리버싱

문제해결 시나리오

악성코드분석 및 모의해킹 전문가 양성과정

김 다 솜

차 례

1. 제시된 프로그램(test.exe)을 대상으로 다음 내용에 알맞은 답을 제시하라.

1-1. 제시된 프로그램을 C언어 코드로 복원한다.

```
#include<windows.h>
#include<stdlib.h>
#include<stdio.h>

typedef struct{
    char *st_one;
}STDATA;

typedef struct{
    char ch;
    int num;
}STDATATWO;

int Func1(char *lpCmd);
STDATA *Func2(char *lpCmd, int *n1);
void Func3(STDATA *st, char *lpCmd);
void Func4(char *st_one);
STDATATWO *Func4_1(char str);
void Func5(char *str);
void Func6(STDATA *st);
void Func7(STDATA *st);

int WINAPI WinMain(HINSTANCE hInstance,
                   HINSTANCE hPrevInstance,
                   LPSTR lpCmd,
                   int nShowCmd){

    STDATA *st; //ebp-4
    int n1 = 1; //ebp-8
    int i;      //ebp-c
    //LPSTR lpCmd == char *lpCmd// ebp+10

    //함수 1) argument의 문자열 길이 확인하는 함수
    Func1(lpCmd);
    if(*lpCmd == NULL){ // argument 예외처리
        MessageBox(0,"Insert An Argument, Plz", 0,0);
        return 0;
    }

    //함수 2) argument의 요소가 공백인지 확인하고 동적할당
    st = Func2(lpCmd, &n1);

    //함수 3) 구조체의 멤버변수에 argument값을 저장(대입)
    Func3(st, lpCmd);

    if(n1>0){ // 공백확인
        Func4(st->st_one);//함수 4) 생성될 문자열 1번째 값 변환
                          //        대, 소문자, 숫자에 따라 변환

        if(n1>1){ // 공백개수 > 1 인경우
            Func5(st[1].st_one); //함수 5) 생성될 문자열 2번째 값 변환
                                 //        XOR화
        }
        if(n1>2){ // 공백개수 > 2 인경우
            Func6(st);//함수 6) 생성될 문자열 3번째 값 변환
            Func7(st);//함수 7) 생성된 문자열과 비교값을 비교하는 함수
            for(i=0; i<3; i++){
                delete(st[i].st_one);//동적할당해제
            }
            delete(st);//동적할당해제
        }
    }
    return 0;
}
```

1-1. 제시된 프로그램을 C언어 코드로 복원한다.

```
int Func1(char *lpCmd){
    int i;//ebp-4

    for(i=0; lpCmd[i]; i++);

    if(i<6){
        MessageBox(0,"String must contain more than 6 chars","ITBank",0);
        return -1;
    }
    else if(i>20){
        MessageBox(0,"Stirng must contain less than 20 chars","ITBank",0);
        return 1;
    }
    return 0;
}
```

> 함수 1: argument 문자열 길이 확인

```
STDATA *Func2(char *lpCmd, int *n1){
    int ns1 = 0;  //ebp-10
    int ns2 = -1; //ebp-c
    int i;        //epb-8
    STDATA *st = new STDATA[3]; //ebp-4

    for(i=0; lpCmd[i]; i++){
        if(lpCmd[i] == ' '){ // argument 요소가 공백인지 확인
            st[ns1++].st_one = new char[i-ns2];
            //int형으로 동적할당하면 switch연산이 추가된다.
            ns2 = i; // char[1]로 유지
            *n1 = *n1+1; // 공백이면 증가(공백 수)
        }
    }
    st[ns1++].st_one = new char[i-ns2];
    return st;
}
```

> 함수 2: 공백인 경우, 공백횟수(*n1)를 증가시키고
공백을 제외한 나머지를 앞으로 이동하도록 한다.
예) _hello -> hello

```
void Func3(STDATA *st, char *lpCmd){
    int i;      //ebp-4
    int n1 = 0; //ebp-8
    int n2 = 0; //ebp-C
    for(i=0; lpCmd[i]; i++){
        st[n2].st_one[i-n1] = lpCmd[i]; //멤버변수의 요소에 argument 요소 대입
        if(lpCmd[i] == ' '){ // 공백인지 확인
            st[n2++].st_one[i-n1] = 0; //공백인 경우 0을 대입
            n1 = i+1;
        }
    }
    st[n2].st_one[i-n1]=0;
}
```

> 함수 3: 공백이 확인되면 구조체를 추가(st[n2++])하여 문자열을 별도로 저장한다.

예) hello world hi

st.st_one = “hello”, st[1].st_one = “world”, st[2].st_one = “hi”

1-1. 제시된 프로그램을 C언어 코드로 복원한다.

```
void Func4(char *st_one){ //st.st_one에 저장
    int i; //ebp-4

    for(i=0; st_one[i]; i++){ //멤버변수 = 문자+5;
        STDATATWO *stwo=Func4_1(st_one[i]);
        //EDX = st_one[i]
        st_one[i] = (st_one[i]-stwo->ch + 5) % stwo->num+stwo->ch;
        //멤버변수 = (멤버변수.문자+5) % (멤버변수.숫자+멤버변수.문자)
        //예) 소문자인경우
        //멤버변수 = ('a'+5) % (26+'a') = 102 % 123 = 102
        //예) 대문자인경우
        //멤버변수 = ('A'+5) % (26+'A') = 70 % 91 = 70
        //예) 숫자인경우
        //멤버변수 = ('0'+5) % (10+'0') = 53 % 58 = 53
    }
}

STDATATWO *Func4_1(char str){
    STDATATWO *stwo = new STDATATWO; //ebp-8

    str &= 'p';//117
    if(str>=96){ //소문자인 경우
        stwo->ch = 'a'; //97
        stwo->num = 26;
    }else if(str>=64){ //대문자인 경우
        stwo->ch = 'A'; //65
        stwo->num = 26;
    }else if(str == '0'){ //숫자인 경우
        stwo->ch = '0'; //48
        stwo->num = 10;
    }
    return stwo;
}
```

> 함수 4: 첫 번째 공백 이전의 문자열의 단일문자를 구별되어 저장되어진
멤버 변수를 이용하여 연산을 한다.
=> 결과적으로 문자(아스키코드) + 5 이다.

> 함수 4_1: 단일 문자를 'p'(117)와 AND 연산을 하여 대·소문자, 숫자를 구별하여
구조체 멤버 변수의 값을 저장한다.

```
void Func5(char *str){ // 생성될 문자열의 두 번째 값 변환
    int i;//ebp-4

    for(i=0; str[i]; i++){
        str[i] ^= 10; //XOR
    }
}
```

> 함수 5: 첫 번째 공백 이후의 문자열의 단일 문자를 10과 XOR 연산하여 저장한다.

1-1. 제시된 프로그램을 C언어 코드로 복원한다.

```
void Func6(STDATA *st){//생성될 문자열 3번째 값 변환
    char *pt; //ebp-4
    int n1;//ebp-8
    int n2;//ebp-c
    int n3;//ebp-10
    int n4;//ebp-14

    for(n2=0; st->st_one[n2]; n2++);//st->st_one의 길이확인
    for(n3=0; st[2].st_one[n3]; n3++);//st[2].st_one의 길이확인
    pt = new char[n3+1];
    for(n1=0; n1<n3; n1++){
        pt[n1]=0;
    }
    pt[n1] = 0; //마지막은 NULL값
    n1=0;
    n4 = (n1+n2)%n3;

    for(;1;){
        if(pt[n4] == 0){//공백이면
            pt[n4] = st[2].st_one[n1];
        }else{ //pt[n4] != 0
            n4++;
            continue;
        }
        n1++;
        n4 = (n1+n2) % n3;
        if(n1>=n3) break; //공백개수 확인
    }
    for(n1=0; pt[n1]; n1++){ // st[2].st_one에 저장
        (st[2].st_one[n1]) = pt[n1];
    }
}
```

> 함수 6: 첫 번째 공백 이전의 문자열의 길이를 확인하여 두 번째 공백 이후의 문자열의 순서를 변경한다.

예) re k xjodmcdongerijw

첫 번째 공백 이전의 문자열의 길이 : 2자

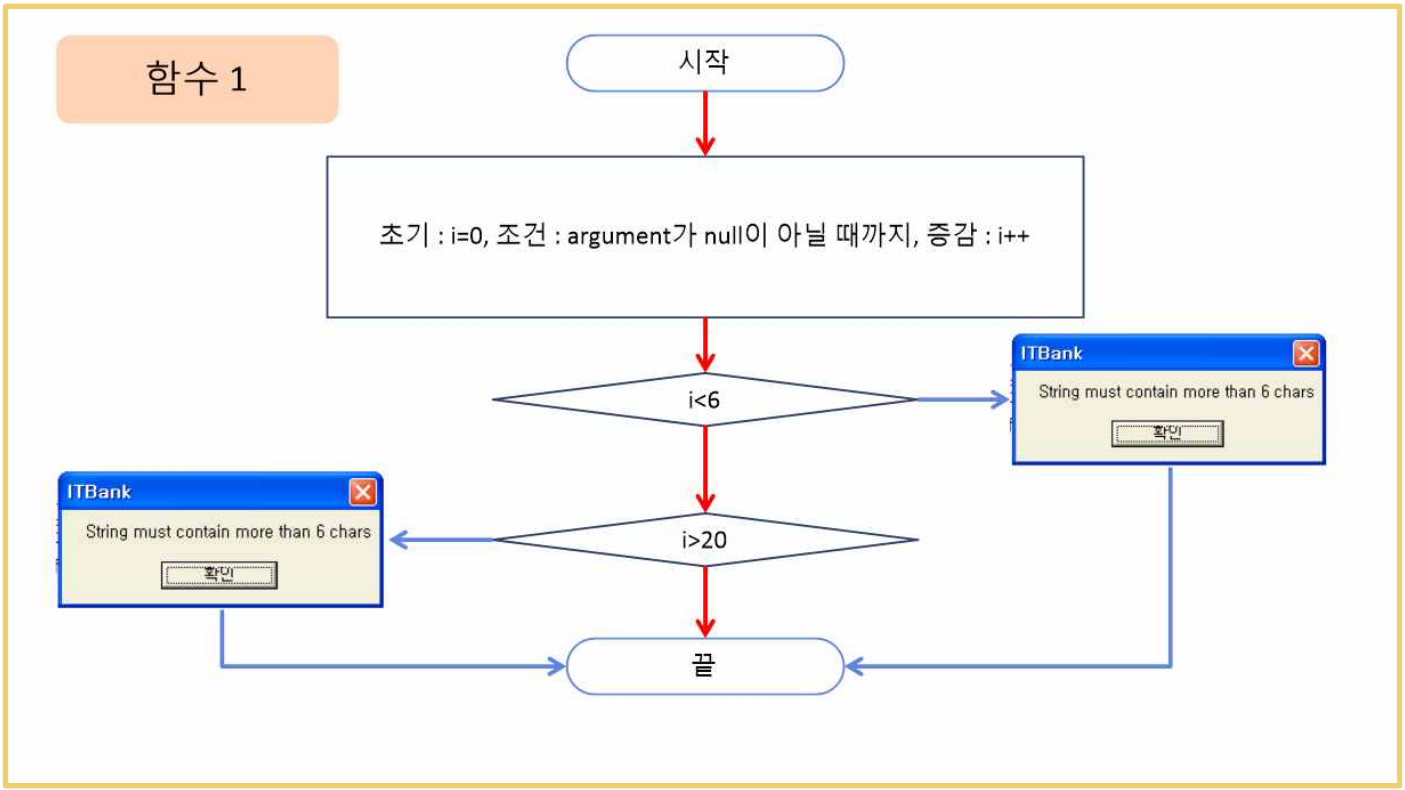
두 번째 공백 이후의 문자열의 길이 : 15자

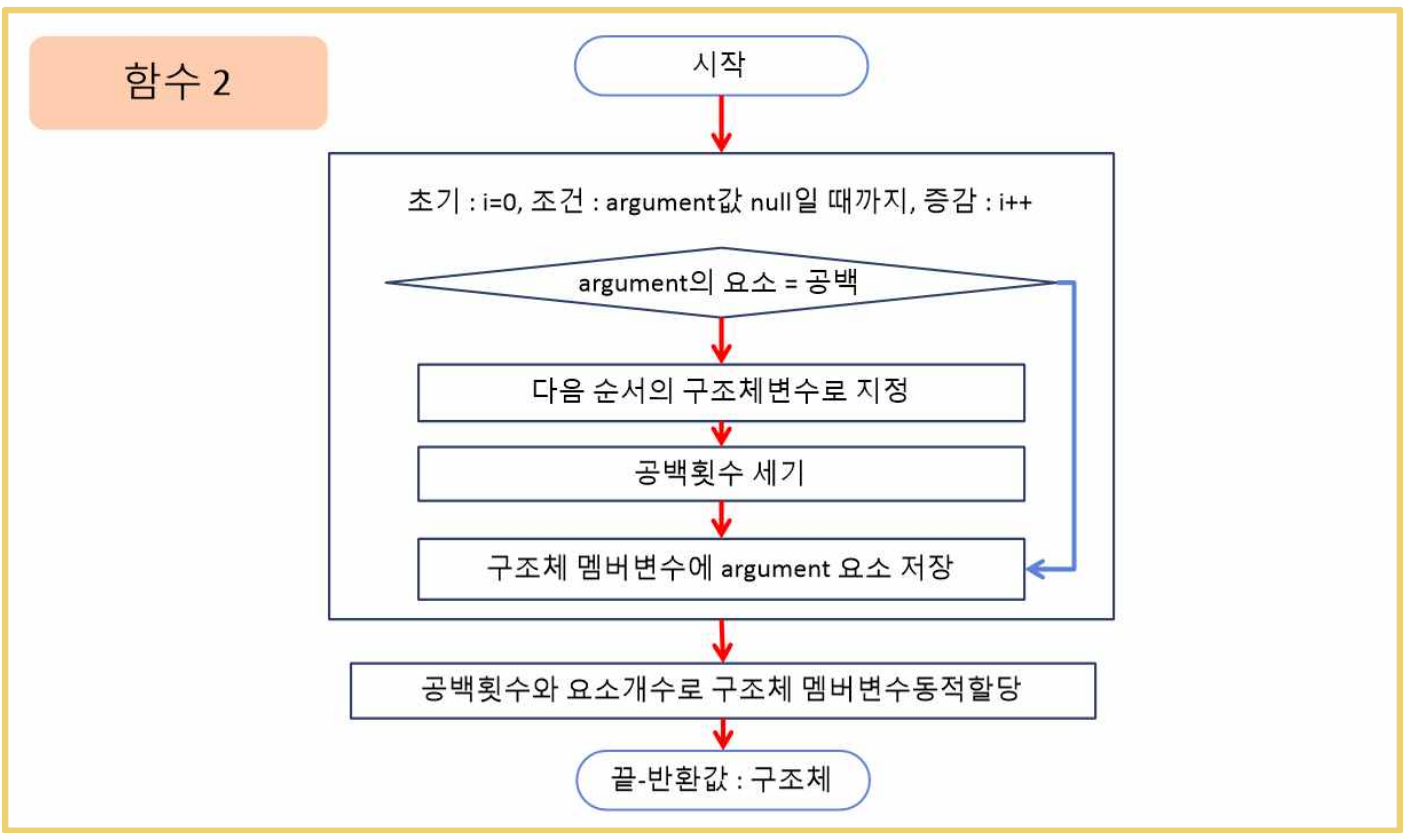
1-1. 제시된 프로그램을 C언어 코드로 복원한다.

```
void Func7(STDATA *st){
    char *str;//ebp-4
    int n1;//ebp-8
    int n2 = 0;//ebp-c
    int n3;//ebp-10

    for(n1=0; n1<3; n1++){
        for(n3=0; st[n1].st_one[n3]; n3++);
        n2 = n2+n3;
    }
    str = new char[n2+1];
    //EDX = st[2] //ECX = st->st_one //EAX = st
    sprintf(str, "%s%s%s",st->st_one,st[1].st_one,st[2].st_one);
    if(!strcmp(str,"wjajwxjodmcdongeri"))//생성된 문자열과 비교값을 비교
        MessageBox(0,"Success","ITBank",0);
    delete(str);
}
```

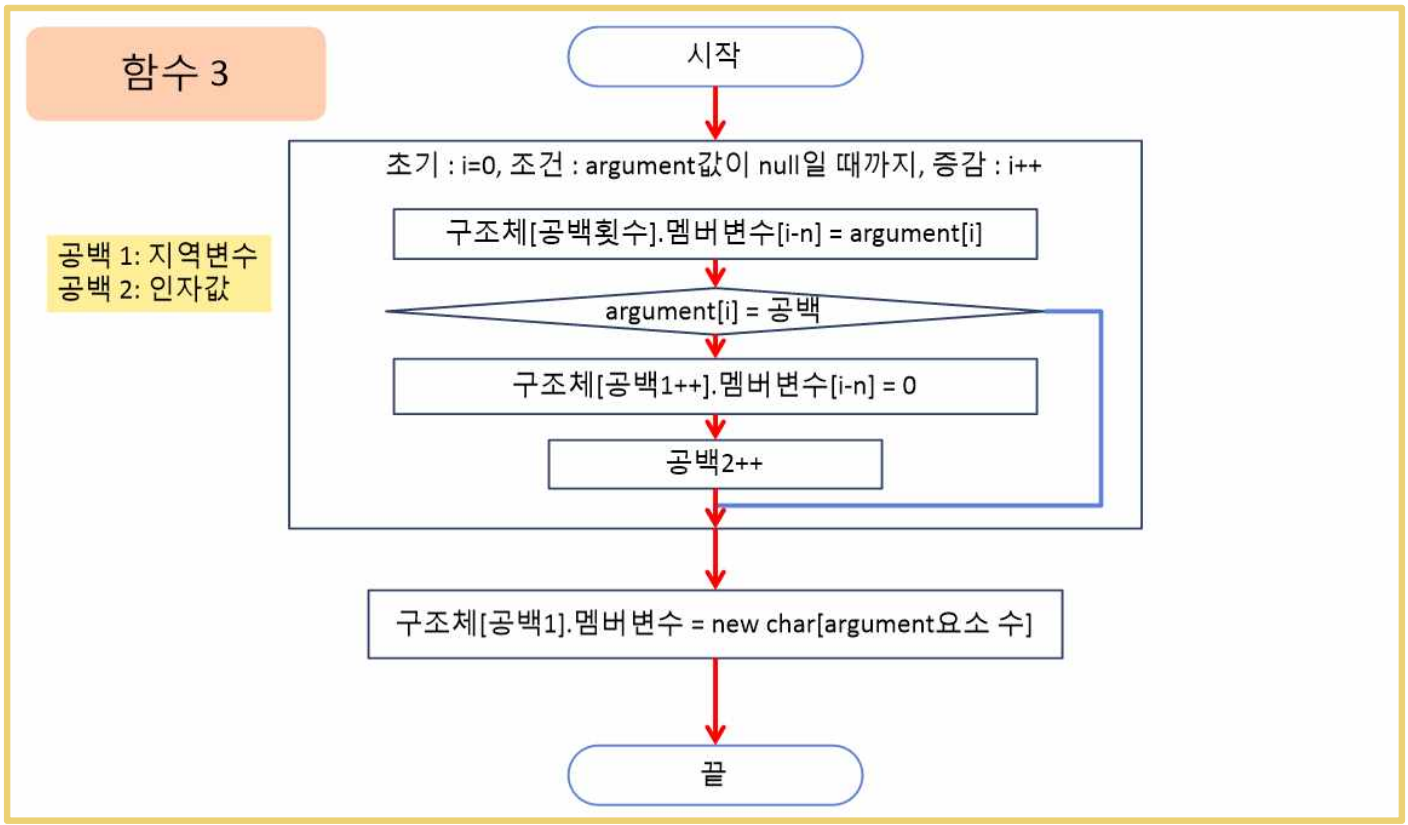
> 함수 7: 공백 2개로 나눠진 3개의 문자열의 길이를 확인하여 동적할당
그런 다음 sprintf() 함수로 3개의 문자열을 붙인다.
"wjajwxjodmcdongeri" 문자열과 비교하여 사용자가 올바른 argument를 입력했는지 확인한다.

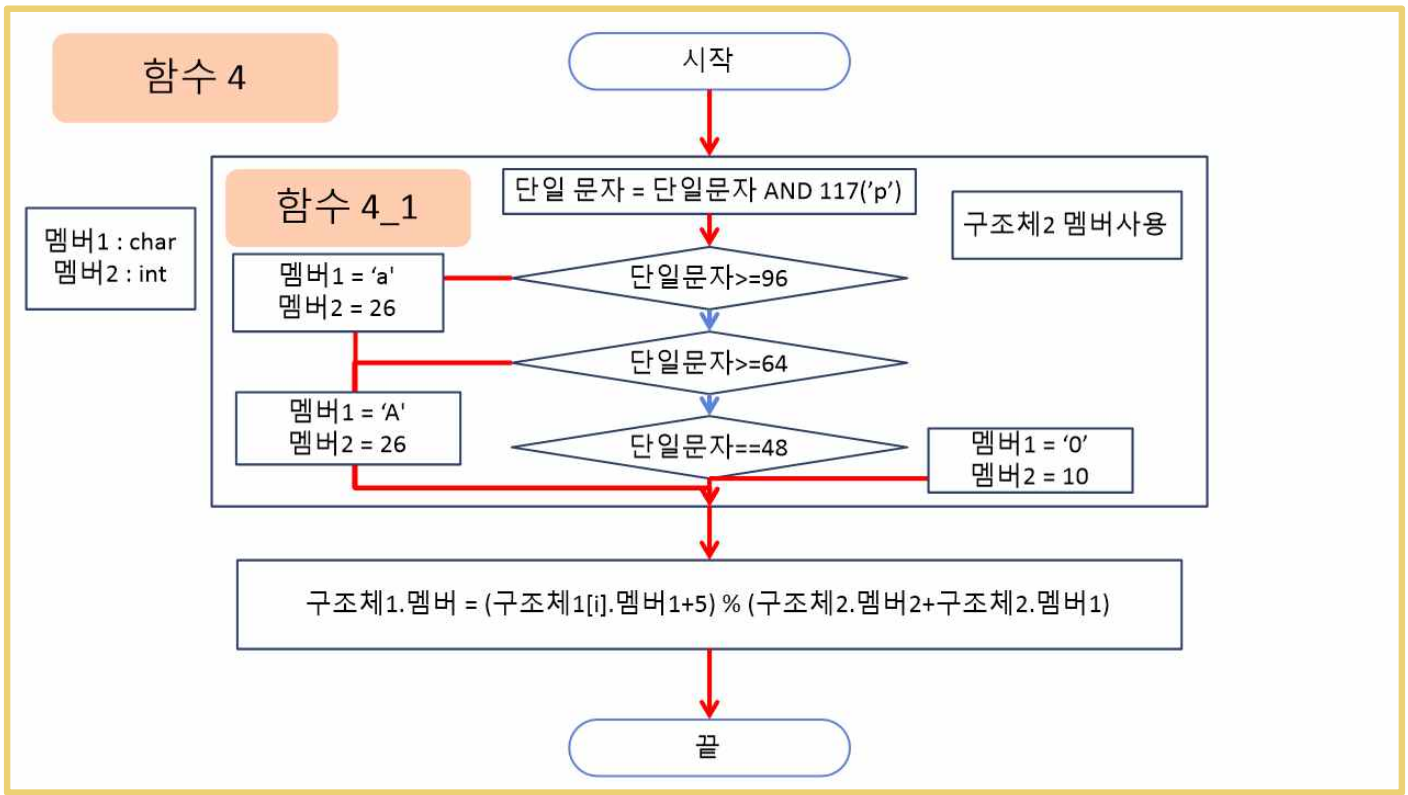
1-2. 복원된 프로그램의 원리를 파악하고 프로그램의 요구사항대로 적절한 값을 입력해 목적을 달성한다.



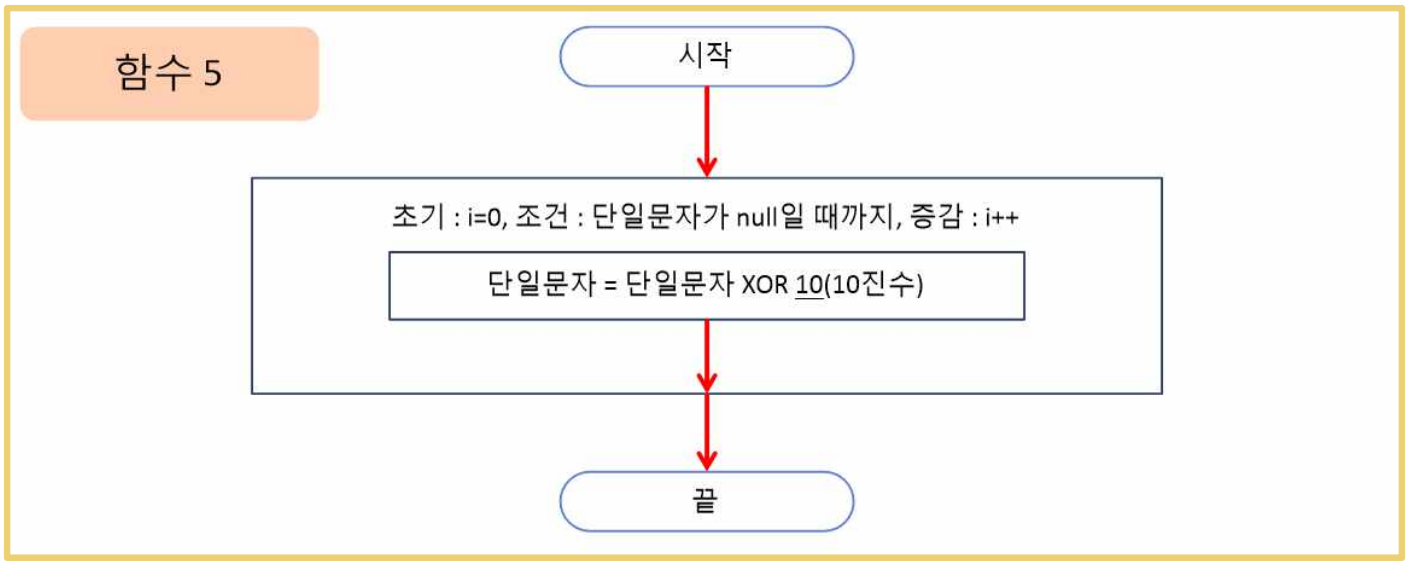


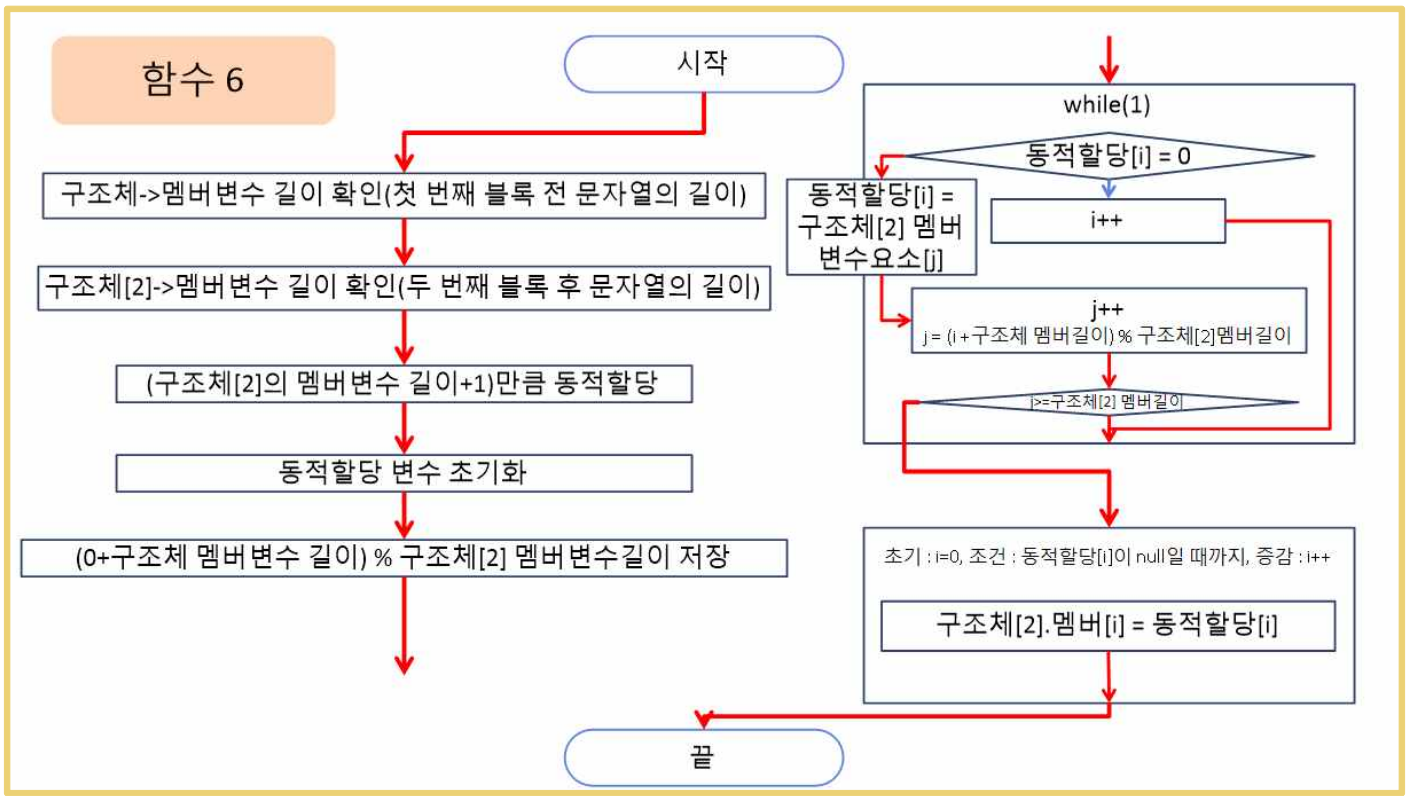
1-2. 복원된 프로그램의 원리를 파악하고 프로그램의 요구사항대로 적절한 값을 입력해 목적을 달성한다.



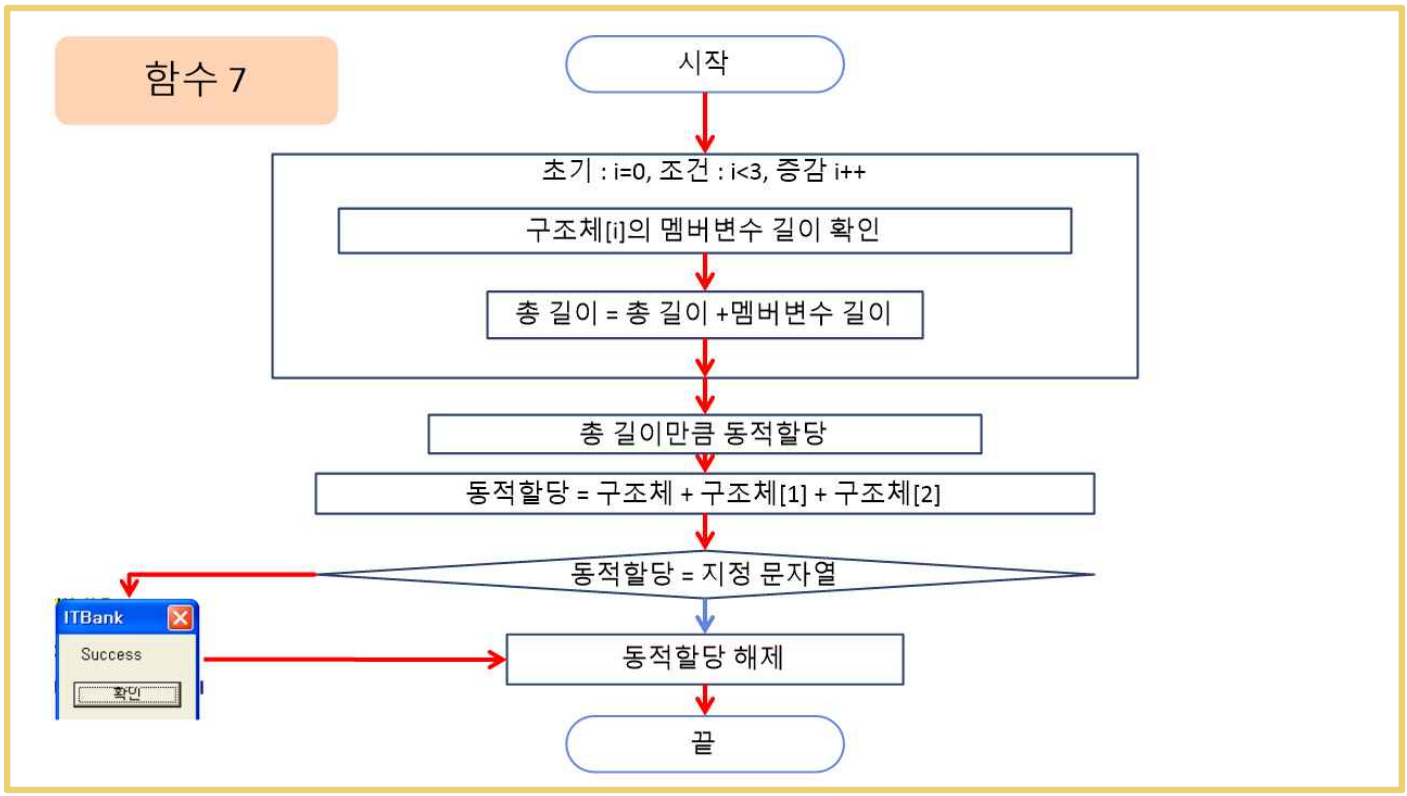


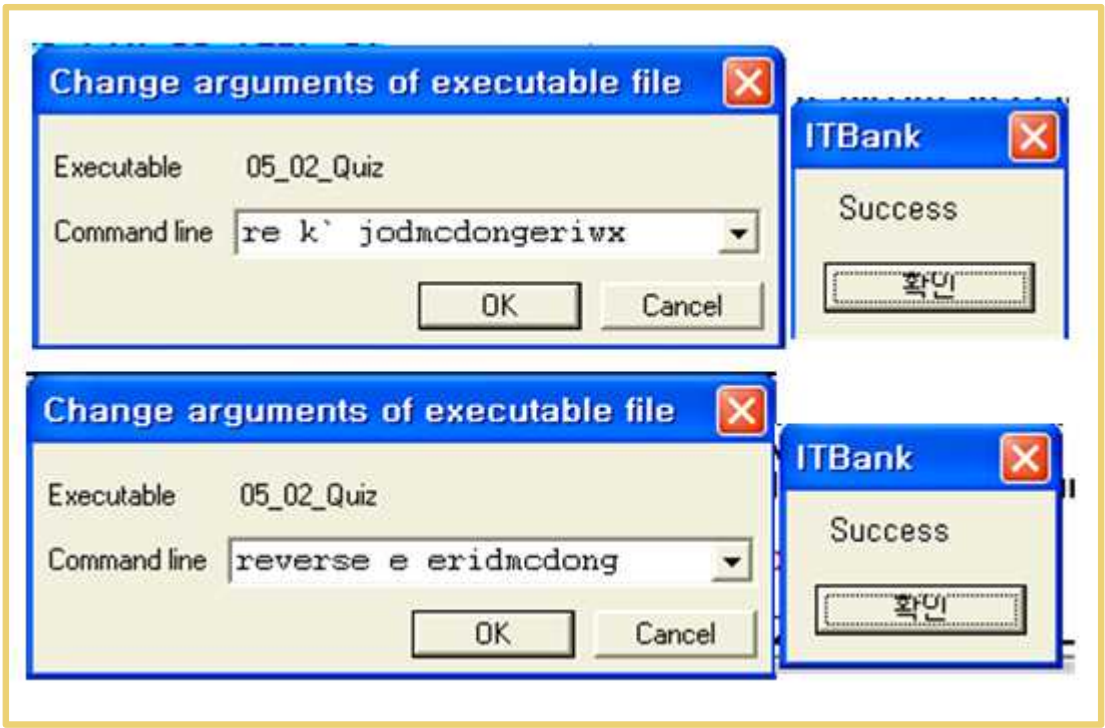
1-2. 복원된 프로그램의 원리를 파악하고 프로그램의 요구사항대로 적절한 값을 입력해 목적을 달성한다.





1-2. 복원된 프로그램의 원리를 파악하고 프로그램의 요구사항대로 적절한 값을 입력해 목적을 달성한다.





2. 제시된 프로그램(crackme.exe)을 대상으로 다음 내용에 알맞은 답을 제시하라.

2-1. crackme를 분석하여 "Next Password ..." 문자열이 출력될 수 있게 적절한 key를 찾아 입력한다.

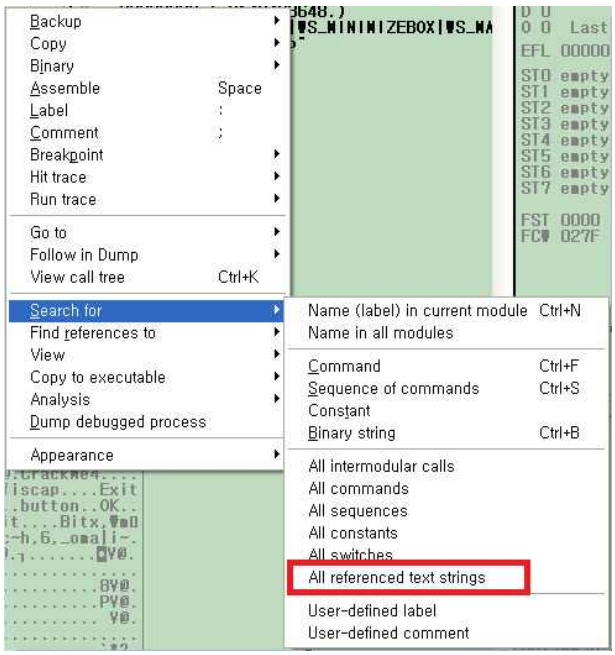
1) 적절한 문자열을 입력하지 않으면 이상한 문자열이 출력된다.

[그림 1. 임의의 문자열 입력]

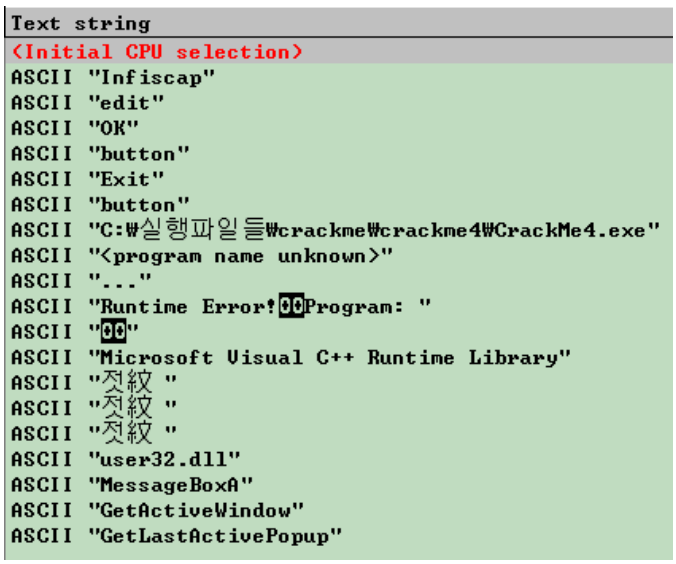
[그림 2. 적절하지 않은 문자열을 입력한 경우]

2) Windows에서 제공하는 API 함수로 만들어진 대화상자이므로
두 가지 방법으로 찾아본다.

2-1) 문자열 기준



[그림 1. All referenced text strings]

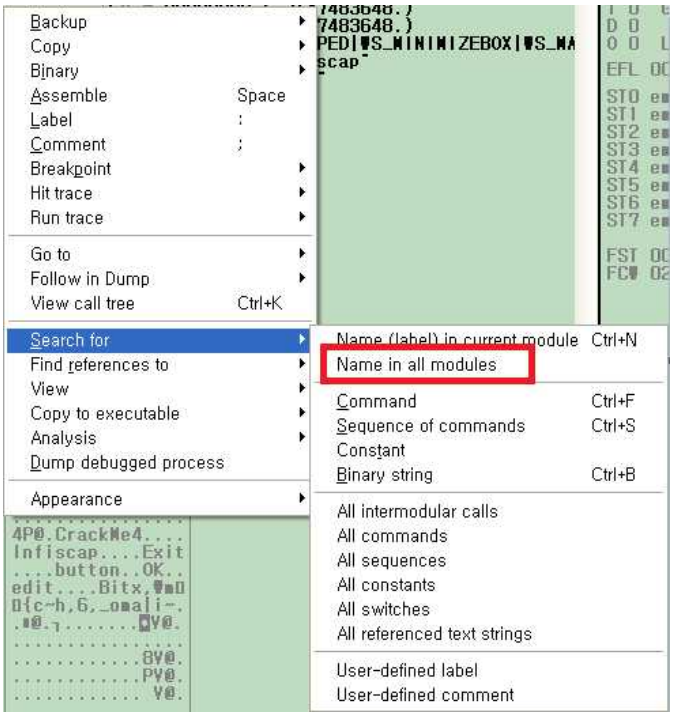


[그림 2. 참조된 모든 문자열 확인하기]

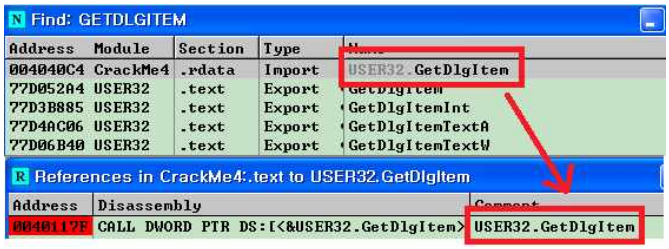
> 참조된 모든 문자열 목록을 보면 의심될만한 문자열이 없다.
> 다른 방법으로 알아봐야 한다.

2-1. crackme를 분석하여 "Next Password ..." 문자열이 출력될 수 있게 적절한 key를 찾아 입력한다.

2-2) 함수 기준



[그림 1. Name in all modules]



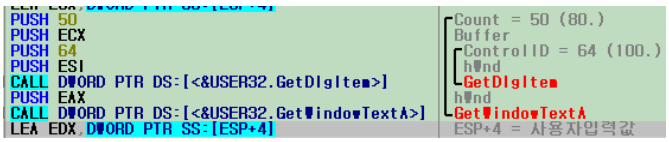
[그림 2. GetDlgItem() 함수에 접근하기]

[그림 3. GetDlgItem() 함수]

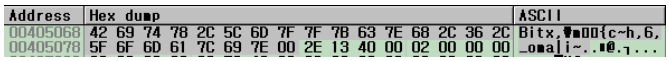
> 사용자가 입력한 입력값을 가져오는 함수

> 참조 :

https://docs.microsoft.com/en-us/windows/win32/api/winuser/nf-winuser-getdlgitem

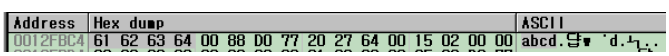


[그림 4. GetDlgItem함수로 입력값 가져오기]



[그림 6. 의심스러운 문자열 확인]

> 적절하지 않은 입력값을 넣은 경우, 출력되는 실패 문자열을 찾았다.

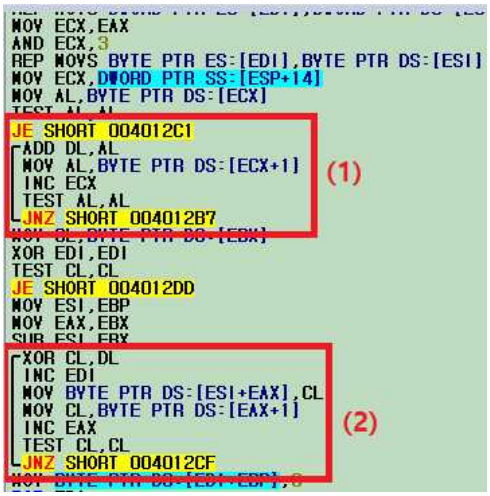


[그림 5. [ESP+4]의 주소가 가리키는 값]

> 해당 주소가 가리키는 값이 사용자가 입력한 입력값을 확인할 수 있다.

[그림 7. 해당 문자열 복사]

2-1. crackme를 분석하여 "Next Password ..." 문자열이 출력될 수 있게 적절한 key를 찾아 입력한다.



[그림 8. 문자열 저장]

> (1) : 사용자 입력값 저장

> (2) : 의심스러운 문자열 저장

```c
#include<stdio.h>

int main(){

    char str[]="Bitx,##m□□{c~h,6,_oma|i~.";
    int i, j;

    for(i=0; str[i]; i++){
        printf("i: %d : ", i);

        for(j=0; str[j]; j++)
            printf("%c", str[j]^i); // xor연산

        printf("#n");
    }
    return 0;
}
```

[그림 9. key를 찾기 위한 코드]

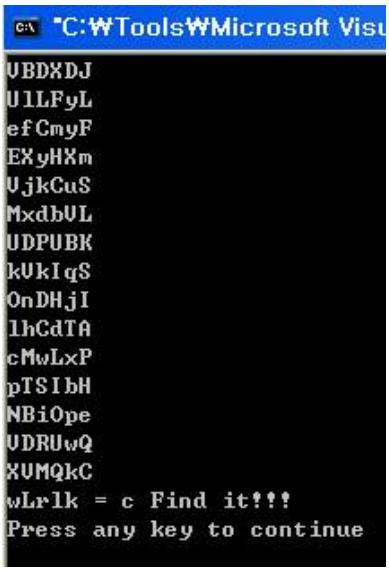
[그림 10. key가 12인 것을 확인]

> 12 = 0x0C 임을 확인

2-2. key 생성 알고리즘을 분석하여 입력값을 추출 해내는 프로그램을 작성한다.

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#include<string.h>
#include<windows.h>

int main(){

    // 예) PpL : P(0x50), p(0x70), L(0x4c) = 10C, 10진수 : 268 그래서 0C0이다.

    srand(time(NULL));

    char sum=0;
    int f, num, count = 0;
    int len = 5;// 길이조정
    while(1){
        if(sum == 12 && count == len){
            printf(" = c Find it!!!₩n",sum);
            break;
        }else if(count>len){
            sum = 0;
            num = 0;
            count = 0;
            printf("₩n");
        }
        f = (rand()%2)+1;
        if(f == 1)
            num = (rand()%25)+65;
        else
            num = (rand()%25)+97;
        sum += (char)num;
        printf("%c",num);
        count++;
    }
    return 0;
}
```

[그림 1. 입력값을 추출 해내는 코드]



[그림 2. 5자로 제한하여 나온 입력값]

[그림 3. 입력값으로 패스워드 도출]