

```

1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     int v4[4]; // [esp+4h] [ebp-14h] BYREF
4
5     printf(
6         "Supported authentication methods:\n"
7         "1. Hard-coded password\n"
8         "2. A pair of 2 numbers\n"
9         "3. Username/password\n"
10        "Enter your choice: ");
11    __isoc99_scanf("%d", v4);
12    fflush(stdin);
13    if ( v4[0] == 1 )
14    {
15        hardCode();
16    }
17    else if ( v4[0] == 2 )
18    {
19        otherhardCode();
20    }
21    else
22    {
23        if ( v4[0] != 3 )
24        {
25            puts("Invalid authentication method.");
26            exit(0);
27        }
28        userpass();
29    }
30    return 0;
31 }

```

Handwritten annotations on the first image:

- A red arrow points from the number "1" to the `hardCode()` function call on line 15.
- A red arrow points from the number "2" to the `otherhardCode()` function call on line 19.
- A red arrow points from the number "3" to the `userpass()` function call on line 28.

1)

```

1 int hardCode()
2 {
3     char s1[1008]; // [esp+8h] [ebp-3F0h] BYREF
4
5     getchar();
6     puts("Enter the hard-coded password(option 1):");
7     __isoc99_scanf("%[^\\n]", s1);
8     printf("Your input hard-coded password: %s\\n", s1);
9     if ( !strcmp(s1, "With age comes wisdom") )
10        return success_1();
11    else
12        return failed();
13 }

```

Handwritten annotations on the second image:

- A red arrow points from the text "s1 is ur input" to the `s1` variable on line 7.
- A red arrow points from the text "if s1 == 'With'" to the `strcmp` function call on line 9.
- Below the code, it says: `=> pwd : With age comes wisdom`

2)

```
1 int otherhardCode()
2 {
3     int v0; // edx
4     int v2; // [esp+4h] [ebp-14h] BYREF
5     int v3[4]; // [esp+8h] [ebp-10h] BYREF
6
7     getchar();
8     puts("Enter your 2 numbers (separated by space) (option 2):");
9     __isoc99_scanf("%d %d", v3, &v2);
10    printf("Your input: %d %d\n", v3[0], v2);
11    v3[1] = 9;
12    if ( v3[0] == 9 && (v0 = funny_func(9, funny_seq[9]), v0 == v2) )
13        return success_2();
14    else
15        return failed();
16 }
```

v3 = first input
v2 = second input

→

$$\left\{ \begin{array}{l} v3 == 9 \\ v0 = \text{funny_func}(9, \text{funny_seq}[9]) \\ v0 == 2 \end{array} \right. \quad (\text{always true})$$

— Let's calculate v0: `(v0 = funny_func(9, funny_seq[9]),`

```

.rodata:08048B60 ; int funny_seq[10]
.rodata:08048B60 funny_seq dd 0Ah ; DATA XREF: otherhardCode+5D↑r
.rodata:08048B64 db 3
.rodata:08048B65 db 0
.rodata:08048B66 db 0
.rodata:08048B67 db 0
.rodata:08048B68 db 6
.rodata:08048B69 db 0
.rodata:08048B6A db 0
.rodata:08048B6B db 0
.rodata:08048B6C db 9
.rodata:08048B6D db 0
.rodata:08048B6E db 0
.rodata:08048B6F db 0
.rodata:08048B70 db 1
.rodata:08048B71 db 0
.rodata:08048B72 db 0
.rodata:08048B73 db 0
.rodata:08048B74 db 4
.rodata:08048B75 db 0
.rodata:08048B76 db 0
.rodata:08048B77 db 0
.rodata:08048B78 db 7
.rodata:08048B79 db 0
.rodata:08048B7A db 0
.rodata:08048B7B db 0
.rodata:08048B7C db 2
.rodata:08048B7D db 0
.rodata:08048B7E db 0
.rodata:08048B7F db 0
.rodata:08048B80 db 5
.rodata:08048B81 db 0
.rodata:08048B82 db 0
.rodata:08048B83 db 0
.rodata:08048B84 db 8
.rodata:08048B85 db 0
.rodata:08048B86 db 0
.rodata:08048B87 db 0
.rodata:08048B88 aYouScratchMyBa db 'You scratch my back and I',27h,'ll scratch yours',0

```

Handwritten annotations on the memory dump:

- Indices [0] through [9] are written in red next to the corresponding memory addresses.
- The value 8 is written in red next to the memory address 08048B79, with the text `funny_seq[9] = 8` written in red.
- An arrow points from the value 8 to the memory address 08048B84.

```

1 int __cdecl funny_func(int a1, int a2)
2 {
3     return a1 * (a1 + a2) + a2;
4 }

```

Handwritten annotations on the code block:

- The value 9 is written in red above the parameter `a1`.
- The value 8 is written in red above the parameter `a2`.
- Arrows point from the handwritten 9 and 8 to the parameters `a1` and `a2` respectively.

$$\begin{aligned}
 \Rightarrow v0 &= \text{funny_func}(9, 8) \\
 &= 9 * (9 + 8) + 8 = 161
 \end{aligned}$$

So 2 numbers we need is

9 161

3)

```
1 int userpass()
2 {
3     size_t v0; // ebx
4     long double v2; // fst7
5     size_t v3; // eax
6     size_t v4; // edx
7     char v5[9]; // [esp+1Ah] [ebp-2Eh]
8     char v6[10]; // [esp+23h] [ebp-25h] BYREF
9     char s[10]; // [esp+2Dh] [ebp-1Bh] BYREF
10    char v8[5]; // [esp+37h] [ebp-11h] BYREF
11    int i; // [esp+3Ch] [ebp-Ch]
12
13    qmemcpy(v8, "cRvlg", sizeof(v8)); v8 = 'cRvlg'
14    getchar();
15    puts("Enter your username:");
16    __isoc99_scanf("%[^\n]", s); → your username
17    getchar();
18    puts("Enter your password:");
19    __isoc99_scanf("%[^\n]", v6); → your pwd
20    printf("Your input username: %s and password: %s\n", s, v6);
```

```

if ( strlen(s) != 9 )
    return failed();
v0 = strlen(s);
if ( v0 != strlen(v6) )
    return failed();
for ( i = 0; i <= 8; ++i )
{
    if ( i > 1 )
    {
        if ( i > 3 )
            v5[i] = v8[8 - i];
        else
            v5[i] = s[i + 2];
    }
    else
    {
        v5[i] = s[i + 5];
    }
}

```

len(s) = len(v6)

SystemProgramming / Lab4 / pwd_gen.py / ...

```

1  username = input("---> Enter your username: ")
2  # username = "399076724"
3
4  v8 = "cRVlg" # random string
5  newStr = ''
6
7  for i in range(0, len(username)):
8      if i > 1:
9          if i > 3:
10             newStr += v8[8-i]
11         else:
12             newStr += username[i+2]
13     else:
14         newStr += username[i+5]
15
16  print("(+) New generated string: " + newStr)
17

```

v5

```

for ( i = 0; ; ++i )
{
    v3 = strlen(s);
    if ( v3 <= i )
        break;
    v2 = ceil((double)((s[i] + v5[i]) / 2));
    if ( (long double)v6[i] != v2 )
        break;
}
v4 = strlen(s);
if ( v4 == i )
    return success_3();
else
    return failed();
}

```

```

17
18 from math import ceil
19
20 pwd = ''
21
22 i = 0
23 while True:
24     v3 = len(username)
25     if v3 <= i:
26         break
27     v22 = ceil(float((ord(username[i]) + ord(newStr[i])) // 2))
28     pwd += chr(v22)
29     i += 1
30
31 print("(+) Your generated password: " + pwd)

```

for example: username "399076724"
 so the pwd will be "48830QFBK"

```

---> Enter your username: 399076724
(+) New generated string: 6776glVRc
(+) Your generated password: 48830QFBK

```