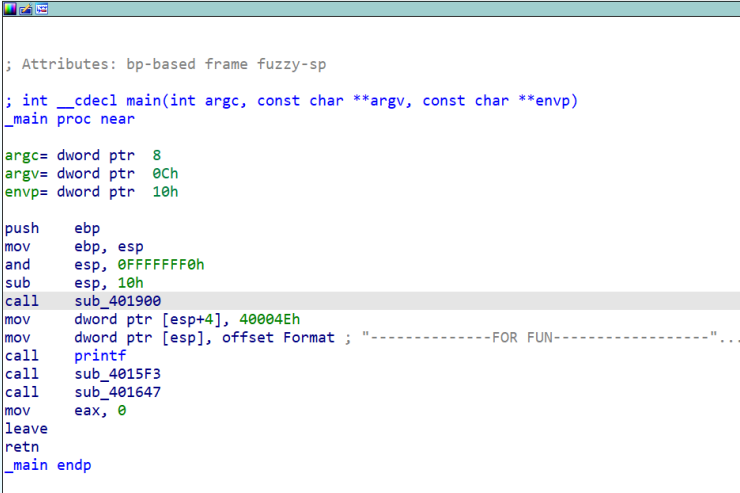


# Homework-2-1 Report

姓名：项 枫      学号：2022211570

## 一、程序的伪代码流程图

### 1、main():

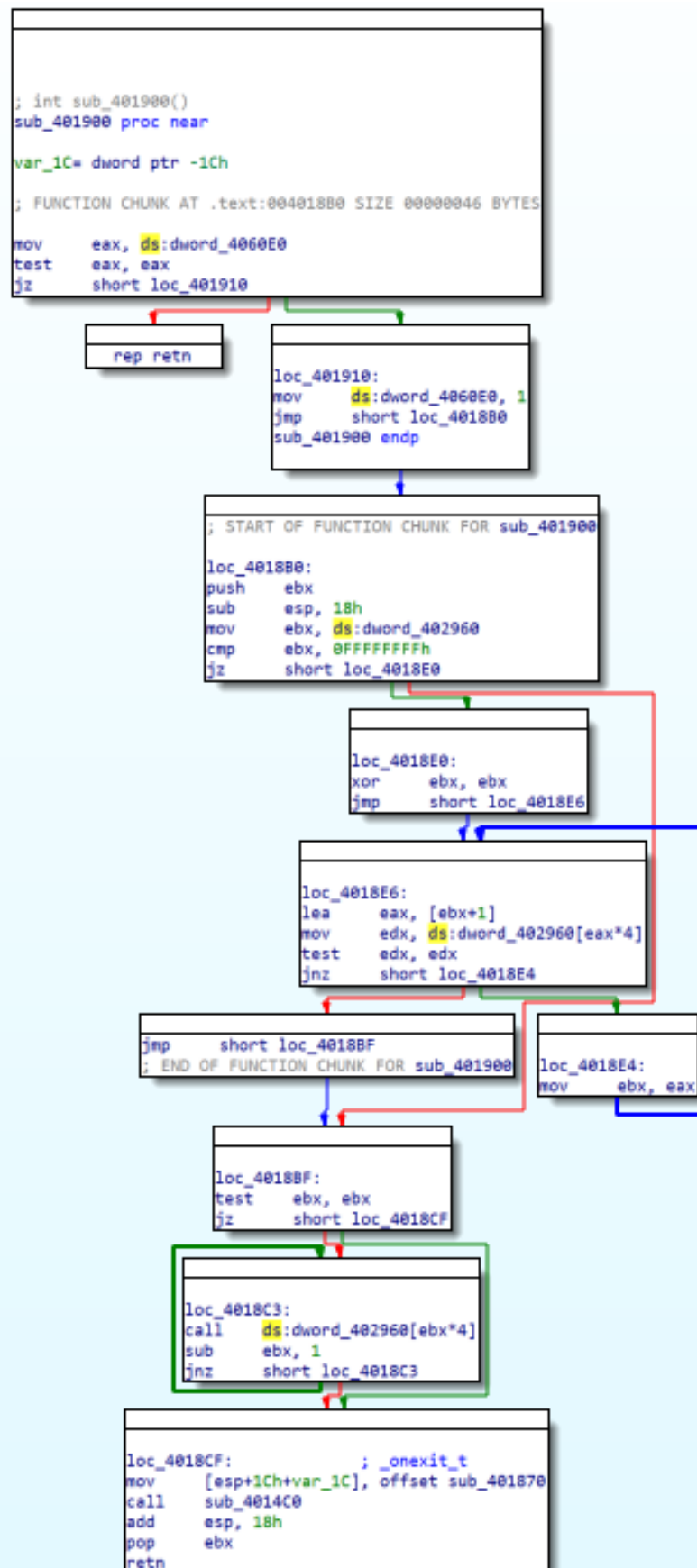


```
; Attributes: bp-based frame fuzzy-sp
; int __cdecl main(int argc, const char **argv, const char **envp)
_main proc near

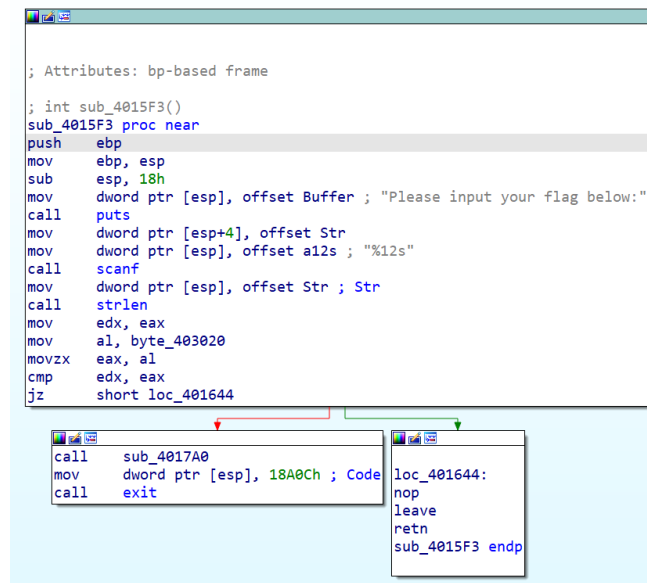
argc= dword ptr 8
argv= dword ptr 0Ch
envp= dword ptr 10h

push    ebp
mov     ebp, esp
and     esp, 0FFFFFFF0h
sub     esp, 10h
call    sub_401900
mov     dword ptr [esp+4], 40004Eh
mov     dword ptr [esp], offset Format ; "-----FOR FUN-----"...
call    printf
call    sub_4015F3
call    sub_401647
mov     eax, 0
leave
retn
_main endp
```

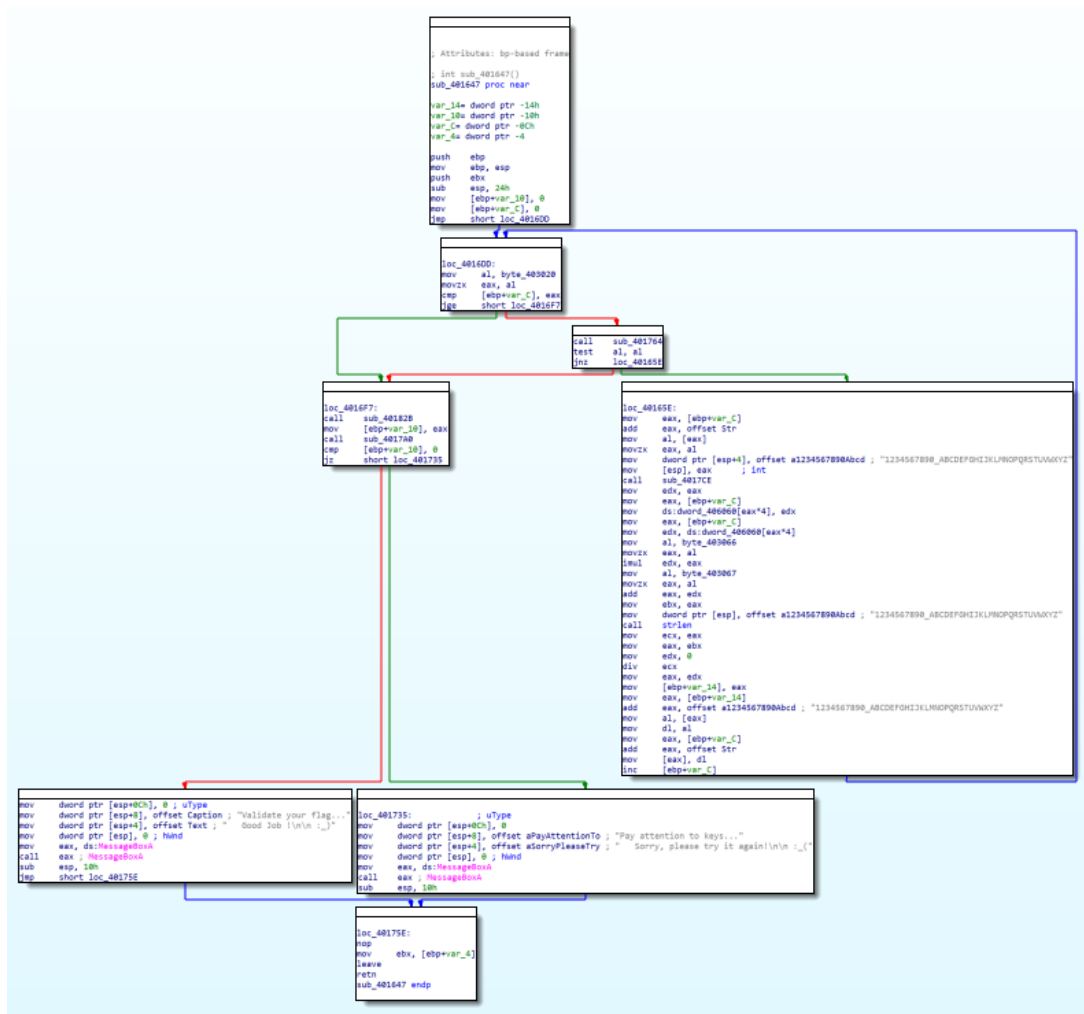
2、 sub\_401900():



### 3、sub\_4015F3():



### 4、sub\_401647():



## 二、分析字符串 “This program cannot be ...”

### 1、内存地址

由下图可知，该字符串内存地址为 0x40004E。

```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     sub_401900();
4     printf(
5         "-----FOR FUN-----\n%s\n-----\n\n",
6         (const char *)0x40004E);
7     sub_4015F3();
8     sub_401647();
9     return 0;
10 }
```

### 2、字符串长度为 39。

## 三、逆向分析与思考的逻辑过程

### 1、程序函数分析

#### (1) main 函数

```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     sub_401900();
4     printf(
5         "-----FOR FUN-----\n%s\n-----\n\n",
6         (const char *)0x40004E);
7     sub_4015F3();
8     sub_401647();
9     return 0;
10 }
```

调用了 sub\_4015F3(), sub\_401647()函数，将分别在 (2) (3) 进行分析。

#### (2) sub\_4015F3()函数

```
1 int sub_4015F3()
2 {
3     size_t v0; // edx
4     int result; // eax
5
6     puts("Please input your flag below:");
7     scanf("%12s", Str);
8     v0 = strlen(Str);
9     result = (unsigned __int8)byte_403020;
10    if ( v0 != (unsigned __int8)byte_403020 )
11    {
12        sub_4017A0();
13        exit(100876);
14    }
15    return result;
16 }
```

第 6 行：输出字符串 “Please input your flag below:”

第 7 行：读入字符串 Str，此处 “%12” 可知字符串长度为 12。

第 8 行：计算 Str 长度 v0。

第 9 行：result 值为 12。如下图，byte\_403020 的值为 0Ch，十进制值即为 12。

```
00+align 20h
byte_403020 db 0Ch
```

第 10 行：判断字符串 Str 长度是否为 12。

第 12、13 行：字符串 Str 长度不为 12，跳转 sub\_4017A0() 函数。如下图，

该函数功能为将 a04footfracd9 数组初始化为全 0，并且 exit。

```
1 char sub_4017A0()
2 {
3     int i; // [esp+Ch] [ebp-4h]
4
5     for ( i = 0; i < (unsigned __int8)byte_403020; ++i )
6         a04footfracd9[i] = 0;
7     return 1;
8 }
```

### (3) sub\_401647() 函数

```
1 int sub_401647()
2 {
3     int v1; // [esp+18h] [ebp-10h]
4     int i; // [esp+1Ch] [ebp-Ch]
5
6     for ( i = 0; i < (unsigned __int8)byte_403020 && (unsigned __int8)sub_401764(); ++i )
7     {
8         dword_406060[i] = sub_4017CE((unsigned __int8)Str[i], a1234567890Abcd);
9         Str[i] = a1234567890Abcd[((unsigned __int8)byte_403066 * dword_406060[i] + (unsigned int)(unsigned __int8)byte_403067)
10             % strlen(a1234567890Abcd)];
11     }
12     v1 = sub_40182B();
13     sub_4017A0();
14     if ( v1 )
15         return MessageBoxA(0, " Good Job !\n\n :)", "Validate your flag...", 0);
16     else
17         return MessageBoxA(0, " Sorry, please try it again!\n\n :(", "Pay attention to keys...", 0);
18 }
```

第 6 行：for 循环，循环 12 次，将输入字符串进行加密（第 8-10 行）；此处

需要注意 sub\_401764()函数，如下图，每次循环都更新 byte\_403066 和 byte\_403067 的值，初值为 12h（对应十进制 18）和 11h（对应十进制 17），执行该函数可知，i 为偶数时，byte\_403066 =3，byte\_403067=18；i 为偶数时，byte\_403066 =17，byte\_403067=3。

```
1 char sub_401764()  
2 {  
3     byte_403066 ^= byte_403067;  
4     byte_403067 ^= byte_403066;  
5     if ( byte_403066 == 18 )  
6         sub_401764();  
7     return 1;  
8 }
```

byte\_403066 db 12h

byte\_403067 db 11h

第 8 行：执行 sub\_4017CE()函数，将函数值赋给 dword\_406060[i]，如下图，该函数功能为：查找字符 Str[i]在字符集 a1234567890Abcd 中的位置，由下图可知该字符集为 1234567890\_ABCDEFGHIJKLMNOPQRSTUVWXYZ，并返回该位置，为第 9 行进行加密过程做准备。

```
1 char __cdecl sub_4017CE(char a1, char *Str)  
2 {  
3     char result; // a1  
4     size_t i; // [esp+2Ch] [ebp-Ch]  
5  
6     for ( i = 0; strlen(Str) > i; ++i )  
7     {  
8         if ( a1 == Str[i] )  
9             return i;  
10    }  
11    result = Str[i];  
12    if ( a1 != result )  
13        exit(100876);  
14    return result;  
15 }
```

```

; char a1234567890Abcd[]
a1234567890Abcd db '1234567890_ABCDEFGHIJKLMNOPQRSTUVWXYZ',0
; DATA XREF: sub_401647
; sub_401647+61fo
; sub_401647+80fo
; DATA XREF: sub_401647
byte_403066 db 12h

```

第 9、10 行：仿射加密过程。Str[i] = byte\_403066 \* dword\_406060[i] + byte\_403067 % strlen(a1234567890Abcd)。

第 11 行：执行 sub\_40182B() 函数，将函数值赋给 v1，如下图，该函数功能为：判断加密后的字符串 Str 与 a04footfracd9 的字符串（由下图可知该字符串为 04FOOTFRACD9）是否相等，相等返回 1，否则返回 0。

```

1 int sub_40182B()
2 {
3     int i; // [esp+Ch] [ebp-4h]
4
5     for ( i = 0; i < (unsigned __int8)byte_403020; ++i )
6     {
7         if ( Str[i] != a04footfracd9[i] )
8             return 0;
9     }
10    return 1;
11 }

```

```

a04footfracd9 db '04FOOTFRACD9',0
; align 10h

```

第 13 行：执行 sub\_4017A0() 函数，如下图，该函数功能为：将字符串 a04footfracd9 初始为 000000000000。

```

1 char sub_4017A0()
2 {
3     int i; // [esp+Ch] [ebp-4h]
4
5     for ( i = 0; i < (unsigned __int8)byte_403020; ++i )
6         a04footfracd9[i] = 0;
7     return 1;
8 }

```

第 14-17 行：如果 v1 为 1，则输出 “ Good Job !\n\n:\_)”，证明 flag 通过；如果 v1 为 0，则输出 “ Sorry, please try it again!\n\n:\_(”，证明 flag 未通过。由此便可知，第 11 行 a04footfracd9 处的字符串（04FOOTFRACD9）即为密文。

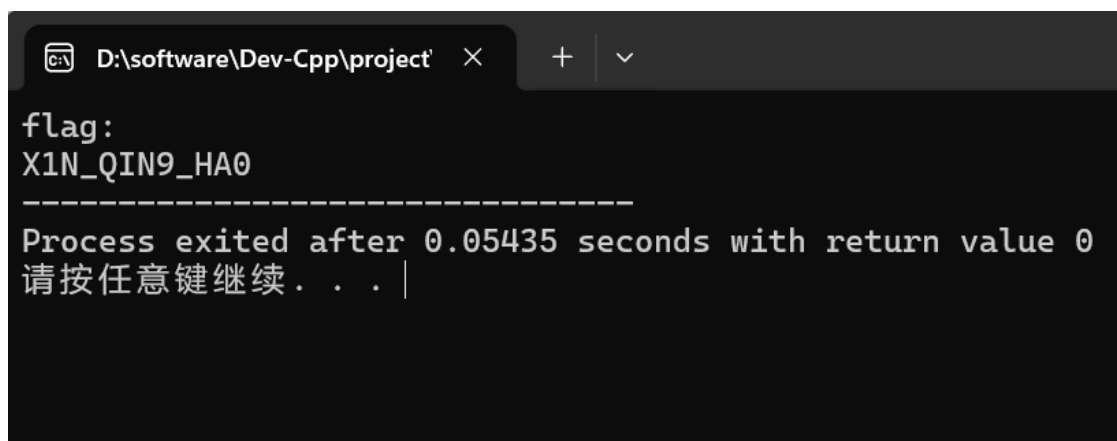
## 2、解 flag 程序

(1) 程序代码 (C++) 如下：

```
1. #include<bits/stdc++.h>
2. using namespace std;
3. int main(){
4.
5.     int key_1_1=3;//i 为偶数时的 key_1
6.     int key_2_1=18;//i 为偶数时的 key_2
7.     int re_key_1_1=25;//key_1_1 的逆
8.     int key_1_2=17;//i 为奇数时的 key_1
9.     int key_2_2=3;//i 为奇数时的 key_2
10.    int re_key_1_2=24;//key_1_2 的逆
11.    char c[]="04FOOTFRACD9";//密文
12.    char dic[]="1234567890_ABCDEFGHIJKLMNOPQRSTUVWXYZ";//字符集
13.    char result[20]={0};//结果 flag
14.    int i,j,c_len,dic_len,temp;
15.
16.    c_len=strlen(c);//密文长度
17.    dic_len=strlen(dic);//字符集长度
18.    for(i=0;i<c_len;i++){
19.        for(j=0;j<dic_len;j++){
20.            if(dic[j]==c[i]) temp=j;
21.        }
22.        if(i%2==0) temp=((temp-
            key_2_1+dic_len)*re_key_1_1)%dic_len;//i 为偶数时解密
23.        else temp=((temp-key_2_2+dic_len)*re_key_1_2)%dic_len;//i 为奇
            数时解密
24.        result[i]=dic[temp];
25.    }
26.
27.    cout<<"flag:"<<endl;
28.    cout<<result;
29.    return 0;
30. }
```

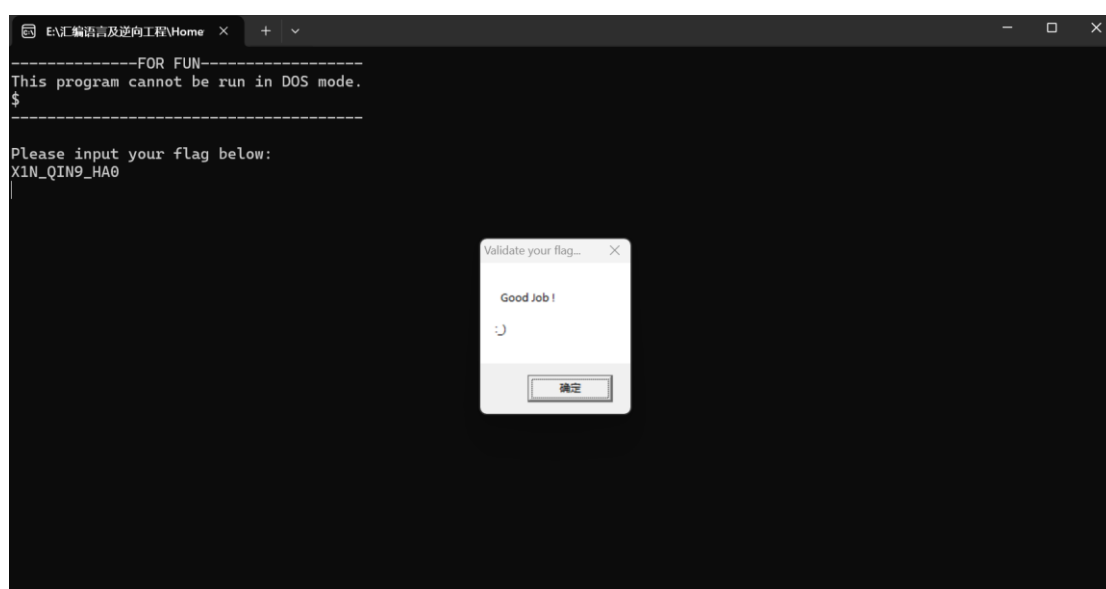


## (2) 程序运行结果



```
D:\software\Dev-Cpp\project X + v
flag:
X1N_QIN9_HA0
-----
Process exited after 0.05435 seconds with return value 0
请按任意键继续. . . |
```

## 3、flag 成功通过



## 四、收获和感受

最开始忽略了 `sub_401764()` 函数，误认为密钥对为 (18, 17)，导致 flag 一直不通过，最后在冷静的观察与分析下，理解了该函数的功能，密钥对是 (3, 18) 和 (17, 3) 的反复循环，最终 flag 成功通过！

通过本次作业，对 IDApro 软件的使用有了更深的理解，也初步学会了逆向分析。

最后，感谢潘老师课上辛勤的教学。