**Phase 4:**

* Trong hàm **main**, ta sẽ thấy **phase4** như sau:

A screenshot of a computer program

Description automatically generated

* Nhấn vào phase4 ta được chương trình như sau:

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generatedA screen shot of a computer code

Description automatically generated

**\*Phân tích:**

- Dựa theo thông tin của đoạn code sau:

A screenshot of a computer program

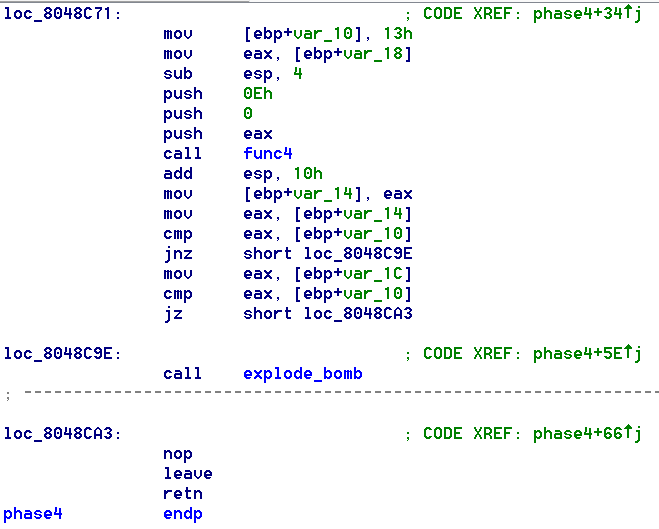
Description automatically generated

* Đoạn code trên là in ra **“%d %d”** và dòng so sánh **[epb+var\_C]** với 2, dựa theo code C ta biết rằng việc cần làm là nhập vào 2 số nguyên. Nếu không phải 2 số thì chương trình sẽ đi đến **short loc\_8048C6C** chính là **explode\_bomb**, khi đó sẽ thất bại.
* Dựa vào những dòng tiếp theo, ta sẽ phân tích được:

A screenshot of a computer code

Description automatically generated

* Số đầu tiên nhập vào phải lớn hơn 0, bởi 3 dòng **mov, test** và **js** ở trên, nếu không thì sẽ nhảy đến **short loc\_8048C6C** chính là **explode\_bomb**, khi đó sẽ thất bại. Với 3 dòng **mov,** **cmp** và **jle** ta thấy số đầu tiên phải là 1 số <=14, vì nếu không thỏa mãn sẽ không nhảy đến **short loc\_8048C71** mà sẽ chạy xuống dòng tiếp theo, chính là **short loc\_8048C6C** chính là **explode\_bomb**, khi đó sẽ thất bại.
* Nếu số nhập vào thỏa mãn sẽ nhảy đến **short loc\_8048C71**, ta sẽ phân tích code sau:

Đ

* Đầu tiên gán giá trị 19 vào ô nhớ (**epb + var\_10**) tạm gọi là **v5**.
* Gán giá trị ô nhớ (**epb + var\_18**) tạm gọi là **v3** vào thanh ghi eax.
* **Push** giá trị **14, 0** và giá trị **v3** vào **stack**.
* Gọi **func4**, khi đó **func4** sẽ có 3 giá trị vào lần lượt là **v3,** **0, 14** (theo quy tắc truy xuất biến cục bộ của stack)
* Kết quả của hàm **func4** được lưu tại thanh ghi **eax,** và được gán vào ô nhớ (**epb + var\_14**) tạm gọi và **v4**, so sánh **v4** và **v5**, nếu **v4** khác **v5** (**v4 != 19**) sẽ nhảy đến **short loc\_8048C9E** chính là **explode\_bomb**, khi đó sẽ thất bại. Đồng thời so sánh (**ebp + var\_1C**) chính là số thứ 2 khi nhập vào với **v5**, nếu **v2==v5** sẽ bỏ qua bomb và đi đến kết thúc. Ngược lại, sẽ đi đến **short loc\_8048C9E** chính là **explode\_bomb**, khi đó sẽ thất bại. từ đó suy ra số thứ 2 cần nhập vào chính là số **19(1).**
* Mở hàm func4, ta được những đoạn code sau:

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A computer code with blue and green text

Description automatically generated

A computer code with blue and green text

Description automatically generated

**\*Phân tích hàm func4:**

- Khi phân tích đoạn sau:

A blue lines on a white background

Description automatically generated

* Ta có thể suy ra được 3 đối số đưa vào hàm lần lượt gọi là **a1, a2, a3** được gán với ô nhớ (**epb+arg\_0), (epb+arg\_)** và **(epb+arg\_8)** biến **temp** là **(epb+var\_C).**
* Qua đoạn code sau có thể suy ra công thức thực thi:

A screenshot of a computer code

Description automatically generated

* Công thức toán tương ứng với đoạn code trên là:

**temp = (a3-a2)/2 + a2**

* So sánh **temp** với **a1:**



* Nếu **temp <= a1** sẽ thực thi hàm **short loc\_8048C08:**

A computer screen shot of a computer code

Description automatically generated

* Hàm trên so sánh **temp** với **a1**, nếu **temp >= a1** thì sẽ thực thi hàm **short loc\_8048C31**

A computer code with blue and green text

Description automatically generated

* Khi đó sẽ gán **temp** cho **eax** rồi đi đến kết thúc hàm, với **temp** được tính với công thức toán ở trên: **temp = (a3-a2)/2 + a2**
* Nếu **temp < a1** trong hàm **short loc\_8048C08:** thực thi **temp + 1**, **push** 3 giá trị **a3, temp + 1, a1** vào bộ nhớ stack và gọi hàm **func4** và cộng cho **temp,** công thức tương ứng sẽ là **result=func4(a1, temp + 1, a3) + temp**
* Nếu không thỏa mãn thực thi hàm **short loc\_8048C08**, chương trình sẽ thực thi các lệnh

A screen shot of a computer code

Description automatically generated

* Các lệnh trên thực thi **push temp – 1, a2, a1**. Sau đó gọi **func4**  với 3 đối số đã **push** rồi cộng với **temp**. Khi đó công thức tương ứng là: **result=func4(a1, a2, temp-1)+temp**
* Khi đó, tổng quan hàm **func4** sẽ như sau:

**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;**

**}**

* Thực hiện tính toán để **result=19** khi ấy sẽ tìm được giá trị nhập đầu tiên ở đầu chương trình tạm gọi là **v3**, cũng chính là tham số đầu tiên **a1** của hàm **func4**

**\*Tính toán:**

**Dùng code sau để chạy dò kết quả với a1 từ 1 đến 14:**

**#include <iostream>**

**using namespace std;**

**int func4(int a1, int a2, int a3)**

**{**

**int result;**

**int v4;**

**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;**

**}**

**int main()**

**{**

**for(int i=1; i<=14; i++)**

**cout << func4(i, 0, 14) << endl;**

**}**

* Ta được kết quả:

A screenshot of a white page

Description automatically generated

Khi đó **a1=4** sẽ cho ra **result=19,** Vậy số đầu tiên nhập vào là **4(2)**

* **Từ (1) và (2) suy ra input nhập vào là: 4 19**