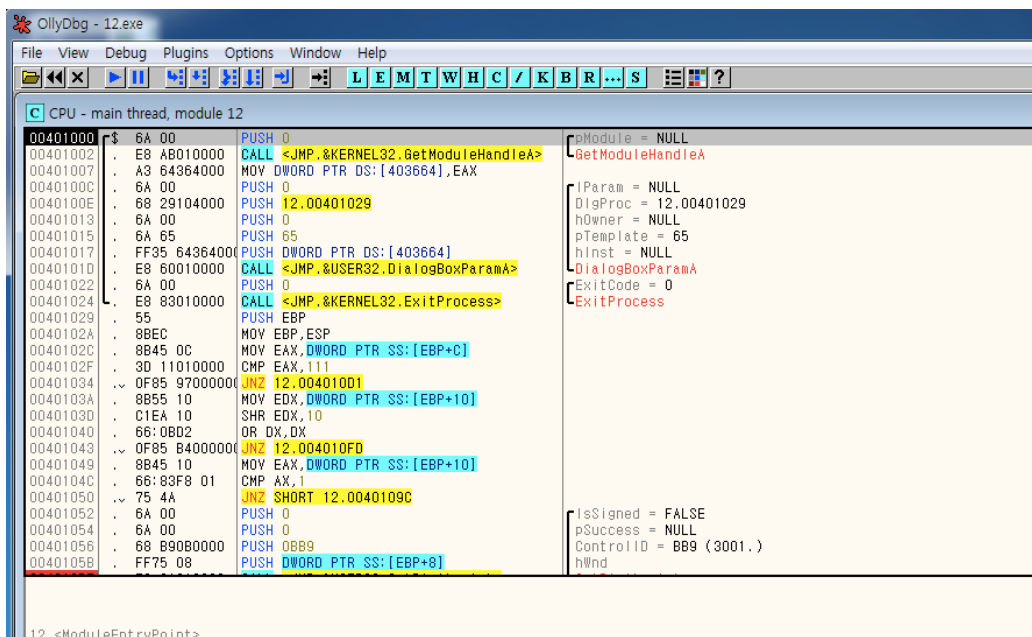
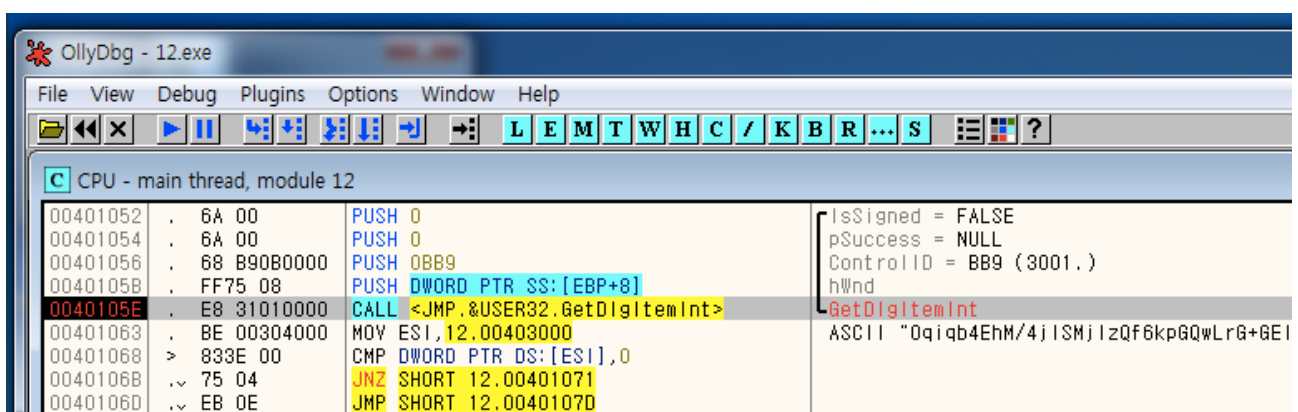


처음 실행한 화면입니다. 아무 값이나 입력하고 Check를 눌러도 반응이 없습니다.

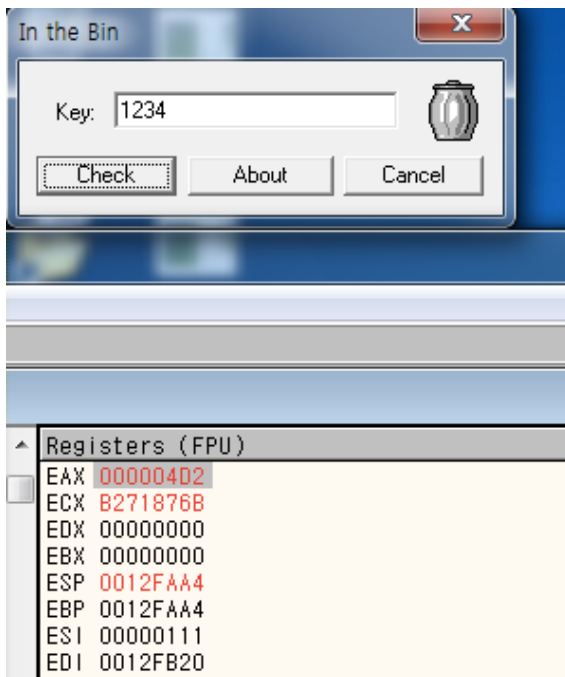


올리디버거로 내부를 분석해 보겠습니다.

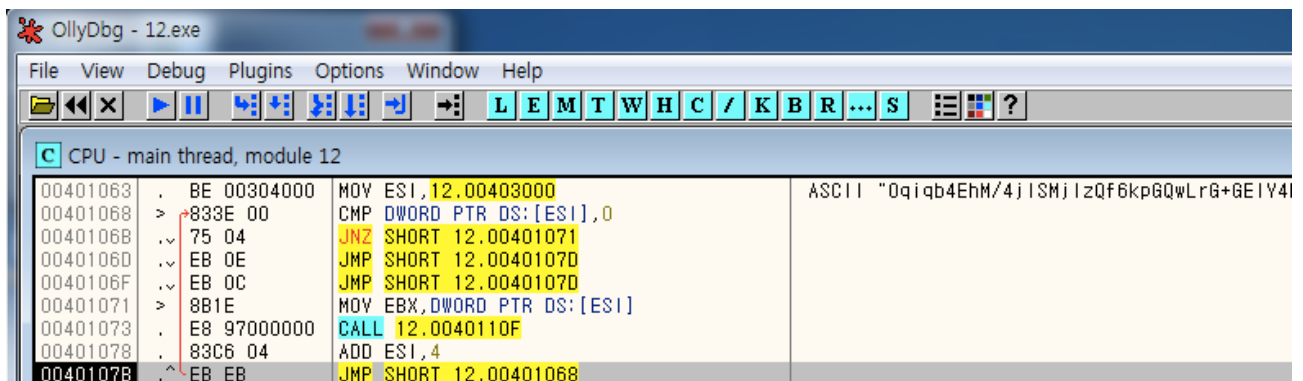


Check버튼을 누르면 GetDlgItemInt를 호출하는 것을 볼 수 있습니다.

이 함수는 입력된 값을 정수형으로 반환하는데 만약 숫자 이외의 다른 값이 입력되어 있다면 0을 반환합니다.



1234의 HEX값인 4D2가 함수를 호출한 리턴값으로 들어오는 것을 알 수 있습니다.



ESI에 엄청 긴 문자열을 집어넣고 루틴을 시작합니다.

00401068 주소를 보시면 'CMP DWORD PTR DS:[ESI], 0' 이라고 되어있습니다.

만일 ESI가 0이 되면 0040107D 주소로 점프하게 되고 아니면 00401071로 점프하게 됩니다.

00401071에서 EBX에 4바이트씩 ESI를 집어넣고 특정 함수를 지난 뒤 ESI에 4를 더합니다.

한번의 루틴에 ESI 4글자가 처리되는 것을 알 수 있습니다.

```

CPU - main thread, module 12
0040110F 51 PUSH ECX
00401110 52 PUSH EDX
00401111 8BD3 MOV EDX,EBX
00401113 8BC8 MOV ECX,EAX
00401115 40 INC EAX
00401116 F7D0 NOT EAX
00401118 43 INC EBX
00401119 F7D3 NOT EBX
0040111B 40 INC EAX
0040111C 43 INC EBX
0040111D 23C2 AND EAX,EDX
0040111F 23D9 AND EBX,ECX
00401121 03C3 ADD EAX,EBX
00401123 5A POP EDX
00401124 59 POP ECX
00401125 C3 RETN

```

루틴에서의 함수 내부입니다.

```

CPU - main thread, module 12
0040105E E8 31010000 CALL <JMP.<USER32.GetDlgItemInt>
00401063 BE 00304000 MOV ESI,12.00403000
00401068 833E 00 CMP DWORD PTR DS:[ESI],0
0040106B 75 04 JNZ SHORT 12.00401071
0040106D EB 0E JMP SHORT 12.00401070
0040106F EB 0C JMP SHORT 12.00401070
00401071 8B1E MOV EBX,DWORD PTR DS:[ESI]
00401073 E8 97000000 CALL 12.0040110F
00401078 83C6 04 ADD ESI,4
0040107B EB EB JMP SHORT 12.00401068
0040107D 3D BF96287A CMP EAX,7A2896BF

```

ESI에 저장된 특정 문자열과 입력값과의 연산을 통해 올바른 키값인지 계산하는 줄 알았는데

ESI의 문자열은 앞뒤로 대칭되는 문자열이고 위의 연산을 지나면 결국 처음 입력했던 값이 EAX에 나오게 됩니다.

```

CPU - main thread, module 12
0040107D 3D BF96287A CMP EAX,7A2896BF
00401082 75 14 JNZ SHORT 12.00401098
00401084 6A 40 PUSH 40
00401086 68 30354000 PUSH 12.00403530
00401088 68 38354000 PUSH 12.00403538
00401090 FF75 08 PUSH DWORD PTR SS:[EBP+8]
00401093 E8 02010000 CALL <JMP.<USER32.MessageBoxA>
00401098 EB 6C JMP SHORT 12.00401106

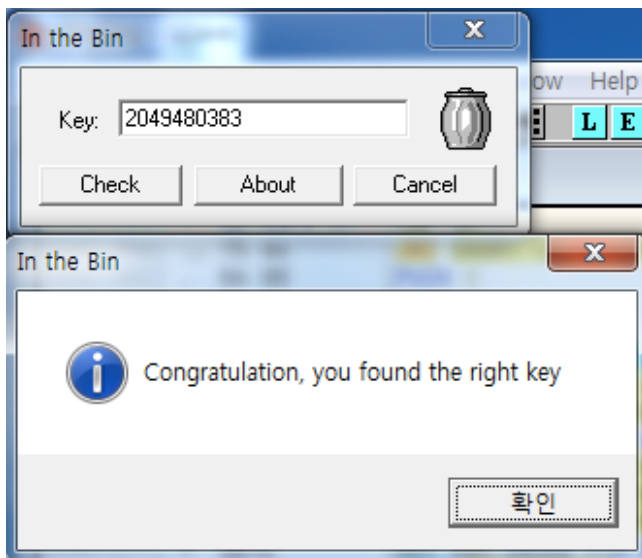
```

중간의 루틴은 디버깅을 방해하는 목적으로 의미없는 루틴을 넣은 것 같습니다.

처음 입력했던 값이 EAX에 들어오게 되고 그 값과 7A2896BF와 비교하게 됩니다.



그 값의 10진수가 이 프로그램의 키값입니다.

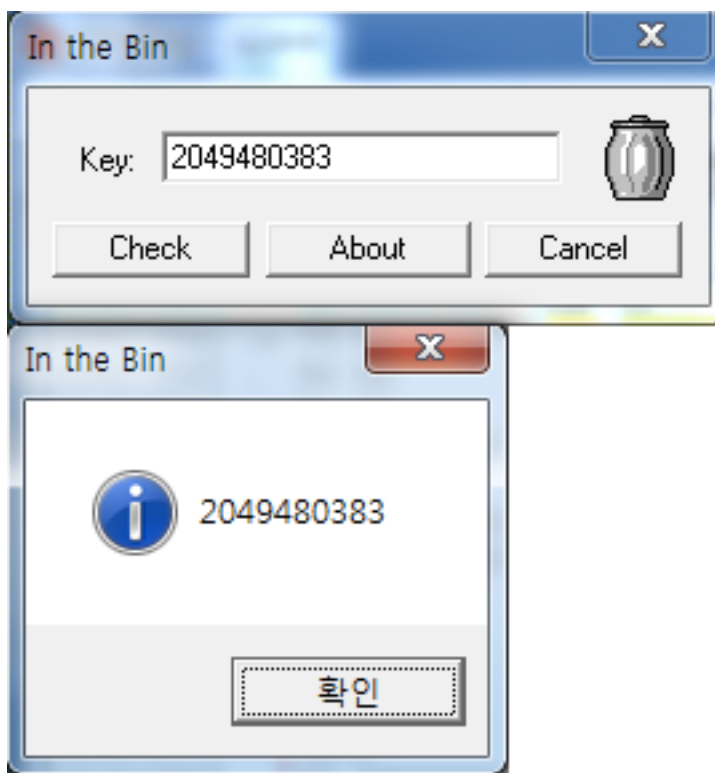


키값이 올바른 것을 알 수 있습니다.

이제 이 메세지 박스에 키값을 출력하도록 하겠습니다.

00000CF0	6D 65 33 67 65 6A 6D 62 42 66 4E 4C 4C 2F 6A 57	me3gejmbBfNLL/jW
00000D00	50 63 30 4A 49 59 34 62 47 2B 47 45 51 77 4C 72	Pc0JIY4bG+GEQwLr
00000D10	36 6B 70 47 6C 7A 51 66 49 53 4D 6A 4D 2F 34 6A	6kpGlzQfISMjM/4j
00000D20	62 34 45 68 4F 71 69 71 00 00 00 00 78 56 34 12	b4EhOqiq....xV4.
00000D30	49 6E 20 74 68 65 20 42 69 6E 00 43 6F 6E 67 72	In the Bin.Congr
00000D40	61 74 75 6C 61 74 69 6F 6E 2C 20 79 6F 75 20 66	atulation, you f
00000D50	6F 75 6E 64 20 74 68 65 20 72 69 67 68 74 20 6B	ound the right k
00000D60	65 79 00 00 00 00 00 00 00 00 00 00 00 00 00	ey.....
00000D70	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000D00	50 63 30 4A 49 59 34 62 47 2B 47 45 51 77 4C 72	Pc0JIY4bG+GEQwLr
00000D10	36 6B 70 47 6C 7A 51 66 49 53 4D 6A 4D 2F 34 6A	6kpGlzQfISMjM/4j
00000D20	62 34 45 68 4F 71 69 71 00 00 00 00 78 56 34 12	b4EhOqiq....xV4.
00000D30	49 6E 20 74 68 65 20 42 69 6E 00 32 30 34 39 34	In the Bin.20494
00000D40	38 30 33 38 33 00 00 00 00 00 00 00 00 00 00	80383.....

HEX Editor를 통해 메시지 박스의 문자열을 키값으로 바꿔준 뒤 저장합니다.



그 후 프로그램을 실행시켜 키값을 입력하면 위의 그림처럼 출력됩니다.