

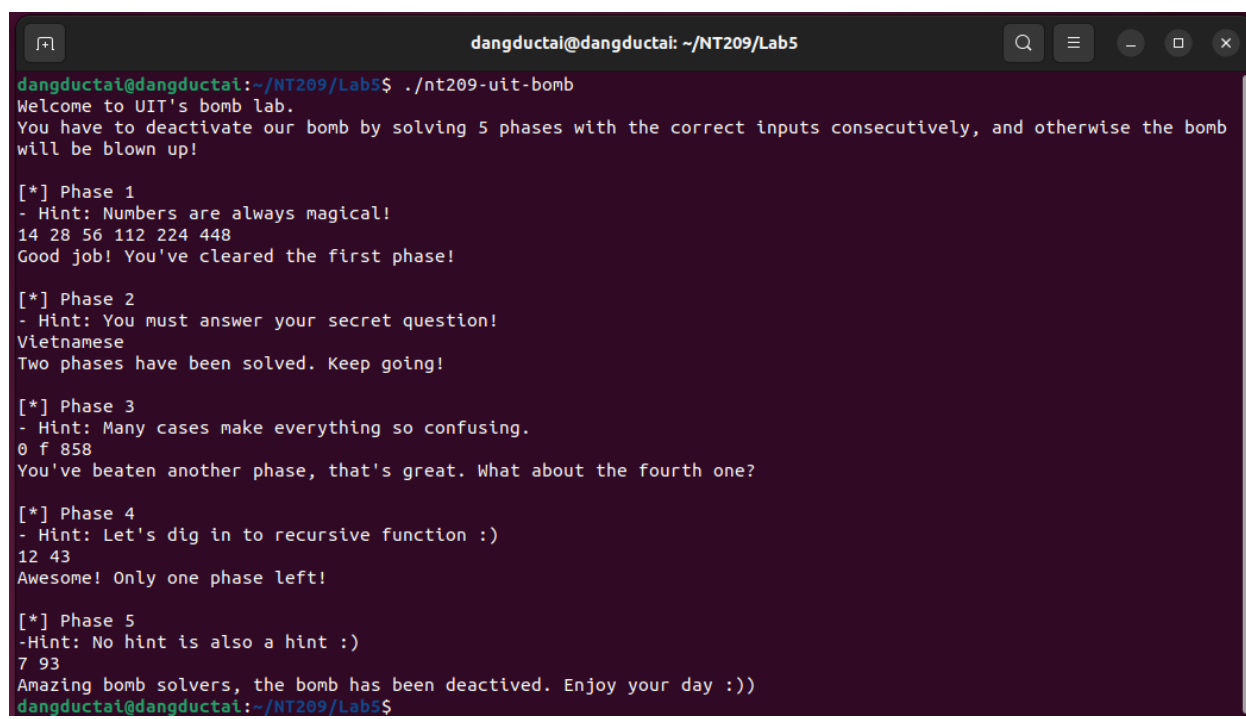
LẬP TRÌNH HỆ THỐNG

BÁO CÁO LAB 5

KỸ THUẬT DỊCH NGƯỢC (TIẾP THEO)

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

Minh chứng hoàn thành đúng 5 phases:



```
dangductai@dangductai: ~/NT209/Lab5
dangductai@dangductai:~/NT209/Lab5$ ./nt209-uit-bomb
Welcome to UIT's bomb lab.
You have to deactivate our bomb by solving 5 phases with the correct inputs consecutively, and otherwise the bomb
will be blown up!

[*] Phase 1
- Hint: Numbers are always magical!
14 28 56 112 224 448
Good job! You've cleared the first phase!

[*] Phase 2
- Hint: You must answer your secret question!
Vietnamese
Two phases have been solved. Keep going!

[*] Phase 3
- Hint: Many cases make everything so confusing.
0 f 858
You've beaten another phase, that's great. What about the fourth one?

[*] Phase 4
- Hint: Let's dig in to recursive function :)
12 43
Awesome! Only one phase left!

[*] Phase 5
-Hint: No hint is also a hint :)
7 93
Amazing bomb solvers, the bomb has been deactivated. Enjoy your day :))
dangductai@dangductai:~/NT209/Lab5$
```

Hình 1: Minh chứng đáp án 5 phases

Phase 1:

```
u4 = __isoc99_sscanf(a1, "%d %d %d %d %d %d", u2, &u2[1], &u2[2], &u2[3], &u2[4], &u2[5]);
if ( u4 != 6 )
    explode_bomb();
u3 = 14;
result = u2[0];
if ( u2[0] != 14 )
    explode_bomb();
for ( i = 1; i <= 5; ++i )
{
    result = u2[i];
    if ( result != 2 * u2[i - 1] )
        explode_bomb();
}
```

Hình 2: Mã giả của phase 1

Giải thích:

- Từ 2 dòng đầu, ta suy ra được rằng, input phải là một dãy gồm 6 số.
- Dòng *if (u2[0] != 14)* cho ta biết số đầu tiên là 14.
- Trong vòng for, ta suy ra được rằng $u2[i] = 2 * u2[i - 1]$. Tức là số sau bằng 2 lần số trước. Biết số đầu tiên là 14, ta suy ra được kết quả cần tìm là:

14 28 56 112 224 448

Phase 2:

```
u1 = QUESTIONS[6];
s2 = ANSWERS[*(&QA_MAP + 6)];
s1 = (char *)transfer(a1);
if ( !*s2 || (LOBYTE(u2) == is_equal(s1, s2), !u2) )
    explode_bomb();
return u2;
```

Hình 3: Mã giả của phase 2

Giải thích:

- Việc ta cần làm ở phase 2 là trả lời câu hỏi thứ 6 trong mảng QUESTIONS.

```

.data:0804B060
.data:0804B060 QUESTIONS
.data:0804B064
.data:0804B068
.data:0804B06C
.data:0804B070
.data:0804B074
.data:0804B078
.data:0804B07C

public QUESTIONS
dd offset aMyVehicleRegis ; DATA XREF: phase2+10↑r
; "My vehicle registration plate starts wi"...
dd offset aWhatIsTheCapit ; "What is the capital of Thailand?"
dd offset aWhatIsYourMajo ; "What is your major in English? (Capital"...
dd offset aWhichSeasonHas ; "Which season has cherry blossoms?"
dd offset aThanksToMeYouC ; "Thanks to me, you can see straight thro"...
dd offset aWhichCountryIs ; "Which country is the Lion city in South"...
dd offset aWhatIsTheMainL ; "What is the main language used in this "...
dd offset aEnterTheCurren ; "Enter the current date using the format"...

```

Hình 4: Mảng QUESTIONS

- Câu hỏi chúng ta phải trả lời là: *What is the main language used in this?*
- Chúng ta có thể dễ dàng đưa ra câu trả lời là **Vietnamese** nhưng giờ hãy thử tìm ra câu trả lời bằng cách dịch ngược theo đúng yêu cầu bài lab.
- Câu trả lời sẽ là phần tử thứ 6 trong mảng ANSWERS.

```

.data:0804B160
.data:0804B160 ANSWERS
.data:0804B160
.data:0804B164
.data:0804B168
.data:0804B16C
.data:0804B170
.data:0804B174
.data:0804B178
.data:0804B17C
.data:0804B180

public ANSWERS
dd offset aFmr1Hysrk ; DATA XREF: phase2+2A↑r
; "Fmr1 Hysrk"
dd offset aFerkoso ; "Ferkoso"
dd offset aMrjsuqexmsrWig ; "Mrjsuqexmsr Wiggyvmxc"
dd offset aWtvmrk ; "Wtvmrk"
dd offset aAmrhsa ; "Amrhsa"
dd offset aWmrketsui ; "Wmrketsui"
dd offset aZmixreqiwi ; "Zmixreqiwi"
dd offset a496468 ; "49/6468"
dd offset aFmr1Hysrk ; "Fmr1 Hysrk"

```

Hình 5: Mảng ANSWERS

- Câu trả lời đã bị thay đổi bởi hàm **transfer** là **Zmixreqiwi** nhưng ta cần dịch ngược để có câu trả lời đúng.
- Dựa theo hàm **transfer**, viết một hàm **revert** để tìm ra câu trả lời ban đầu:

```

int __cdecl transfer(int a1)
{
    char u2; // [sp+0h] [bp-6h]@8
    char u3; // [sp+Bh] [bp-5h]@2
    int i; // [sp+Ch] [bp-4h]@1

    for ( i = 0; !(_BYTE)(i + a1); ++i )
    {
        u3 = !(_BYTE)(i + a1);
        if ( (u3 <= 96 || u3 > 122) && (u3 <= 64 || u3 > 90) )
        {
            if ( u3 > 47 && u3 <= 57 )
                u3 = (u3 - 48 + 4) % 10 + 48;
        }
        else
        {
            if ( u3 <= 96 || u3 > 122 )
                u2 = 65;
            else
                u2 = 97;
            u3 = (u3 - u2 + 4) % 26 + u2;
        }
        !(_BYTE)(a1 + i) = u3;
    }
    return a1;
}

```

Hình 6: Hàm transfer

```

// Function to revert transformed string back to original
char* revert(const char* transformed) {
    char* result = new char[strlen(transformed) + 1];
    strcpy(result, transformed);

    for (int i = 0; result[i]; ++i) {
        char v3 = result[i];
        if ((v3 <= 96 || v3 > 122) && (v3 <= 64 || v3 > 90)) {
            if (v3 > 47 && v3 <= 57)
                v3 = (v3 - 48 - 4 + 10) % 10 + 48;
            } else {
                char v2 = (v3 <= 96 || v3 > 122) ? 65 : 97;
                v3 = (v3 - v2 - 4 + 26) % 26 + v2;
            }
            result[i] = v3;
        }
    }
    return result;
}

// Main function
int main() {
    char* original_str = revert("Zmixreqiwi");
    cout << original_str << endl;
    return 0;
}

```

Hình 7: Hàm revert

- Sau khi revert, ta được kết quả là *Vietnamese*:

```
PS C:\MyData\UIT\Semester_4\Lập trình Hệ thống\Lab_4_ltht> ./phase_2.exe
Vietnamese
```

Hình 8: Kết quả phase 2 sau khi revert

Phase 3:

```
int result; // eax@21
unsigned __int8 v2; // [sp+Fh] [bp-19h]@1
int v3; // [sp+10h] [bp-18h]@1
int v4; // [sp+14h] [bp-14h]@1
int v5; // [sp+18h] [bp-10h]@1
char v6; // [sp+1Fh] [bp-9h]@4

v5 = 0;
v5 = __isoc99_sscanf(a1, "%d%c%d", &v4, &v2, &v3);
if ( v5 <= 2 )
    explode_bomb();
switch ( v4 )
{
    case 0:
        v6 = 102;
        if ( v3 != 858 )
            explode_bomb();
        return result;
    case 1:
        v6 = 121;
        if ( v3 != 57 )
            explode_bomb();
        return result;
    case 2:
        v6 = 107;
        if ( v3 != 913 )
            explode_bomb();
        return result;
    case 3:
        v6 = 117;
        if ( v3 != 973 )
            explode_bomb();
        return result;
    case 4:
        v6 = 120;
        if ( v3 != 465 )
            explode_bomb();
        return result;
    case 5:
        v6 = 110;
        if ( v3 != 797 )
            explode_bomb();
        return result;
    case 6:
        v6 = 99;
        if ( v3 != 186 )
            explode_bomb();
        return result;
    case 7:
        v6 = 97;
        if ( v3 != 598 )
            explode_bomb();
        return result;
    default:
        v6 = 110;
        explode_bomb();
        return result;
}
result = v2;
if ( v6 != v2 )
    explode_bomb();
return result;
```

Hình 9: Mã giả của phase 3

Giải thích:

- Có 8 cases nên sẽ có 8 đáp án khác nhau.
- Nhìn hai dòng v5 ta dễ dàng nhận ra rằng input là 2 số (%d) và 1 kí tự (%c).
- Số thứ nhất (v4) sẽ là số của từng case. Ví dụ: case 0 thì v4 = 0, case 1 thì v4 = 1...

- Chú ý dòng ***if (v6 != v2)***, nếu v6 khác v2 thì bom sẽ nổ, vậy nên v6 phải = v2. Đầu vào của v2 là %c, tức là 1 ký tự nên v6 là mã ASCII của v2. Dựa vào bảng mã ASCII dễ dàng tra ra được v2.
- Giả sử ở case 0, ***if (v3 != 858)***, nếu v3 khác 858 thì bom nổ nên v3 phải bằng 858. Vậy ở mỗi case, v3 phải bằng số đang được so sánh với nó trong hàm if.
- Từ các điều kiện trên, ta suy ra kết quả của các case lần lượt là:

Case 0: ***0 f 858***

Case 2: ***2 k 913***

Case 4: ***4 x 465***

Case 6: ***6 c 186***

Case 1: ***1 y 57***

Case 3: ***3 u 973***

Case 5: ***5 n 797***

Case 7: ***7 a 598***

Phase 4:

```

v6 = __isoc99_sscanf(a1, "%d %d", &v3, &v2);
if ( v6 != 2 || v3 < 0 || v3 > 14 )
    explode_bomb();
v5 = 43;
v4 = func4(v3, 0, 14);
if ( v4 != v5 || (result = v2, v2 != v5) )
    explode_bomb();
return result;

```

Hình 10: Mã giả của phase 4

Giải thích:

- Từ dòng đầu tiên, suy ra được input là hai số v3 và v2.
- Dòng thứ hai, nếu v3 < 0 hoặc v3 > 14 bom sẽ nổ nên 0 < v3 < 14.
- Từ dòng if, nếu v4 khác v5 và v2 khác v5 bom sẽ nổ nên v4 phải bằng v5 và v2 phải bằng v5.
- Ta có v5 = 43, suy ra v2 = 43 và v4 = 43.
- Ta thấy: ***v4 = func4(v3, 0, 14)***. Ta đã tìm được v4 = 43. Hàm func4 ta truyền ba tham số là v3, 0, 14. Vậy ***43 = func4(v3, 0, 14)***. Tìm đến func4, viết chương trình bằng C++ để brute force v3 từ hàm func4:

```

int __cdecl func4(int a1, int a2, int a3)
{
    int result; // eax@2
    int v4; // [sp+Ch] [bp-Ch]@1

    v4 = (a3 - a2) / 2 + a2;
    if ( v4 <= a1 )
    {
        if ( v4 >= a1 )
            result = (a3 - a2) / 2 + a2;
        else
            result = func4(a1, v4 + 1, a3) + v4;
    }
    else
    {
        result = func4(a1, a2, v4 - 1) + v4;
    }
    return result;
}

```

Hình 11: Hàm func4

The screenshot shows a C++ IDE with the following code in `phase_4.cpp`:

```

3  int func4(int a1, int a2, int a3) {
4      int result;
5
6      int v4 = (a3 - a2) / 2 + a2;
7      if (v4 <= a1) {
8          if (v4 >= a1)
9              result = (a3 - a2) / 2 + a2;
10         else
11             result = func4(a1, v4 + 1, a3) + v4;
12     } else {
13         result = func4(a1, a2, v4 - 1) + v4;
14     }
15     return result;
16 }
17
18 int main() {
19     const int a2 = 0;
20     const int a3 = 14;
21     const int desired_result = 43;
22
23     for (int a1 = 0; a1 <= 100000000; ++a1) {
24         int result = func4(a1, a2, a3);
25         if (result == desired_result) {
26             std::cout << "Found v3: " << a1 << std::endl;
27             break;
28         }
29     }
}

```

The terminal output shows: `Found v3: 12`.

Hình 12: Brute force phase 4

- Vậy kết quả cần tìm là: **12 43**

Phase 5:

```

v5 = __isoc99_sscanf(a1, "%d %d", &v3, &v2);
if ( v5 <= 1 )
    explode_bomb();
v3 &= 0xFu;
v4 = v3;
v7 = 0;
v6 = 0;
while ( v3 != 15 )
{
    ++v7;
    v3 = array_3855[v3];
    v6 += v3;
}
if ( v7 != 12 || (result = v2, v6 != v2) )
    explode_bomb();

```

Hình 13: Mã giả của phase 5

Giải thích:

- Từ hai dòng đầu tiên, suy ra input là 2 số.
- Trong vòng lặp while, v7 đóng vai trò là biến index. Ở dòng if, ta suy ra được khi kết thúc vòng lặp $v7 = 12$, tức là while đã lặp 12 lần và $v6 = v2$.

- Tìm đến mảng `array_3855` ta thấy:

```
.data:0804B200 array_3855      dd 0Ah
.data:0804B204              dd 2
.data:0804B208              dd 0Eh
.data:0804B20C              dd 7
.data:0804B210              dd 8
.data:0804B214              dd 0Ch
.data:0804B218              dd 0Fh
.data:0804B21C              dd 0Bh
.data:0804B220              dd 0
.data:0804B224              dd 4
.data:0804B228              dd 1
.data:0804B22C              dd 0Dh
.data:0804B230              dd 3
.data:0804B234              dd 9
.data:0804B238              dd 6
.data:0804B23C              dd 5
.data:0804B23C _data        ends
```

Hình 14: Mảng `array_3855`

- Các giá trị có đuôi “*h*” là giá trị trong hệ HEX. Chuyển toàn bộ về DEC ta có:

`int array_3855[] = {10, 2, 14, 7, 8, 12, 15, 11, 0, 4, 1, 13, 3, 9, 6, 5};`

- Vòng lặp `while` chỉ chạy khi `v3 != 15` và kết thúc khi `v7 = 12`. Do `v3 &= 0xFu` nên `v3` sẽ có giá trị từ 0 đến 15. Giá trị ta cần tìm sẽ là `v3` ban đầu và `v6` sau khi kết thúc vòng lặp. Viết chương trình bằng C để brute force tìm ra `v3` và `v6`:

```
1 #include <stdio.h>
2
3 int main() {
4     int array_3855[] = {10, 2, 14, 7, 8, 12, 15, 11, 0, 4, 1, 13, 3, 9, 6, 5};
5     int v3_initial, v3, v6, v7;
6
7     // Thử tất cả các giá trị có thể của v3 từ 0 đến 14
8     for (v3_initial = 0; v3_initial < 15; v3_initial++) {
9         v3 = v3_initial;
10        v6 = 0;
11        v7 = 0;
12
13        // Vòng lặp dựa trên mô tả
14        while (v3 != 15) {
15            ++v7;
16            v3 = array_3855[v3];
17            v6 += v3;
18        }
19
20        // Kiểm tra điều kiện v7 == 12
21        if (v7 == 12) {
22            printf("v3 ban dau: %d\n", v3_initial);
23            printf("v6 sau khi ket thuc vong lap: %d\n", v6);
24            break;
25        }
26    }
```

PROBLEMS OUTPUT DEBUG CONSOLE PORTS TERMINAL

Executing task: C:/windows/System32/cmd.exe /d /c .\build\Debug\outDebug.exe

v3 ban dau: 7
v6 sau khi ket thuc vong lap: 93

Hình 15: Brute force phase 5

- Vậy kết quả cần tìm là: **7 93**