
44         if ( v2 <= 1 || (s[i] + v4[i]) / 2 != v5[i] )
45             break;
46     }
47     v3 = strlen(s);
48     if ( v3 == i )
49         result = success_3();
50     else
51         result = failed();
52 }
53 else
54 {
55     result = failed();
56 }
57 return result;
58 }
```

Hình 10: Mã giả của hàm userpass()

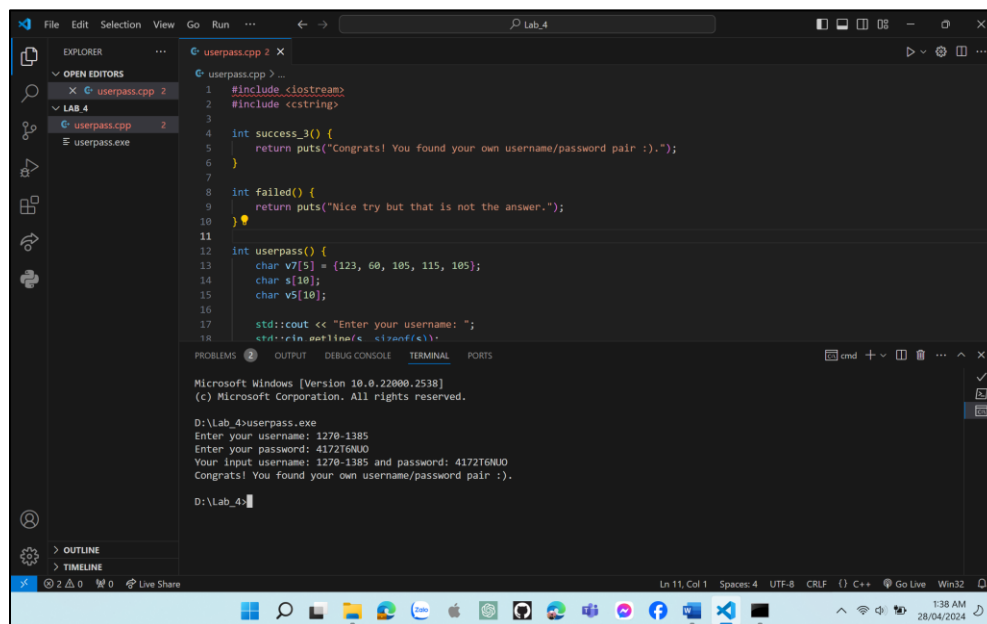
- Từ đoạn mã giả trên, ta có thể chuyển về code C/C++ để dễ hiểu hơn, cụ thể như sau:

```
1  #include <iostream>
2  #include <cstring>
3
4  int success_3()
5  {
6      return puts("Congrats! You found your own username/password pair :.");
7  }
8
9  int failed()
10 {
11     return puts("Nice try but that is not the answer.");
12 }
13
14 int userpass()
15 {
16     char v7[5] = {123, 60, 105, 115, 105};
17     char s[10];
18     char v5[10];
19
20     std::cout << "Enter your username: ";
21     std::cin.getline(s, sizeof(s));
22
23     std::cout << "Enter your password: ";
24     std::cin.getline(v5, sizeof(v5));
25
26     std::cout << "Your input username: " << s
27               << " and password: " << v5 << std::endl;
```

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

```
28
29     if (strlen(s) == 9 && strlen(s) == strlen(v5))
30     {
31         char v4[9];
32         for (unsigned int i = 0; i <= 8; ++i)
33         {
34             if (i > 1)
35             {
36                 if (i > 3)
37                     v4[i] = v7[i - 4];
38                 else
39                     v4[i] = s[i + 5];
40             }
41             else
42             {
43                 v4[i] = s[i + 2];
44             }
45         }
46         unsigned int i;
47         for (i = 0; i < strlen(s); ++i)
48         {
49             if ((s[i] + v4[i]) / 2 != v5[i])
50                 break;
51         }
52         if (strlen(s) == 1)
53             return success_3();
54         else
55             return failed();
56     }
57     else
58     {
59         return failed();
60     }
61 }
62
63 int main()
64 {
65     userpass();
66     return 0;
67 }
68
```

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

The image shows a screenshot of a C++ IDE, likely Visual Studio Code, with a dark theme. The Explorer panel on the left shows the project structure with files 'userpass.cpp' and 'userpass.exe'. The main editor window displays the source code of 'userpass.cpp', which is identical to the code shown in Figure 11. Below the editor, the 'TERMINAL' panel shows the output of running the program. The output text is: 'D:\Lab_4>userpass.exe', 'Enter your username: 1270-1385', 'Enter your password: 417216NU0', 'Your input username: 1270-1385 and password: 417216NU0', and 'Congrats! You found your own username/password pair :.'. The status bar at the bottom indicates the file is 'Ln 11, Col 1' and the encoding is 'UTF-8'.

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>i</i>	0	1	2	3	4	5	6	7	8
<i>s[i]</i>	‘1’	‘2’	‘7’	‘0’	‘-’	‘1’	‘3’	‘8’	‘5’
<i>ASCII</i>	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>i</i>	0	1	2	3	4	5	6	7	8
<i>v4[i]</i>	s[2]	s[3]	s[7]	s[8]	v7[0]	v7[1]	v7[2]	v7[3]	v7[4]
<i>Giá trị v4[i]</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 thỏa 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>i</i>	0	1	2	3
<i>v5[i]</i>	(s[0] + v4[0])/2	(s[1] + v4[1])/2	(s[2] + v4[2])/2	(s[3] + v4[3])/2
<i>Giá trị v5[i]</i>	(49 + 55)/2 = 52	(50 + 48)/2 = 49	(55 + 56)/2 = 55	(48 + 53)/2 = 50

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

<i>i</i>	8
<i>v5[i]</i>	(s[8] + v4[8])/2
<i>Giá trị v5[i]</i>	(53 + 105)/2 = 79

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

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

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