Reverse engineering Heroes 3 - Basics

By Raistlin, put together by Jakub Grzana
Last update: 17.02.2021

Prerequisites:

- IDA Pro (interactive disassembler by Hex-Rays) which is NOT free program!
- IDA Database for Heroes 3 SoD LINK
- Debugger OllyDbg LINK
- Russian SoD Executable LINK Password: GoogleDoesNotLikeExe

Introduction

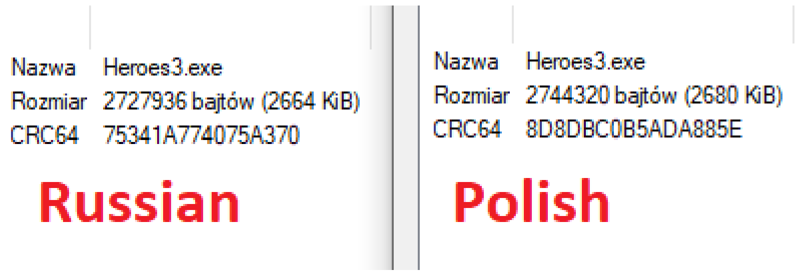
This tutorial will explain only basics of using IDA on example of changing values hardcoded in game memory. It is just first step for real reverse engineering – but important one.

IDA is a disassembler for computer software which generates assembly language source code from machine-executable code IDA performs automatic code analysis, using cross-references between code sections, knowledge of parameters of API calls, and other information. However, IDA itself will not bring you so much, you need the database, that was created and updated by developers during last 20 years.

To start, get a copy of IDA Pro, download database and debugger. After installation of IDA, you can open database HoMM3.idb just by double-clicking on it. IDA allows you to understand what is going on inside exec - it has named variables, functions etc.

Debugger can be used, among other things, to find precise addresses of constants you found in IDA. After installation, you must start instance of Debugger, then „open" executable of Heroes 3 SoD inside this debugger. Any SoD version should do, but i think WoG is based on russian version, so i encourage to use russian. *(i don't have any proofs for that tho)*

Here is proof that there're different versions of SoD exec – it is clearly visible in main menu, I don't know if the differences are meaningful though.



	Russian	Polish
Nazwa	Heroes3.exe	Heroes3.exe
Rozmiar	2727936 bajtów (2664 KiB)	2744320 bajtów (2680 KiB)
CRC64	75341A774075A370	8D8DBC0B5ADA885E

Basic assembler instructions

mov dest, source – copy value from source to dest
jmp – jumps to some place in code (pretty much like goto in C)
call name – call function with given name/address
push val – push value on stack
pop var – pop value from stack, save it in var

eax, ebx … - registers of processor. You can store values or pointers in tchem
mov eax, ebx - copies value stored in ebx to eax
mov eax, [ebx] - copies value from memory to eax (ebx is pointer)
Note: since Heroes 3 is 32bit executable, registers are 32bit=4bytes aswell.

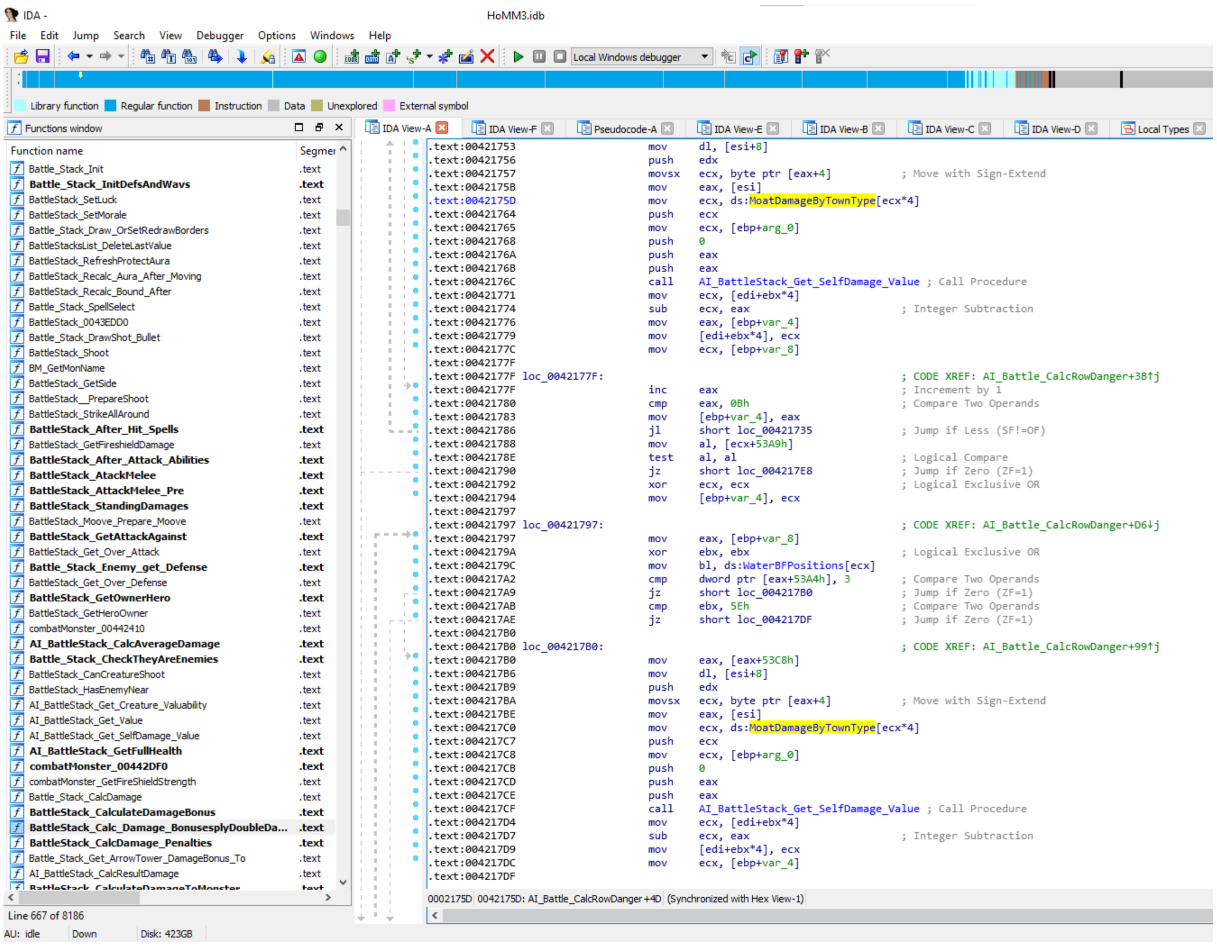
Usage of stack in functions:
(eax = 5, ebx = 8, esi = 0x446752)
push eax
push ebx
push esi
call 00678453
To get access to the arguments in the function use the registr esp. The first argument is [esp], the second - [esp+4], the third - [esp+8]

If you need more informations, there're many tutorials on the internet, for example this one.

Example 1 – Remove double moat from Fortress

Imagine, that you want to disable double moat in the Fortress, and you do not know which function sets it. The first thing you have to do is open your IDA and start searching. Press **alt+T** and enter the simplest word: Moat.

When you did it you did it and pressed "Ok", you see a random function, which does something with the moat



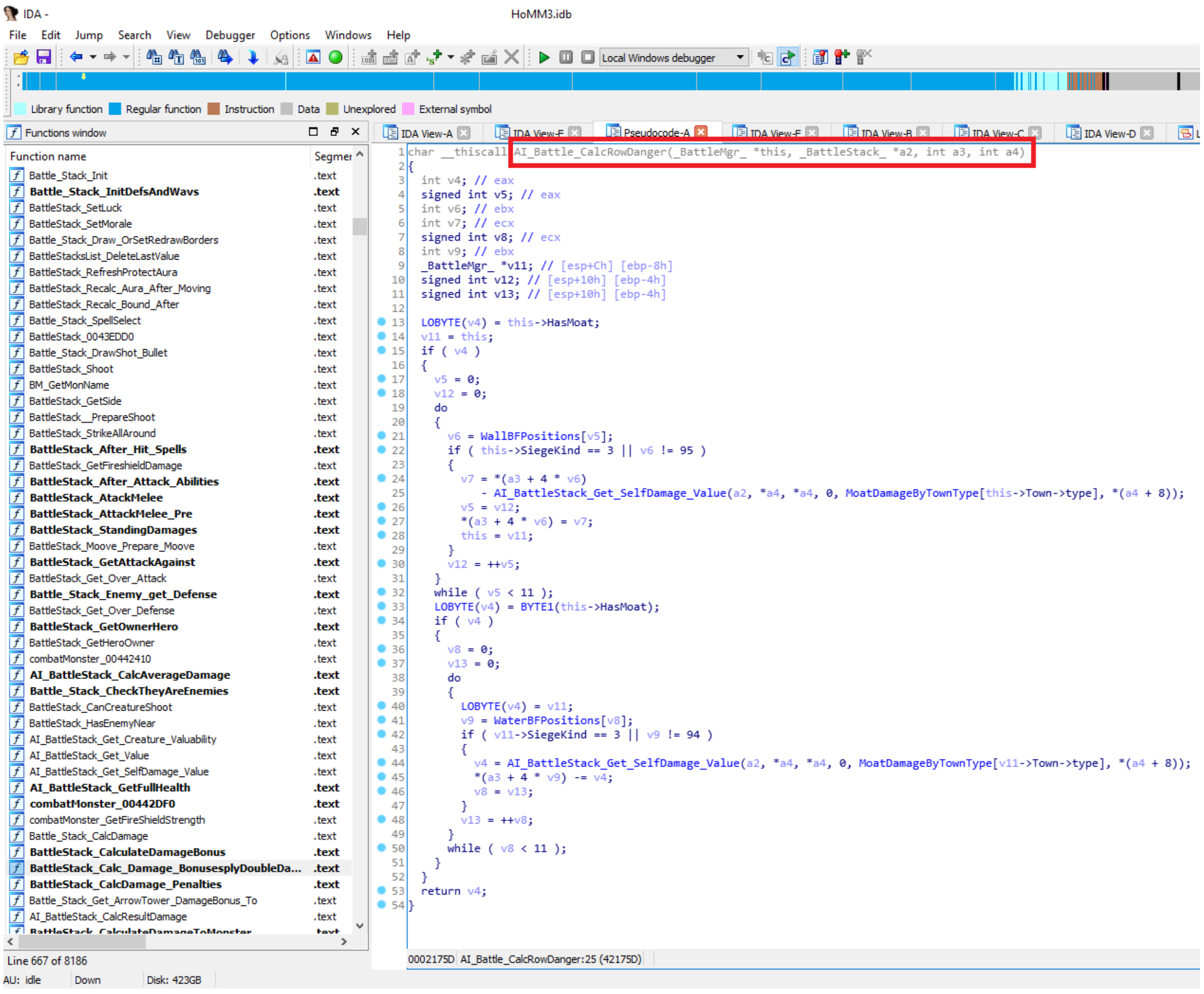
Function name	Segmer
Battle_Stack_Init	.text
Battle_Stack_InitDefsAndWavs	**.text**
BattleStack_SetLuck	.text
BattleStack_SetMorale	.text
Battle_Stack_Draw_OrSetRedrawBorders	.text
BattleStacksList_DeleteLastValue	.text
BattleStack_RefreshProtectAura	.text
BattleStack_Recalc_Aura_After_Moving	.text
BattleStack_Recalc_Bound_After	.text
Battle_Stack_SpellSelect	.text
BattleStack_0043EDD0	.text
Battle_Stack_DrawShot_Bullet	.text
BattleStack_Shoot	.text
BM_GetMonName	.text
BattleStack_GetSide	.text
BattleStack__PrepareShoot	.text
BattleStack_StrikeAllAround	.text
BattleStack_After_Hit_Spells	**.text**
BattleStack_GetFireshieldDamage	.text
BattleStack_After_Attack_Abilities	**.text**
BattleStack_AtackMelee	**.text**
BattleStack_AttackMelee_Pre	**.text**
BattleStack_StandingDamages	**.text**
BattleStack_Moove_Prepare_Moove	.text
BattleStack_GetAttackAgainst	**.text**
BattleStack_Get_Over_Attack	.text
Battle_Stack_Enemy_get_Defense	**.text**
BattleStack_Get_Over_Defense	.text
BattleStack_GetOwnerHero	**.text**
BattleStack_GetHeroOwner	.text
combatMonster_00442410	.text
AI_BattleStack_CalcAverageDamage	**.text**
Battle_Stack_CheckTheyAreEnemies	**.text**
BattleStack_CanCreatureShoot	.text
BattleStack_HasEnemyNear	.text
AI_BattleStack_Get_Creature_Valuability	.text
AI_BattleStack_Get_Value	.text
AI_BattleStack_Get_SelfDamage_Value	.text
AI_BattleStack_GetFullHealth	**.text**
combatMonster_00442DF0	**.text**
combatMonster_GetFireShieldStrength	.text
Battle_Stack_CalcDamage	.text
BattleStack_CalculateDamageBonus	**.text**
BattleStack_Calc_Damage_BonusesplyDoubleDa...	**.text**
BattleStack_CalcDamage_Penalties	**.text**
Battle_Stack_Get_ArrowTower_DamageBonus_To	.text
AI_BattleStack_CalcResultDamage	.text

Line 667 of 8186
AU: idle Down Disk: 423GB

```
.text:00421753                 mov     dl, [esi+8]
.text:00421756                 push    edx
.text:00421757                 movsx   ecx, byte ptr [eax+4]          ; Move with Sign-Extend
.text:0042175B                 mov     eax, [esi]
.text:0042175D                 mov     ecx, ds:MoatDamageByTownType[ecx*4]
.text:00421764                 push    ecx
.text:00421765                 mov     ecx, [ebp+arg_0]
.text:00421768                 push    0
.text:0042176A                 push    eax
.text:0042176B                 push    eax
.text:0042176C                 call    AI_BattleStack_Get_SelfDamage_Value ; Call Procedure
.text:00421771                 mov     ecx, [edi+ebx*4]
.text:00421774                 sub     ecx, eax                       ; Integer Subtraction
.text:00421776                 mov     eax, [ebp+var_4]
.text:00421779                 mov     [edi+ebx*4], ecx
.text:0042177C                 mov     ecx, [ebp+var_8]
.text:0042177F
.text:0042177F loc_0042177F:                                          ; CODE XREF: AI_Battle_CalcRowDanger+3B↑j
.text:0042177F                 inc     eax                            ; Increment by 1
.text:00421780                 cmp     eax, 0Bh                       ; Compare Two Operands
.text:00421783                 mov     [ebp+var_4], eax
.text:00421786                 jl      short loc_00421735             ; Jump if Less (SF!=OF)
.text:00421788                 mov     al, [ecx+53A9h]
.text:0042178E                 test    al, al                         ; Logical Compare
.text:00421790                 jz      short loc_004217E8             ; Jump if Zero (ZF=1)
.text:00421792                 xor     ecx, ecx                       ; Logical Exclusive OR
.text:00421794                 mov     [ebp+var_4], ecx
.text:00421797
.text:00421797 loc_00421797:                                          ; CODE XREF: AI_Battle_CalcRowDanger+D6↓j
.text:00421797                 mov     eax, [ebp+var_8]
.text:0042179A                 xor     ebx, ebx                       ; Logical Exclusive OR
.text:0042179C                 mov     bl, ds:WaterBFPositions[ecx]
.text:004217A2                 cmp     dword ptr [eax+53A4h], 3       ; Compare Two Operands
.text:004217A9                 jz      short loc_004217B0             ; Jump if Zero (ZF=1)
.text:004217AB                 cmp     ebx, 5Eh                       ; Compare Two Operands
.text:004217AE                 jz      short loc_004217DF             ; Jump if Zero (ZF=1)
.text:004217B0
.text:004217B0 loc_004217B0:                                          ; CODE XREF: AI_Battle_CalcRowDanger+99↑j
.text:004217B0                 mov     eax, [eax+53C8h]
.text:004217B6                 mov     dl, [esi+8]
.text:004217B9                 push    edx
.text:004217BA                 movsx   ecx, byte ptr [eax+4]          ; Move with Sign-Extend
.text:004217BE                 mov     eax, [esi]
.text:004217C0                 mov     ecx, ds:MoatDamageByTownType[ecx*4]
.text:004217C7                 push    ecx
.text:004217C8                 mov     ecx, [ebp+arg_0]
.text:004217CB                 push    0
.text:004217CD                 push    eax
.text:004217CE                 push    eax
.text:004217CF                 call    AI_BattleStack_Get_SelfDamage_Value ; Call Procedure
.text:004217D4                 mov     ecx, [edi+ebx*4]
.text:004217D7                 sub     ecx, eax                       ; Integer Subtraction
.text:004217D9                 mov     [edi+ebx*4], ecx
.text:004217DC                 mov     ecx, [ebp+var_4]
.text:004217DF
```

0002175D 0042175D: AI_Battle_CalcRowDanger+4D (Synchronized with Hex View-1)

Use **Tab** key to get into pseudocode, representing what's going on here – you will see that it's something you don't need.



IDA - HoMM3.idb

Function name	Segment
Battle_Stack_Init	.text
Battle_Stack_InitDefsAndWavs	**.text**
BattleStack_SetLuck	.text
BattleStack_SetMorale	.text
Battle_Stack_Draw_OrSetRedrawBorders	.text
BattleStacksList_DeleteLastValue	.text
BattleStack_RefreshProtectAura	.text
BattleStack_Recalc_Aura_After_Moving	.text
BattleStack_Recalc_Bound_After	.text
Battle_Stack_SpellSelect	.text
BattleStack_0043EDD0	.text
Battle_Stack_DrawShot_Bullet	.text
BattleStack_Shoot	.text
BM_GetMonName	.text
BattleStack_GetSide	.text
BattleStack__PrepareShoot	.text
BattleStack_StrikeAllAround	.text
BattleStack_After_Hit_Spells	**.text**
BattleStack_GetFireshieldDamage	.text
BattleStack_After_Attack_Abilities	**.text**
BattleStack_AtackMelee	**.text**
BattleStack_AttackMelee_Pre	**.text**
BattleStack_StandingDamages	**.text**
BattleStack_Moove_Prepare_Moove	.text
BattleStack_GetAttackAgainst	**.text**
BattleStack_Get_Over_Attack	.text
Battle_Stack_Enemy_get_Defense	**.text**
BattleStack_Get_Over_Defense	.text
BattleStack_GetOwnerHero	**.text**
BattleStack_GetHeroOwner	.text
combatMonster_00442410	.text
AI_BattleStack_CalcAverageDamage	**.text**
Battle_Stack_CheckTheyAreEnemies	**.text**
BattleStack_CanCreatureShoot	.text
BattleStack_HasEnemyNear	.text
AI_BattleStack_Get_Creature_Valuability	.text
AI_BattleStack_Get_Value	.text
AI_BattleStack_Get_SelfDamage_Value	.text
AI_BattleStack_GetFullHealth	**.text**
combatMonster_00442DF0	**.text**
combatMonster_GetFireShieldStrength	.text
Battle_Stack_CalcDamage	.text
BattleStack_CalculateDamageBonus	**.text**
BattleStack_Calc_Damage_BonusesplyDoubleDa...	**.text**
BattleStack_CalcDamage_Penalties	**.text**
Battle_Stack_Get_ArrowTower_DamageBonus_To	.text
AI_BattleStack_CalcResultDamage	.text
BattleStack_CalculateDamageToMonster	**.text**

Line 667 of 8186

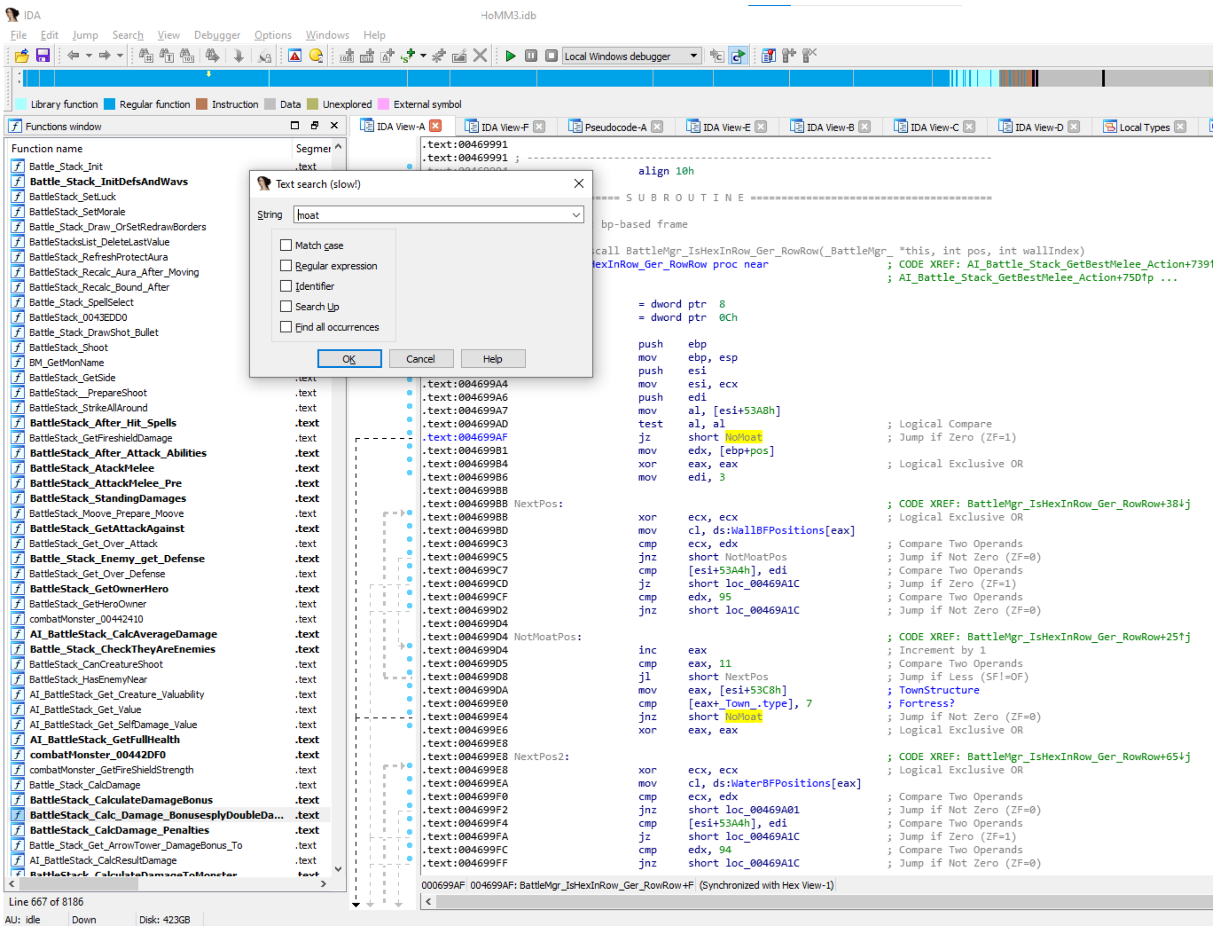
AU: idle Down Disk: 423GB

```
char __thiscall AI_Battle_CalcRowDanger(_BattleMgr_ *this, _BattleStack_ *a2, int a3, int a4)
{
  int v4; // eax
  signed int v5; // eax
  int v6; // ebx
  int v7; // ecx
  signed int v8; // ecx
  int v9; // ebx
  _BattleMgr_ *v11; // [esp+Ch] [ebp-8h]
  signed int v12; // [esp+10h] [ebp-4h]
  signed int v13; // [esp+10h] [ebp-4h]

  LOBYTE(v4) = this->HasMoat;
  v11 = this;
  if ( v4 )
  {
    v5 = 0;
    v12 = 0;
    do
    {
      v6 = WallBFPositions[v5];
      if ( this->SiegeKind == 3 || v6 != 95 )
      {
        v7 = *(a3 + 4 * v6)
           - AI_BattleStack_Get_SelfDamage_Value(a2, *a4, *a4, 0, MoatDamageByTownType[this->Town->type], *(a4 + 8));
        v5 = v12;
        *(a3 + 4 * v6) = v7;
        this = v11;
      }
      v12 = ++v5;
    }
    while ( v5 < 11 );
    LOBYTE(v4) = BYTE1(this->HasMoat);
    if ( v4 )
    {
      v8 = 0;
      v13 = 0;
      do
      {
        LOBYTE(v4) = v11;
        v9 = WaterBFPositions[v8];
        if ( v11->SiegeKind == 3 || v9 != 94 )
        {
          v4 = AI_BattleStack_Get_SelfDamage_Value(a2, *a4, *a4, 0, MoatDamageByTownType[v11->Town->type], *(a4 + 8));
          *(a3 + 4 * v9) -= v4;
          v8 = v13;
        }
        v13 = ++v8;
      }
      while ( v8 < 11 );
    }
  }
  return v4;
}
```

0002175D AI_Battle_CalcRowDanger:25 (42175D)

File Edit Jump Search View Debugger Options Windows Help
Local Windows debugger
Library function Regular function Instruction Data Unexplored External symbol
Functions window
IDA View-A IDA View-F Pseudocode-A IDA View-E IDA View-B IDA View-C IDA View-D

Use **Tab** again to go back into IDA view. After a few attempts more, you will find something more interesting



Text search (slow!)

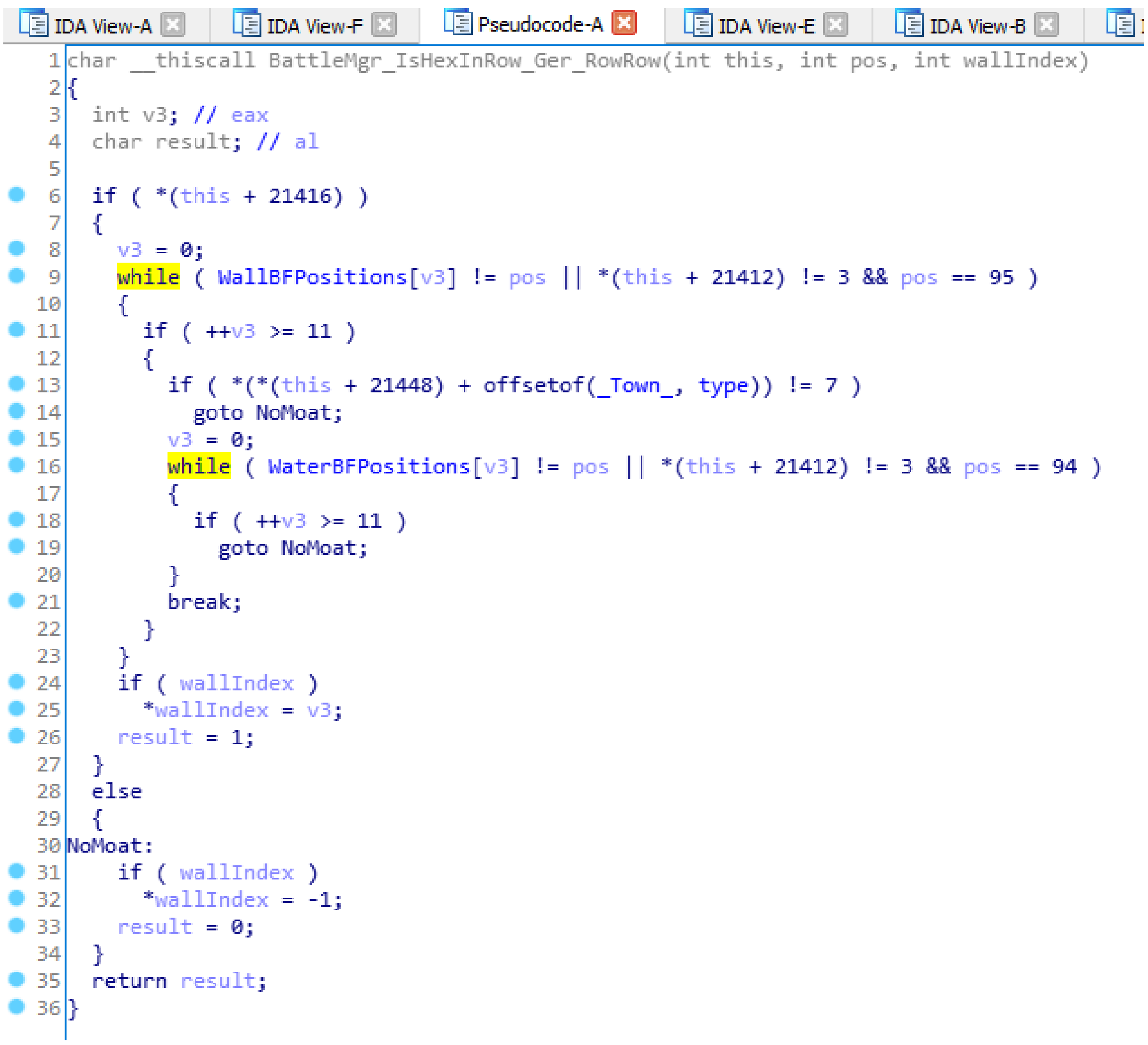
String: moat

☐ Match case
☐ Regular expression
☐ Identifier
☐ Search Up
☐ Find all occurrences

```
Function name                                  Segment
Battle_Stack_Init                              .text
Battle_Stack_InitDefsAndWavs
BattleStack_SetLuck
BattleStack_SetMorale
Battle_Stack_Draw_OrSetRedrawBorders
BattleStacksList_DeleteLastValue
BattleStack_RefreshProtectAura
BattleStack_Recalc_Aura_After_Moving
BattleStack_Recalc_Bound_After
Battle_Stack_SpellSelect
BattleStack_0043EDD0
Battle_Stack_DrawShot_Bullet
BattleStack_Shoot
BM_GetMonName
BattleStack_GetSide                            .text
BattleStack__PrepareShoot                      .text
BattleStack_StrikeAllAround                    .text
BattleStack_After_Hit_Spells                   .text
BattleStack_GetFireshieldDamage                .text
BattleStack_After_Attack_Abilities             .text
BattleStack_AtackMelee                         .text
BattleStack_AttackMelee_Pre                    .text
BattleStack_StandingDamages                    .text
BattleStack_Moove_Prepare_Moove                .text
BattleStack_GetAttackAgainst                   .text
BattleStack_Get_Over_Attack                    .text
Battle_Stack_Enemy_get_Defense                 .text
BattleStack_Get_Over_Defense                   .text
BattleStack_GetOwnerHero                       .text
BattleStack_GetHeroOwner                       .text
combatMonster_00442410                         .text
AI_BattleStack_CalcAverageDamage               .text
Battle_Stack_CheckTheyAreEnemies               .text
BattleStack_CanCreatureShoot                   .text
BattleStack_HasEnemyNear                       .text
AI_BattleStack_Get_Creature_Valuability        .text
AI_BattleStack_Get_Value                       .text
AI_BattleStack_Get_SelfDamage_Value            .text
AI_BattleStack_GetFullHealth                   .text
combatMonster_00442DF0                         .text
combatMonster_GetFireShieldStrength            .text
Battle_Stack_CalcDamage                        .text
BattleStack_CalculateDamageBonus               .text
BattleStack_Calc_Damage_BonusesplyDoubleDa...  .text
BattleStack_CalcDamage_Penalties               .text
Battle_Stack_Get_ArrowTower_DamageBonus_To     .text
AI_BattleStack_CalcResultDamage                .text
BattleStack_CalculateDamageToMonster           .text

Line 667 of 8186
AU: idle    Down    Disk: 423GB
```

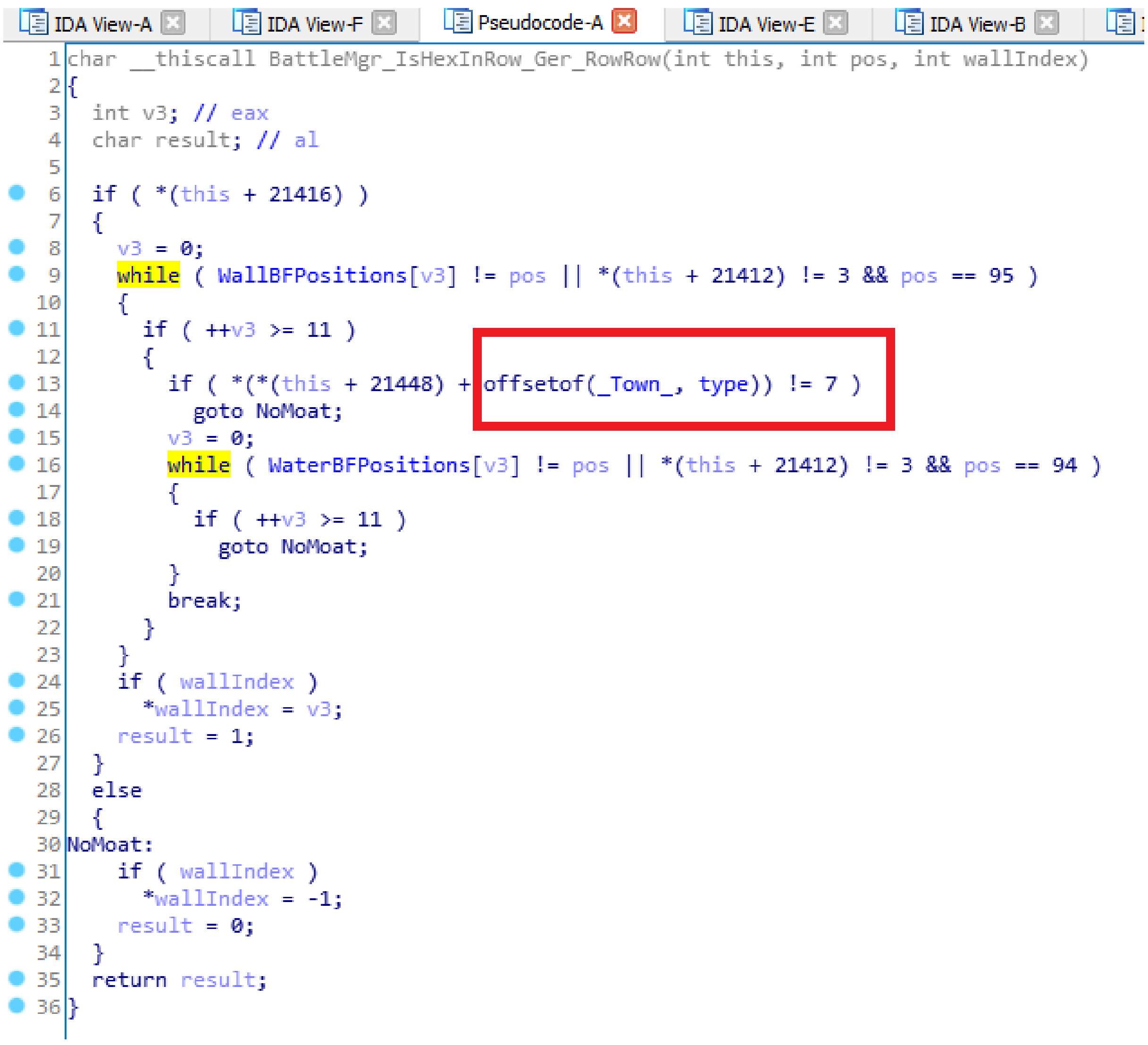
```
.text:00469991
.text:00469991 ; ---------------------------------------------------------------------------
                                align 10h

                ==== S U B R O U T I N E =======================================

                ; bp-based frame

                ;call BattleMgr_IsHexInRow_Ger_RowRow(_BattleMgr_ *this, int pos, int wallIndex)
                HexInRow_Ger_RowRow proc near            ; CODE XREF: AI_Battle_Stack_GetBestMelee_Action+739↑
                                                         ; AI_Battle_Stack_GetBestMelee_Action+75D↑p ...

                                = dword ptr  8
                                = dword ptr  0Ch

                                push    ebp
                                mov     ebp, esp
                                push    esi
.text:004699A4                  mov     esi, ecx
.text:004699A6                  push    edi
.text:004699A7                  mov     al, [esi+53A8h]
.text:004699AD                  test    al, al                   ; Logical Compare
.text:004699AF                  jz      short NoMoat             ; Jump if Zero (ZF=1)
.text:004699B1                  mov     edx, [ebp+pos]
.text:004699B4                  xor     eax, eax                 ; Logical Exclusive OR
.text:004699B6                  mov     edi, 3
.text:004699BB
.text:004699BB NextPos:                                          ; CODE XREF: BattleMgr_IsHexInRow_Ger_RowRow+38↓j
.text:004699BB                  xor     ecx, ecx                 ; Logical Exclusive OR
.text:004699BD                  mov     cl, ds:WallBFPositions[eax]
.text:004699C3                  cmp     ecx, edx                 ; Compare Two Operands
.text:004699C5                  jnz     short NotMoatPos         ; Jump if Not Zero (ZF=0)
.text:004699C7                  cmp     [esi+53A4h], edi         ; Compare Two Operands
.text:004699CD                  jz      short loc_00469A1C       ; Jump if Zero (ZF=1)
.text:004699CF                  cmp     edx, 95                  ; Compare Two Operands
.text:004699D2                  jnz     short loc_00469A1C       ; Jump if Not Zero (ZF=0)
.text:004699D4
.text:004699D4 NotMoatPos:                                       ; CODE XREF: BattleMgr_IsHexInRow_Ger_RowRow+25↑j
.text:004699D4                  inc     eax                      ; Increment by 1
.text:004699D5                  cmp     eax, 11                  ; Compare Two Operands
.text:004699D8                  jl      short NextPos            ; Jump if Less (SF!=OF)
.text:004699DA                  mov     eax, [esi+53C8h]         ; TownStructure
.text:004699E0                  cmp     [eax+_Town_.type], 7     ; Fortress?
.text:004699E4                  jnz     short NoMoat             ; Jump if Not Zero (ZF=0)
.text:004699E6                  xor     eax, eax                 ; Logical Exclusive OR
.text:004699E8
.text:004699E8 NextPos2:                                         ; CODE XREF: BattleMgr_IsHexInRow_Ger_RowRow+65↓j
.text:004699E8                  xor     ecx, ecx                 ; Logical Exclusive OR
.text:004699EA                  mov     cl, ds:WaterBFPositions[eax]
.text:004699F0                  cmp     ecx, edx                 ; Compare Two Operands
.text:004699F2                  jnz     short loc_00469A01       ; Jump if Not Zero (ZF=0)
.text:004699F4                  cmp     [esi+53A4h], edi         ; Compare Two Operands
.text:004699FA                  jz      short loc_00469A1C       ; Jump if Zero (ZF=1)
.text:004699FC                  cmp     edx, 94                  ; Compare Two Operands
.text:004699FF                  jnz     short loc_00469A1C       ; Jump if Not Zero (ZF=0)

000699AF 004699AF: BattleMgr_IsHexInRow_Ger_RowRow+F (Synchronized with Hex View-1)
```

Press **Tab** again and – bingo! This is the right function



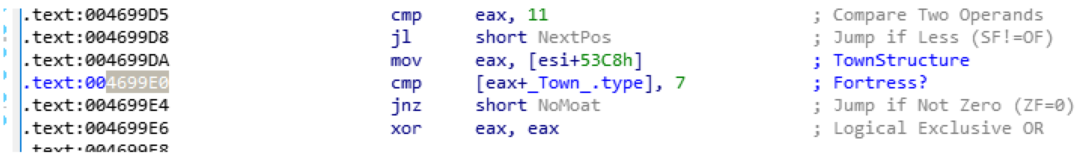
```
char __thiscall BattleMgr_IsHexInRow_Ger_RowRow(int this, int pos, int wallIndex)
{
  int v3; // eax
  char result; // al

  if ( *(this + 21416) )
  {
    v3 = 0;
    while ( WallBFPositions[v3] != pos || *(this + 21412) != 3 && pos == 95 )
    {
      if ( ++v3 >= 11 )
      {
        if ( *(*(this + 21448) + offsetof(_Town_, type)) != 7 )
          goto NoMoat;
        v3 = 0;
        while ( WaterBFPositions[v3] != pos || *(this + 21412) != 3 && pos == 94 )
        {
          if ( ++v3 >= 11 )
            goto NoMoat;
        }
        break;
      }
    }
    if ( wallIndex )
      *wallIndex = v3;
    result = 1;
  }
  else
  {
NoMoat:
    if ( wallIndex )
      *wallIndex = -1;
    result = 0;
  }
  return result;
}
```

Here is the check



```
char __thiscall BattleMgr_IsHexInRow_Ger_RowRow(int this, int pos, int wallIndex)
{
  int v3; // eax
  char result; // al

  if ( *(this + 21416) )
  {
    v3 = 0;
    while ( WallBFPositions[v3] != pos || *(this + 21412) != 3 && pos == 95 )
    {
      if ( ++v3 >= 11 )
      {
        if ( *(*(this + 21448) + offsetof(_Town_, type)) != 7 )
          goto NoMoat;
        v3 = 0;
        while ( WaterBFPositions[v3] != pos || *(this + 21412) != 3 && pos == 94 )
        {
          if ( ++v3 >= 11 )
            goto NoMoat;
        }
        break;
      }
    }
    if ( wallIndex )
      *wallIndex = v3;
    result = 1;
  }
  else
  {
NoMoat:
    if ( wallIndex )
      *wallIndex = -1;
    result = 0;
  }
  return result;
}
```

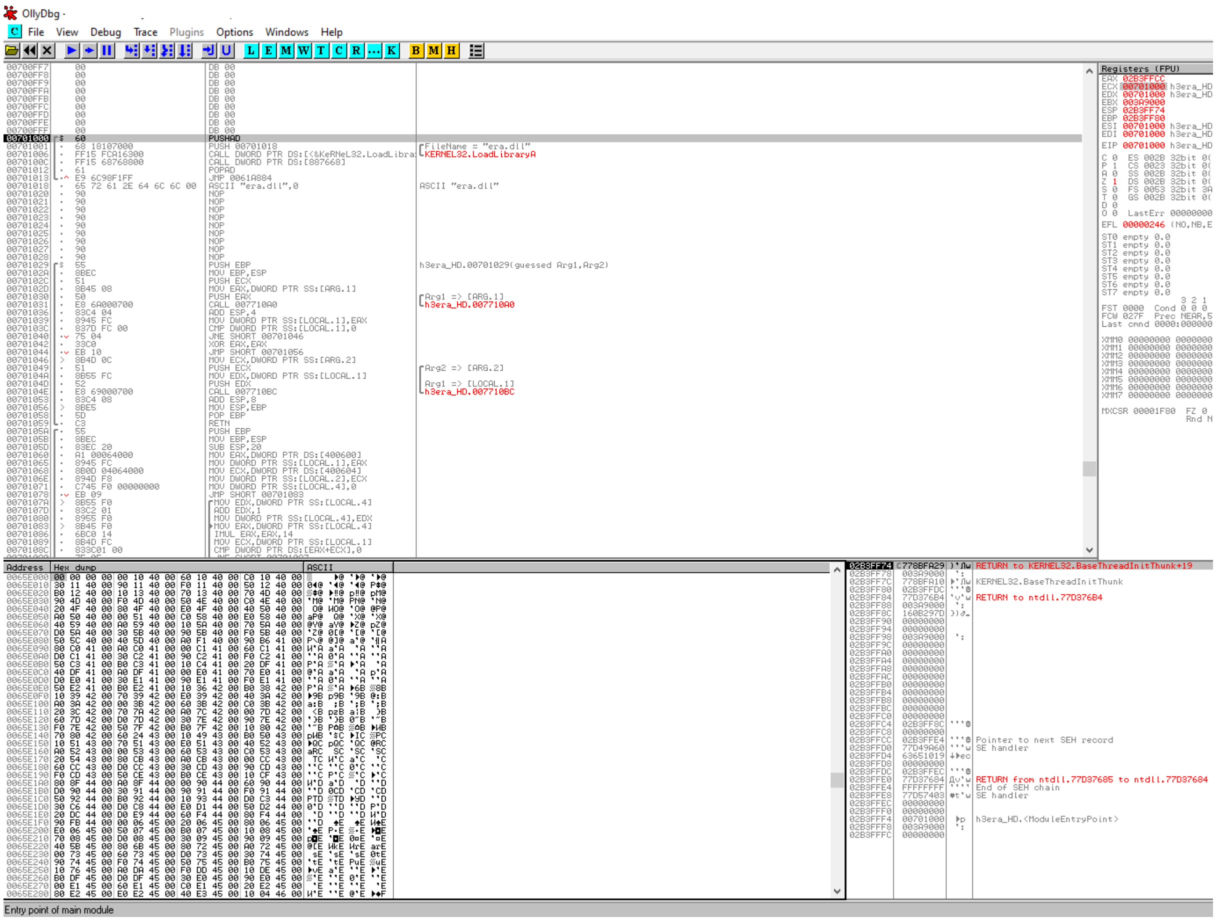
Mouse-click on it, then press **Tab** once again. You will get this:



```
.text:004699D5                  cmp     eax, 11                          ; Compare Two Operands
.text:004699D8                  jl      short NextPos                    ; Jump if Less (SF!=OF)
.text:004699DA                  mov     eax, [esi+53C8h]                 ; TownStructure
.text:004699E0                  cmp     [eax+_Town_.type], 7             ; Fortress?
.text:004699E4                  jnz     short NoMoat                     ; Jump if Not Zero (ZF=0)
.text:004699E6                  xor     eax, eax                         ; Logical Exclusive OR
.text:004699E8
```

Select and copy address of comparison (cmp asm instruction)

We’re done in IDA at this point. Open (olly) debugger->File->Open->Select SoD executable. Click on hex dump frame (it is at the bottom of the window) *(Note: here h3era HD.exe is used – doesn’t change much, but don’t do it if you work on 3.59, only SoD then)*



OllyDbg -
File View Debug Trace Plugins Options Windows Help
L E M W T C R ... K B M H

```
00700FF7     00                       DB 00
00700FF8     00                       DB 00
00700FF9     00                       DB 00
00700FFA     00                       DB 00
00700FFB     00                       DB 00
00700FFC     00                       DB 00
00700FFD     00                       DB 00
00700FFE     00                       DB 00
00700FFF     00                       DB 00
00701000  $  60                       PUSHAD
00701001  .  68 18107000              PUSH 00701018                              FileName = "era.dll"
00701006  .  FF15 FCA16300            CALL DWORD PTR DS:[<&KeRNeL32.LoadLibra    KERNEL32.LoadLibraryA
0070100C  .  FF15 68768800            CALL DWORD PTR DS:[887668]
00701012  .  61                       POPAD
00701013  .^ E9 6C98F1FF              JMP 0061A884
00701018  .  65 72 61 2E 64 6C 6C 00  ASCII "era.dll",0                      ASCII "era.dll"
00701020  .  90                       NOP
00701021  .  90                       NOP
00701022  .  90                       NOP
00701023  .  90                       NOP
00701024  .  90                       NOP
00701025  .  90                       NOP
00701026  .  90                       NOP
00701027  .  90                       NOP
00701028  .  90                       NOP
00701029  $  55                       PUSH EBP                                   h3era_HD.00701029(guessed Arg1,Arg2)
0070102A  .  8BEC                     MOV EBP,ESP
0070102C  .  51                       PUSH ECX
0070102D  .  8B45 08                  MOV EAX,DWORD PTR SS:[ARG.1]
00701030  .  50                       PUSH EAX                                   Arg1 => [ARG.1]
00701031  .  E8 6A000700              CALL 007710A0                              h3era_HD.007710A0
00701036  .  83C4 04                  ADD ESP,4
00701039  .  8945 FC                  MOV DWORD PTR SS:[LOCAL.1],EAX
0070103C  .  837D FC 00               CMP DWORD PTR SS:[LOCAL.1],0
00701040  .v 75 04                    JNE SHORT 00701046
00701042  .  33C0                     XOR EAX,EAX
00701044  .v EB 10                    JMP SHORT 00701056
00701046  >  8B4D 0C                  MOV ECX,DWORD PTR SS:[ARG.2]
00701049  .  51                       PUSH ECX                                   Arg2 => [ARG.2]
0070104A  .  8B55 FC                  MOV EDX,DWORD PTR SS:[LOCAL.1]
0070104D  .  52                       PUSH EDX                                   Arg1 => [LOCAL.1]
0070104E  .  E8 69000700              CALL 007710BC                              h3era_HD.007710BC
00701053  .  83C4 08                  ADD ESP,8
00701056  >  8BE5                     MOV ESP,EBP
00701058  .  5D                       POP EBP
00701059  .  C3                       RETN
0070105A  .  55                       PUSH EBP
0070105B  .  8BEC                     MOV EBP,ESP
0070105D  .  83EC 20                  SUB ESP,20
00701060  .  A1 00064000              MOV EAX,DWORD PTR DS:[400600]
00701065  .  8945 FC                  MOV DWORD PTR SS:[LOCAL.1],EAX
00701068  .  8B0D 04064000            MOV ECX,DWORD PTR DS:[400604]
0070106E  .  894D F8                  MOV DWORD PTR SS:[LOCAL.2],ECX
00701071  .  C745 F0 00000000         MOV DWORD PTR SS:[LOCAL.4],0
00701078  .v EB 09                    JMP SHORT 00701083
0070107A  >  8B55 F0                  MOV EDX,DWORD PTR SS:[LOCAL.4]
0070107D  .  83C2 01                  ADD EDX,1
00701080  .  8955 F0                  MOV DWORD PTR SS:[LOCAL.4],EDX
00701083  >  8B45 F0                  MOV EAX,DWORD PTR SS:[LOCAL.4]
00701086  .  6BC0 14                  IMUL EAX,EAX,14
00701089  .  8B4D FC                  MOV ECX,DWORD PTR SS:[LOCAL.1]
0070108C  .  833C01 00                CMP DWORD PTR DS:[EAX+ECX],0
```

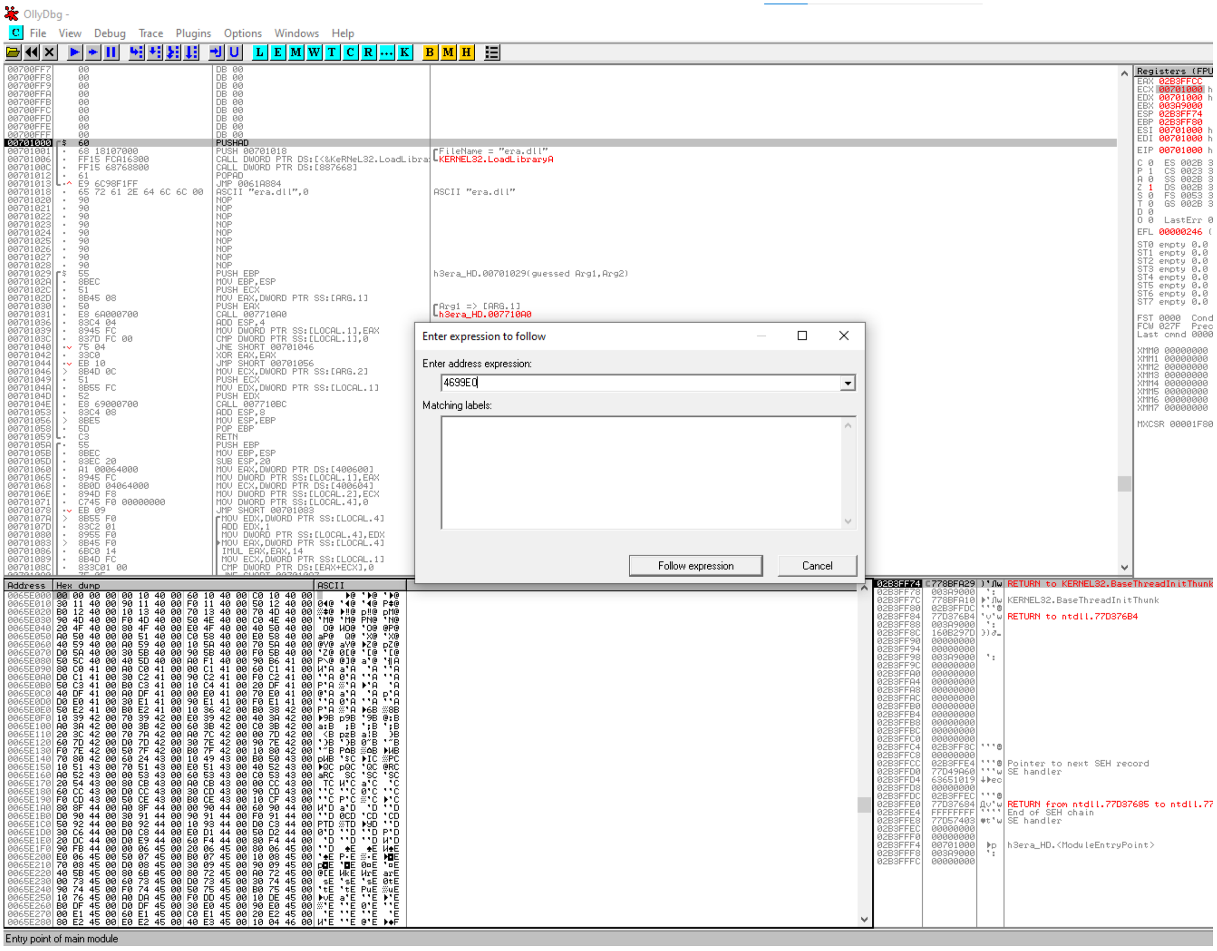
```
Registers (FPU)
EAX 02B3FFCC
ECX 00701000 h3era_HD
EDX 00701000 h3era_HD
EBX 003A9000
ESP 02B3FF74
EBP 02B3FF80
ESI 00701000 h3era_HD
EDI 00701000 h3era_HD

EIP 00701000 h3era_HD

C 0  ES 002B 32bit 0(
P 1  CS 0023 32bit 0(
A 0  SS 002B 32bit 0(
Z 1  DS 002B 32bit 0(
S 0  FS 0053 32bit 3A
T 0  GS 002B 32bit 0(
D 0
O 0  LastErr 00000000

EFL 00000246 (NO,NB,E

ST0 empty 0.0
ST1 empty 0.0
ST2 empty 0.0
ST3 empty 0.0
ST4 empty 0.0
ST5 empty 0.0
ST6 empty 0.0
ST7 empty 0.0
               3 2 1
FST 0000  Cond 0 0 0
FCW 027F  Prec NEAR,5
Last cmnd 0000:000000

XMM0 00000000 0000000
XMM1 00000000 0000000
XMM2 00000000 0000000
XMM3 00000000 0000000
XMM4 00000000 0000000
XMM5 00000000 0000000
XMM6 00000000 0000000
XMM7 00000000 0000000

MXCSR 00001F80  FZ 0
                Rnd N
```

```
Address   Hex dump                                          ASCII
0065E000  00 00 00 00 00 10 40 00 60 10 40 00 C0 10 40 00      ▶@ '▶@ '▶@
0065E010  30 11 40 00 90 11 40 00 F0 11 40 00 50 12 40 00  0◀@ '◀@ '◀@ P‡@
0065E020  B0 12 40 00 10 13 40 00 70 13 40 00 70 4D 40 00  ░‡@ ▶‼@ p‼@ pM@
0065E030  90 4D 40 00 F0 4D 40 00 50 4E 40 00 C0 4E 40 00  'M@ 'M@ PN@ 'N@
0065E040  20 4F 40 00 80 4F 40 00 E0 4F 40 00 40 50 40 00   O@ ИO@ 'O@ @P@
0065E050  A0 50 40 00 00 51 40 00 C0 58 40 00 E0 58 40 00  aP@  Q@ 'X@ 'X@
0065E060  40 59 40 00 A0 59 40 00 10 5A 40 00 70 5A 40 00  @Y@ aY@ ▶Z@ pZ@
0065E070  D0 5A 40 00 30 5B 40 00 90 5B 40 00 F0 5B 40 00  'Z@ 0[@ '[@ '[@
0065E080  50 5C 40 00 40 5D 40 00 A0 F1 40 00 90 B6 41 00  P\@ @]@ a'@ '╢A
0065E090  80 C0 41 00 A0 C0 41 00 00 C1 41 00 60 C1 41 00  И'A a'A  'A ''A
0065E0A0  D0 C1 41 00 30 C2 41 00 90 C2 41 00 F0 C2 41 00  ''A 0'A ''A ''A
0065E0B0  50 C3 41 00 B0 C3 41 00 10 C4 41 00 20 DF 41 00  P'A ░'A ▶'A  'A
0065E0C0  40 DF 41 00 A0 DF 41 00 00 E0 41 00 70 E0 41 00  @'A a'A  'A p'A
0065E0D0  D0 E0 41 00 30 E1 41 00 90 E1 41 00 F0 E1 41 00  ''A 0'A ''A ''A
0065E0E0  50 E2 41 00 B0 E2 41 00 10 36 42 00 B0 38 42 00  P'A ░'A ▶6B ░8B
0065E0F0  10 39 42 00 70 39 42 00 E0 39 42 00 40 3A 42 00  ▶9B p9B '9B @:B
0065E100  A0 3A 42 00 00 3B 42 00 60 3B 42 00 C0 3B 42 00  a:B  ;B ';B ';B
0065E110  20 3C 42 00 70 7A 42 00 A0 7C 42 00 00 7D 42 00   <B pzB a¦B  }B
0065E120  60 7D 42 00 D0 7D 42 00 30 7E 42 00 90 7E 42 00  '}B '}B 0~B '~B
0065E130  F0 7E 42 00 50 7F 42 00 B0 7F 42 00 10 80 42 00  '~B P⌂B ░⌂B ▶ИB
0065E140  70 80 42 00 60 24 43 00 10 49 43 00 B0 50 43 00  pИB '$C ▶IC ░PC
0065E150  10 51 43 00 70 51 43 00 E0 51 43 00 40 52 43 00  ▶QC pQC 'QC @RC
0065E160  A0 52 43 00 00 53 43 00 60 53 43 00 C0 53 43 00  aRC  SC 'SC 'SC
0065E170  20 54 43 00 80 CB 43 00 A0 CB 43 00 00 CC 43 00   TC И'C a'C  'C
0065E180  60 CC 43 00 D0 CC 43 00 30 CD 43 00 90 CD 43 00  ''C ''C 0'C ''C
0065E190  F0 CD 43 00 50 CE 43 00 B0 CE 43 00 10 CF 43 00  ''C P'C ░'C ▶'C
0065E1A0  80 8F 44 00 A0 8F 44 00 00 90 44 00 60 90 44 00  И'D a'D  'D ''D
0065E1B0  D0 90 44 00 30 91 44 00 90 91 44 00 F0 91 44 00  ''D 0CD 'CD 'CD
0065E1C0  50 92 44 00 B0 92 44 00 10 93 44 00 D0 C3 44 00  PTD ░TD ▶УD ''D
0065E1D0  30 C6 44 00 D0 C8 44 00 E0 D1 44 00 50 D2 44 00  0'D ''D ''D P'D
0065E1E0  20 DC 44 00 D0 E9 44 00 60 F4 44 00 80 F4 44 00   'D ''D ''D И'D
0065E1F0  90 FB 44 00 00 06 45 00 20 06 45 00 80 06 45 00  ''D  ♠E  ♠E И♠E
0065E200  E0 06 45 00 50 07 45 00 B0 07 45 00 10 08 45 00  '♠E P•E ░•E ▶◘E
0065E210  70 08 45 00 D0 08 45 00 30 09 45 00 90 09 45 00  p◘E '◘E 0○E '○E
0065E220  40 5B 45 00 80 6B 45 00 80 72 45 00 A0 72 45 00  @[E ИkE ИrE arE
0065E230  00 73 45 00 60 73 45 00 D0 73 45 00 30 74 45 00   sE 'sE 'sE 0tE
0065E240  90 74 45 00 F0 74 45 00 50 75 45 00 B0 75 45 00  'tE 'tE PuE ░uE
0065E250  10 76 45 00 A0 DA 45 00 F0 DD 45 00 10 DE 45 00  ▶vE a'E ''E ▶'E
0065E260  B0 DF 45 00 D0 DF 45 00 30 E0 45 00 90 E0 45 00  ░'E ''E 0'E ''E
0065E270  00 E1 45 00 60 E1 45 00 C0 E1 45 00 20 E2 45 00   'E ''E ''E  'E
0065E280  80 E2 45 00 E0 E2 45 00 40 E3 45 00 10 04 46 00  И'E ''E @'E ▶♦F
```

```
02B3FF74  778BFA29  )'Лw  RETURN to KERNEL32.BaseThreadInitThunk+19
02B3FF78  003A9000   ':
02B3FF7C  778BFA10  ▶'Лw  KERNEL32.BaseThreadInitThunk
02B3FF80  02B3FFDC  '''@
02B3FF84  77D376B4  'v'w  RETURN to ntdll.77D376B4
02B3FF88  003A9000   ':
02B3FF8C  160B297D  })♂▬
02B3FF90  00000000
02B3FF94  00000000
02B3FF98  003A9000   ':
02B3FF9C  00000000
02B3FFA0  00000000
02B3FFA4  00000000
02B3FFA8  00000000
02B3FFAC  00000000
02B3FFB0  00000000
02B3FFB4  00000000
02B3FFB8  00000000
02B3FFBC  00000000
02B3FFC0  00000000
02B3FFC4  02B3FF8C  '''@
02B3FFC8  00000000
02B3FFCC  02B3FFE4  '''@  Pointer to next SEH record
02B3FFD0  77D49A60  '''w  SE handler
02B3FFD4  63651019  ↓▶ec
02B3FFD8  00000000
02B3FFDC  02B3FFEC  '''@
02B3FFE0  77D37684  Дv'w  RETURN from ntdll.77D37685 to ntdll.77D37684
02B3FFE4  FFFFFFFF  ''''  End of SEH chain
02B3FFE8  77D57403  ●t'w  SE handler
02B3FFEC  00000000
02B3FFF0  00000000
02B3FFF4  00701000   ▶p   h3era_HD.<ModuleEntryPoint>
02B3FFF8  003A9000   ':
02B3FFFC  00000000
```

Entry point of main module

Press ctrl+G and enter the adress you have found in IDA:


OllyDbg -
File View Debug Trace Plugins Options Windows Help
L E M W T C R ... K B M H

```
00700FF7      00                      DB 00
00700FF8      00                      DB 00
00700FF9      00                      DB 00
00700FFA      00                      DB 00
00700FFB      00                      DB 00
00700FFC      00                      DB 00
00700FFD      00                      DB 00
00700FFE      00                      DB 00
00700FFF      00                      DB 00
00701000  $   60                      PUSHAD
00701001  .   68 18107000             PUSH 00701018                              FileName = "era.dll"
00701006  .   FF15 FCA16300           CALL DWORD PTR DS:[<&KeRNeL32.LoadLibra    KERNEL32.LoadLibraryA
0070100C  .   FF15 68768800           CALL DWORD PTR DS:[887668]
00701012  .   61                      POPAD
00701013  .^  E9 6C98F1FF             JMP 0061A884
00701018  .   65 72 61 2E 64 6C 6C 00 ASCII "era.dll",0                       ASCII "era.dll"
00701020  .   90                      NOP
00701021  .   90                      NOP
00701022  .   90                      NOP
00701023  .   90                      NOP
00701024  .   90                      NOP
00701025  .   90                      NOP
00701026  .   90                      NOP
00701027  .   90                      NOP
00701028  .   90                      NOP
00701029  $   55                      PUSH EBP                                   h3era_HD.00701029(guessed Arg1,Arg2)
0070102A  .   8BEC                    MOV EBP,ESP
0070102C  .   51                      PUSH ECX
0070102D  .   8B45 08                 MOV EAX,DWORD PTR SS:[ARG.1]
00701030  .   50                      PUSH EAX                                   Arg1 => [ARG.1]
00701031  .   E8 6A000700             CALL 007710A0                              h3era_HD.007710A0
00701036  .   83C4 04                 ADD ESP,4
00701039  .   8945 FC                 MOV DWORD PTR SS:[LOCAL.1],EAX
0070103C  .   837D FC 00              CMP DWORD PTR SS:[LOCAL.1],0
00701040  .v  75 04                   JNE SHORT 00701046
00701042  .   33C0                    XOR EAX,EAX
00701044  .v  EB 10                   JMP SHORT 00701056
00701046  >   8B4D 0C                 MOV ECX,DWORD PTR SS:[ARG.2]
00701049  .   51                      PUSH ECX
0070104A  .   8B55 FC                 MOV EDX,DWORD PTR SS:[LOCAL.1]
0070104D  .   52                      PUSH EDX
0070104E  .   E8 69000700             CALL 007710BC
00701053  .   83C4 08                 ADD ESP,8
00701056  >   8BE5                    MOV ESP,EBP
00701058  .   5D                      POP EBP
00701059  .   C3                      RETN
0070105A  .   55                      PUSH EBP
0070105B  .   8BEC                    MOV EBP,ESP
0070105D  .   83EC 20                 SUB ESP,20
00701060  .   A1 00064000             MOV EAX,DWORD PTR DS:[400600]
00701065  .   8945 FC                 MOV DWORD PTR SS:[LOCAL.1],EAX
00701068  .   8B0D 04064000           MOV ECX,DWORD PTR DS:[400604]
0070106E  .   894D F8                 MOV DWORD PTR SS:[LOCAL.2],ECX
00701071  .   C745 F0 00000000        MOV DWORD PTR SS:[LOCAL.4],0
00701078  .v  EB 09                   JMP SHORT 00701083
0070107A  >   8B55 F0                 MOV EDX,DWORD PTR SS:[LOCAL.4]
0070107D  .   83C2 01                 ADD EDX,1
00701080  .   8955 F0                 MOV DWORD PTR SS:[LOCAL.4],EDX
00701083  >   8B45 F0                 MOV EAX,DWORD PTR SS:[LOCAL.4]
00701086  .   6BC0 14                 IMUL EAX,EAX,14
00701089  .   8B4D FC                 MOV ECX,DWORD PTR SS:[LOCAL.1]
0070108C  .   833C01 00               CMP DWORD PTR DS:[EAX+ECX],0
```

```
Address   Hex dump                                         ASCII
0065E000  00 00 00 00 00 10 40 00 60 10 40 00 C0 10 40 00
0065E010  30 11 40 00 90 11 40 00 F0 11 40 00 50 12 40 00
0065E020  B0 12 40 00 10 13 40 00 70 13 40 00 70 4D 40 00
0065E030  90 4D 40 00 F0 4D 40 00 50 4E 40 00 C0 4E 40 00
0065E040  20 4F 40 00 80 4F 40 00 E0 4F 40 00 40 50 40 00
0065E050  A0 50 40 00 00 51 40 00 C0 58 40 00 E0 58 40 00
0065E060  40 59 40 00 A0 59 40 00 10 5A 40 00 70 5A 40 00
0065E070  D0 5A 40 00 30 5B 40 00 90 5B 40 00 F0 5B 40 00
0065E080  50 5C 40 00 40 5D 40 00 A0 F1 40 00 90 B6 41 00
0065E090  80 C0 41 00 A0 C0 41 00 00 C1 41 00 60 C1 41 00
0065E0A0  D0 C1 41 00 30 C2 41 00 90 C2 41 00 F0 C2 41 00
0065E0B0  50 C3 41 00 B0 C3 41 00 10 C4 41 00 20 DF 41 00
[illegible]
```

```
RETURN to KERNEL32.BaseThreadInitThunk
KERNEL32.BaseThreadInitThunk
RETURN to ntdll.77D376B4
Pointer to next SEH record
SE handler
RETURN from ntdll.77D37685 to ntdll.77
End of SEH chain
SE handler
h3era_HD.<ModuleEntryPoint>
```

Enter expression to follow

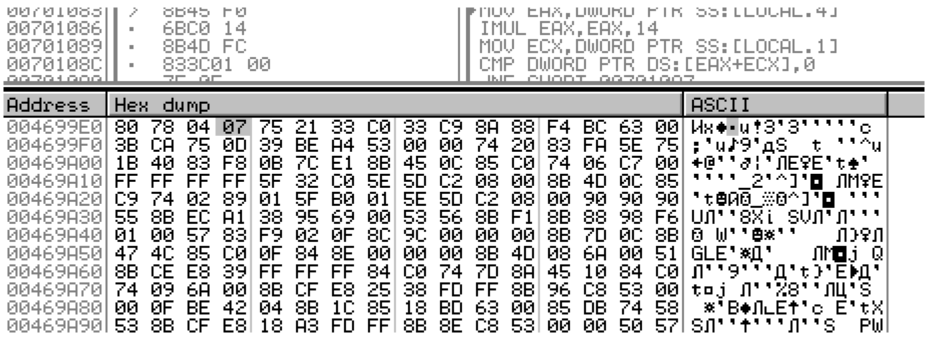
Enter address expression:
4699E0

Matching labels:

Follow expression   Cancel

Entry point of main module

Here, at this address, you can find data determining assembly instruction and it's parameters. Find town number of fortress *(which is 7 (Format T) –also remember values here are in hex)*

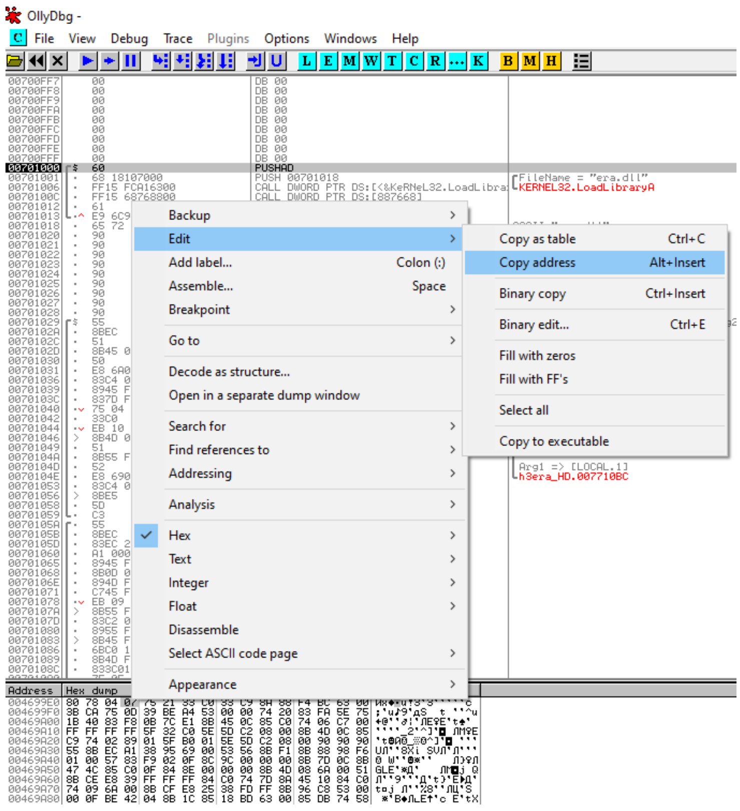


```
00701086  .  6BC0 14      IMUL EAX,EAX,14
00701089  .  8B4D FC      MOV ECX,DWORD PTR SS:[LOCAL.1]
0070108C  .  833C01 00    CMP DWORD PTR DS:[EAX+ECX],0
```

```
Address   Hex dump                                         ASCII
004699E0  80 78 04 07 75 21 33 C0 33 C9 8A 88 F4 BC 63 00
004699F0  3B CA 75 0D 39 BE A4 53 00 00 74 20 83 FA 5E 75
00469A00  1B 40 83 F8 0B 7C E1 8B 45 0C 85 C0 74 06 C7 00
00469A10  FF FF FF FF 5F 32 C0 5E 5D C2 08 00 8B 4D 0C 85
00469A20  C9 74 02 89 01 5F B0 01 5E 5D C2 08 00 90 90 90
00469A30  55 8B EC A1 38 95 69 00 53 56 8B F1 8B 88 98 F6
00469A40  01 00 57 83 F9 02 0F 8C 9C 00 00 00 8B 7D 0C 8B
00469A50  47 4C 85 C0 0F 84 8E 00 00 00 8B 4D 08 6A 00 51
00469A60  8B CE E8 39 FF FF 84 C0 74 7D 8A 45 10 84 C0
00469A70  74 09 6A 00 8B CF E8 25 38 FD FF 8B 96 C8 53 00
00469A80  00 0F BE 42 04 8B 1C 85 18 BD 63 00 85 DB 74 58
00469A90  53 8B CF E8 18 A3 FD FF 8B 8E C8 53 00 00 50 57
```

On this example, we can see that Town type used in comparison *(which is hardcoded)* is stored in byte on address 0x4699E3 *(every two hex values represent one byte)* Making the comparison to always return false would be more complicated, so easiest solution is to change that hardcoded value to some non-existant town, for example Town=99.

Copy address by rightclick on the value -> edit -> copy address



```
00700FF7    00              DB 00
00700FF8    00              DB 00
00700FF9    00              DB 00
00700FFA    00              DB 00
00700FFB    00              DB 00
00700FFC    00              DB 00
00700FFD    00              DB 00
00700FFE    00              DB 00
00700FFF    00              DB 00
00701000  $ 60              PUSHAD
00701001  . 68 18107000     PUSH 00701018                             FileName = "era.dll"
00701006  . FF15 FCA16300   CALL DWORD PTR DS:[<&KeRNeL32.LoadLibra   KERNEL32.LoadLibraryA
0070100C  . FF15 68768800   CALL DWORD PTR DS:[887668]
00701012  . 61
00701013  . E9 6C9
00701018  . 65 72
00701020  . 90
00701021  . 90
00701022  . 90
00701023  . 90
00701024  . 90
00701025  . 90
00701026  . 90
00701027  . 90
00701028  . 90
00701029  $ 55
0070102A  . 8BEC
0070102C  . 51
0070102D  . 8B45 0
00701030  . 50
00701031  . E8 6A0
00701036  . 83C4 0
00701039  . 8945 F
0070103C  . 837D F
00701040  . 75 04
00701042  . 33C0
00701044  . EB 10
00701046  > 8B4D 0
00701049  . 51
0070104A  . 8B55 F
0070104D  . 52
0070104E  . E8 690                                                    Arg1 => [LOCAL.1]
00701053  . 83C4 0                                                    h3era_HD.007710BC
00701056  > 8BE5
00701058  . 5D
00701059  . C3
0070105A  . 55
0070105B  . 8BEC
0070105D  . 83EC 2
00701060  . A1 000
00701065  . 8945 F
00701068  . 8B0D 0
0070106E  . 894D F
00701071  . C745 F
00701078  . EB 09
0070107A  > 8B55 F
0070107D  . 83C2 0
00701080  . 8955 F
00701083  > 8B45 F
00701086  . 6BC0 1
00701089  . 8B4D F
0070108C  . 833C01
```

```
Address   Hex dump
004699E0  80 78 04 07 75 21 33 C0 33 C9 8A 88 F4 BC 63 00
004699F0  3B CA 75 0D 39 BE A4 53 00 00 74 20 83 FA 5E 75
00469A00  1B 40 83 F8 0B 7C E1 8B 45 0C 85 C0 74 06 C7 00
00469A10  FF FF FF FF 5F 32 C0 5E 5D C2 08 00 8B 4D 0C 85
00469A20  C9 74 02 89 01 5F B0 01 5E 5D C2 08 00 90 90 90
00469A30  55 8B EC A1 38 95 69 00 53 56 8B F1 8B 88 98 F6
00469A40  01 00 57 83 F9 02 0F 8C 9C 00 00 00 8B 7D 0C 8B
00469A50  47 4C 85 C0 0F 84 8E 00 00 00 8B 4D 08 6A 00 51
00469A60  8B CE E8 39 FF FF FF 84 C0 74 7D 8A 45 10 84 C0
00469A70  74 09 6A 00 8B CF E8 25 38 FD FF 8B 96 C8 53 00
00469A80  00 0F BE 42 04 8B 1C 85 18 BD 63 00 85 DB 74 58
```

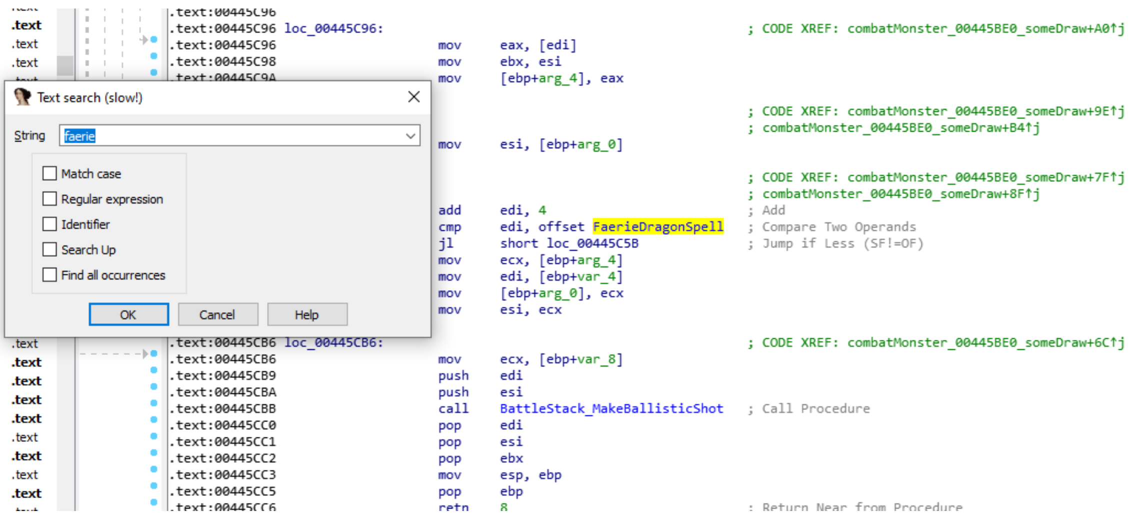
Now you can use it. There’re many ways to change value in memory in WoG 3.59: by Lua mem, by ERM UN:C, by using Copiers array… but let’s stick to easiest solution for now.

*(Byte *)0x4699E3 = 99;

Done! This will remove damage taken from first row of moat in fortress – it will still stop when stepped upon though.

Example 2 – Make Faerie Dragon to spell „Magic arrow” instead of „Chain Lightning”

Just start searching (**alt+T**) and find what you need



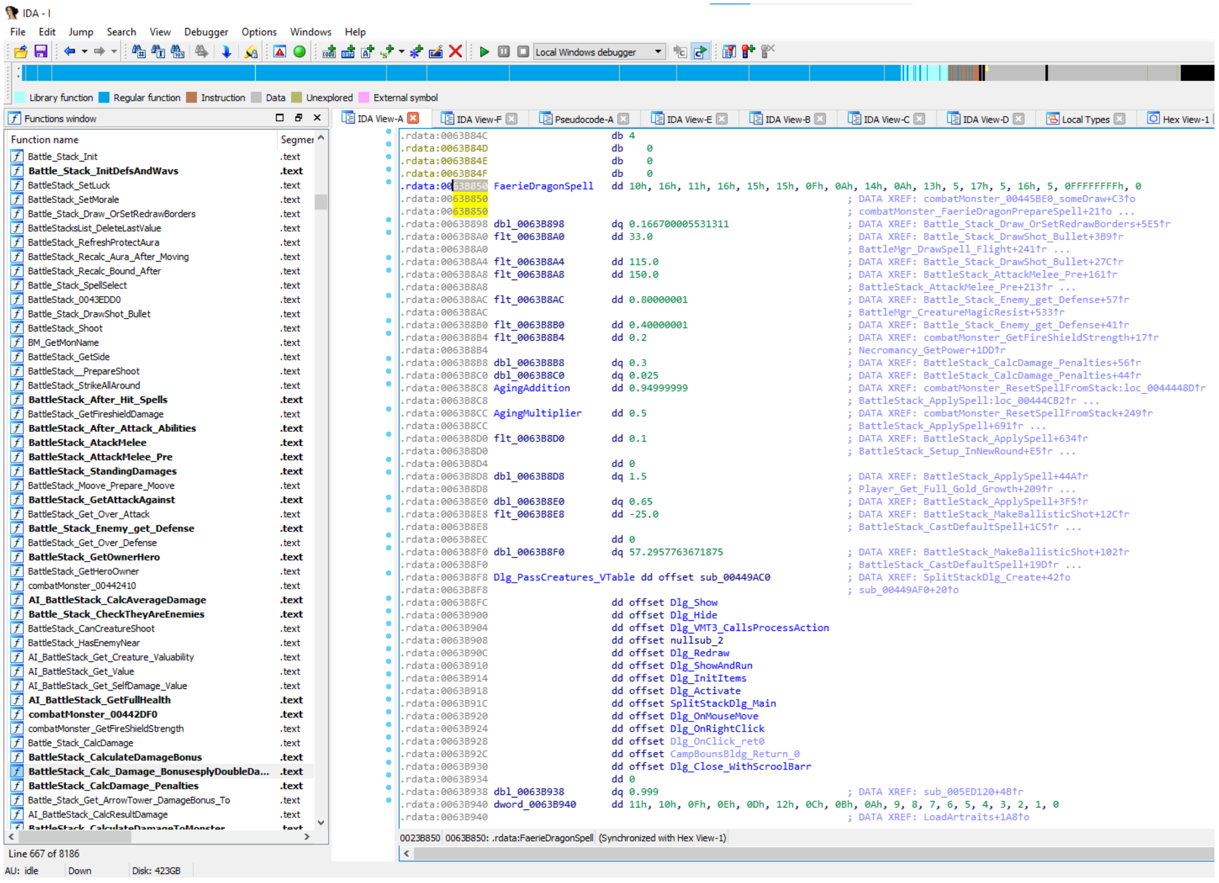
```
.text:00445C96
.text:00445C96 loc_00445C96:                                          ; CODE XREF: combatMonster_00445BE0_someDraw+A0↑j
.text:00445C96                     mov     eax, [edi]
.text:00445C98                     mov     ebx, esi
.text:00445C9A                     mov     [ebp+arg_4], eax
                                                                      ; CODE XREF: combatMonster_00445BE0_someDraw+9E↑j
                                                                      ; combatMonster_00445BE0_someDraw+B4↑j
                                   mov     esi, [ebp+arg_0]
                                                                      ; CODE XREF: combatMonster_00445BE0_someDraw+7F↑j
                                                                      ; combatMonster_00445BE0_someDraw+8F↑j
                                   add     edi, 4                     ; Add
                                   cmp     edi, offset FaerieDragonSpell ; Compare Two Operands
                                   jl      short loc_00445C5B         ; Jump if Less (SF!=OF)
                                   mov     ecx, [ebp+arg_4]
                                   mov     edi, [ebp+var_4]
                                   mov     [ebp+arg_0], ecx
                                   mov     esi, ecx
.text:00445CB6 loc_00445CB6:                                          ; CODE XREF: combatMonster_00445BE0_someDraw+6C↑j
.text:00445CB6                     mov     ecx, [ebp+var_8]
.text:00445CB9                     push    edi
.text:00445CBA                     push    esi
.text:00445CBB                     call    BattleStack_MakeBallisticShot ; Call Procedure
.text:00445CC0                     pop     edi
.text:00445CC1                     pop     esi
.text:00445CC2                     pop     ebx
.text:00445CC3                     mov     esp, ebp
.text:00445CC5                     pop     ebp
.text:00445CC6                     retn    8                          ; Return Near from Procedure
```

Text search (slow!)

String: faerie

- [ ] Match case
- [ ] Regular expression
- [ ] Identifier
- [ ] Search Up
- [ ] Find all occurrences

double click on the "FaerieDragonSpell", and you will go to the offset



Function name	Segment
Battle_Stack_Init	.text
Battle_Stack_InitDefsAndWavs	**.text**
BattleStack_SetLuck	.text
BattleStack_SetMorale	.text
Battle_Stack_Draw_OrSetRedrawBorders	.text
BattleStacksList_DeleteLastValue	.text
BattleStack_RefreshProtectAura	.text
BattleStack_Recalc_Aura_After_Moving	.text
BattleStack_Recalc_Bound_After	.text
Battle_Stack_SpellSelect	.text
BattleStack_0043EDD0	.text
Battle_Stack_DrawShot_Bullet	.text
BattleStack_Shoot	.text
BM_GetMonName	.text
BattleStack_GetSide	.text
BattleStack__PrepareShoot	.text
BattleStack_StrikeAllAround	.text
BattleStack_After_Hit_Spells	**.text**
BattleStack_GetFireshieldDamage	.text
BattleStack_After_Attack_Abilities	**.text**
BattleStack_AtackMelee	**.text**
BattleStack_AttackMelee_Pre	**.text**
BattleStack_StandingDamages	**.text**
BattleStack_Moove_Prepare_Moove	.text
BattleStack_GetAttackAgainst	**.text**
BattleStack_Get_Over_Attack	.text
Battle_Stack_Enemy_get_Defense	**.text**
BattleStack_Get_Over_Defense	.text
BattleStack_GetOwnerHero	**.text**
BattleStack_GetHeroOwner	.text
combatMonster_00442410	.text
AI_BattleStack_CalcAverageDamage	**.text**
Battle_Stack_CheckTheyAreEnemies	**.text**
BattleStack_CanCreatureShoot	.text
BattleStack_HasEnemyNear	.text
AI_BattleStack_Get_Creature_Valuability	.text
AI_BattleStack_Get_Value	.text
AI_BattleStack_Get_SelfDamage_Value	.text
AI_BattleStack_GetFullHealth	**.text**
combatMonster_00442DF0	**.text**
combatMonster_GetFireShieldStrength	.text
Battle_Stack_CalcDamage	.text
BattleStack_CalculateDamageBonus	**.text**
BattleStack_Calc_Damage_BonusesplyDoubleDa...	**.text**
BattleStack_CalcDamage_Penalties	**.text**
Battle_Stack_Get_ArrowTower_DamageBonus_To	.text
AI_BattleStack_CalcResultDamage	.text
BattleStack_CalculateDamageToMonster	**.text**

Line 667 of 8186

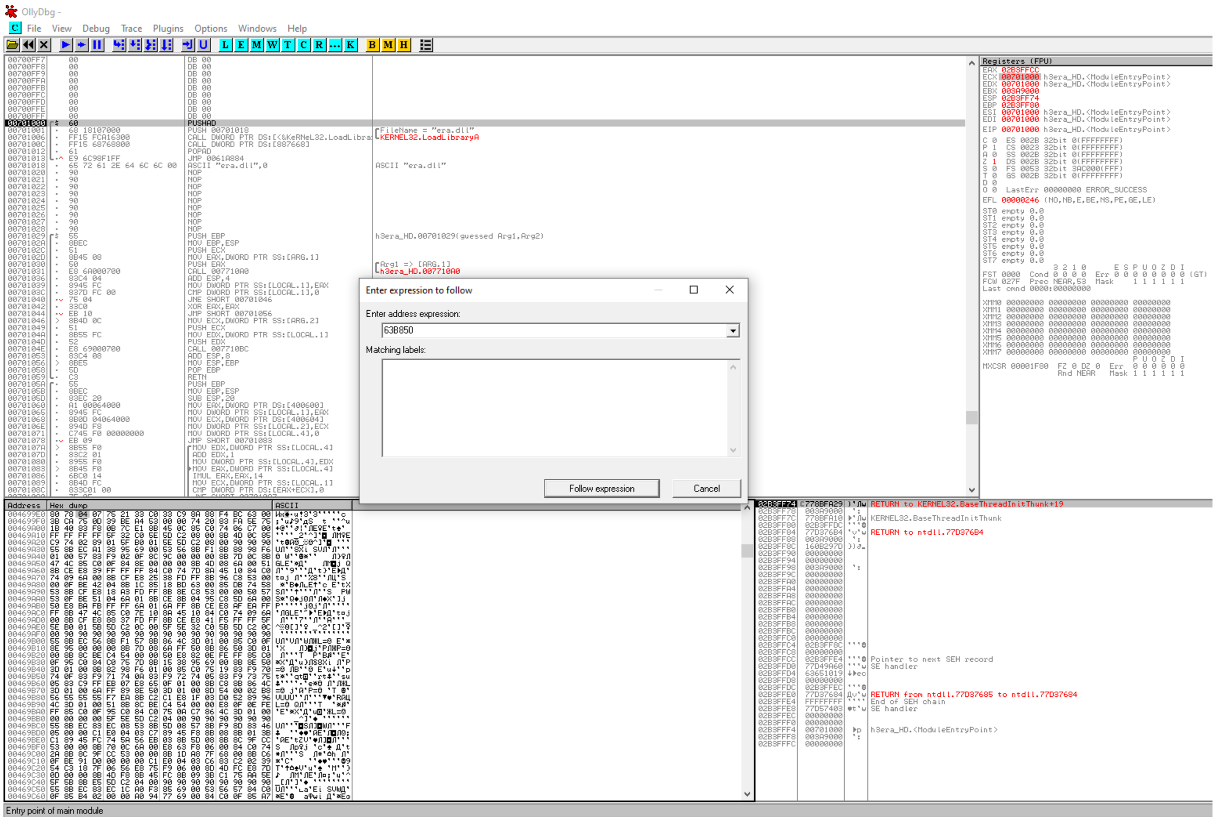
AU: idle   Down   Disk: 423GB

```
.rdata:0063B84C                  db 4
.rdata:0063B84D                  db    0
.rdata:0063B84E                  db    0
.rdata:0063B84F                  db    0
.rdata:0063B850 FaerieDragonSpell dd 10h, 16h, 11h, 16h, 15h, 15h, 0Fh, 0Ah, 14h, 0Ah, 13h, 5, 17h, 5, 16h, 5, 0FFFFFFFFh, 0
.rdata:0063B850                                   ; DATA XREF: combatMonster_00445BE0_someDraw+C3↑o
.rdata:0063B850                                   ; combatMonster_FaerieDragonPrepareSpell+21↑o ...
.rdata:0063B898 dbl_0063B898     dq 0.166700005531311 ; DATA XREF: Battle_Stack_Draw_OrSetRedrawBorders+5E5↑r
.rdata:0063B8A0 flt_0063B8A0     dd 33.0          ; DATA XREF: Battle_Stack_DrawShot_Bullet+3B9↑r
.rdata:0063B8A0                                   ; BattleMgr_DrawSpell_Flight+241↑r ...
.rdata:0063B8A4 flt_0063B8A4     dd 115.0         ; DATA XREF: Battle_Stack_DrawShot_Bullet+27C↑r
.rdata:0063B8A8 flt_0063B8A8     dd 150.0         ; DATA XREF: BattleStack_AttackMelee_Pre+161↑r
.rdata:0063B8A8                                   ; BattleStack_AttackMelee_Pre+213↑r ...
.rdata:0063B8AC flt_0063B8AC     dd 0.80000001    ; DATA XREF: Battle_Stack_Enemy_get_Defense+57↑r
.rdata:0063B8AC                                   ; BattleMgr_CreatureMagicResist+533↑r
.rdata:0063B8B0 flt_0063B8B0     dd 0.40000001    ; DATA XREF: Battle_Stack_Enemy_get_Defense+41↑r
.rdata:0063B8B4 flt_0063B8B4     dd 0.2           ; DATA XREF: combatMonster_GetFireShieldStrength+17↑r
.rdata:0063B8B4                                   ; Necromancy_GetPower+1DD↑r
.rdata:0063B8B8 dbl_0063B8B8     dq 0.3           ; DATA XREF: BattleStack_CalcDamage_Penalties+56↑r
.rdata:0063B8C0 dbl_0063B8C0     dq 0.025         ; DATA XREF: BattleStack_CalcDamage_Penalties+44↑r
.rdata:0063B8C8 AgingAddition    dd 0.94999999    ; DATA XREF: combatMonster_ResetSpellFromStack:loc_0044448D↑r
.rdata:0063B8C8                                   ; BattleStack_ApplySpell:loc_00444CB2↑r ...
.rdata:0063B8CC AgingMultiplier  dd 0.5           ; DATA XREF: combatMonster_ResetSpellFromStack+249↑r
.rdata:0063B8CC                                   ; BattleStack_ApplySpell+691↑r ...
.rdata:0063B8D0 flt_0063B8D0     dd 0.1           ; DATA XREF: BattleStack_ApplySpell+634↑r
.rdata:0063B8D0                                   ; BattleStack_Setup_InNewRound+E5↑r ...
.rdata:0063B8D4                  dd 0
.rdata:0063B8D8 dbl_0063B8D8     dq 1.5           ; DATA XREF: BattleStack_ApplySpell+44A↑r
.rdata:0063B8D8                                   ; Player_Get_Full_Gold_Growth+209↑r ...
.rdata:0063B8E0 dbl_0063B8E0     dq 0.65          ; DATA XREF: BattleStack_ApplySpell+3F5↑r
.rdata:0063B8E8 flt_0063B8E8     dd -25.0         ; DATA XREF: BattleStack_MakeBallisticShot+12C↑r
.rdata:0063B8E8                                   ; BattleStack_CastDefaultSpell+1C5↑r ...
.rdata:0063B8EC                  dd 0
.rdata:0063B8F0 dbl_0063B8F0     dq 57.2957763671875 ; DATA XREF: BattleStack_MakeBallisticShot+102↑r
.rdata:0063B8F0                                   ; BattleStack_CastDefaultSpell+19D↑r ...
.rdata:0063B8F8 Dlg_PassCreatures_VTable dd offset sub_00449AC0 ; DATA XREF: SplitStackDlg_Create+42↑o
.rdata:0063B8F8                                   ; sub_00449AF0+20↑o
.rdata:0063B8FC                  dd offset Dlg_Show
.rdata:0063B900                  dd offset Dlg_Hide
.rdata:0063B904                  dd offset Dlg_VMT3_CallsProcessAction
.rdata:0063B908                  dd offset nullsub_2
.rdata:0063B90C                  dd offset Dlg_Redraw
.rdata:0063B910                  dd offset Dlg_ShowAndRun
.rdata:0063B914                  dd offset Dlg_InitItems
.rdata:0063B918                  dd offset Dlg_Activate
.rdata:0063B91C                  dd offset SplitStackDlg_Main
.rdata:0063B920                  dd offset Dlg_OnMouseMove
.rdata:0063B924                  dd offset Dlg_OnRightClick
.rdata:0063B928                  dd offset Dlg_OnClick_ret0
.rdata:0063B92C                  dd offset CampBounsBldg_Return_0
.rdata:0063B930                  dd offset Dlg_Close_WithScroolBarr
.rdata:0063B934                  dd 0
.rdata:0063B938 dbl_0063B938     dq 0.999         ; DATA XREF: sub_005ED120+4B↑r
.rdata:0063B940 dword_0063B940   dd 11h, 10h, 0Fh, 0Eh, 0Dh, 12h, 0Ch, 0Bh, 0Ah, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0
.rdata:0063B940                                   ; DATA XREF: LoadArtraits+1A8↑o
```

0023B850 0063B850: .rdata:FaerieDragonSpell (Synchronized with Hex View-1)

Select and copy the address

Go to Olly, click on the Hex Dump, press ctrl+G and follow the expression

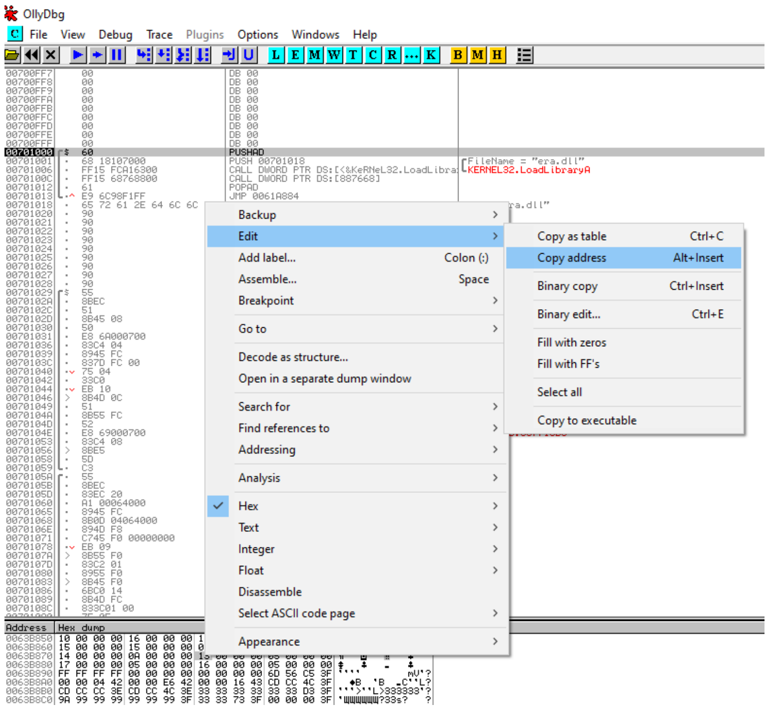

Enter expression to follow

Enter address expression:

```
63B850
```

Matching labels:

the value 0x13=19 is the chain lightning (Format SP), copy it's address and do what you want



```
00700FF7     00                 DB 00
00700FF8     00                 DB 00
00700FF9     00                 DB 00
00700FFA     00                 DB 00
00700FFB     00                 DB 00
00700FFC     00                 DB 00
00700FFD     00                 DB 00
00700FFE     00                 DB 00
00700FFF     00                 DB 00
00701000  ┌$ 60                 PUSHAD
00701001  │. 68 18107000        PUSH 00701018                               ┌FileName = "era.dll"
00701006  │. FF15 FCA16300      CALL DWORD PTR DS:[<&KeRNeL32.LoadLibra     └KERNEL32.LoadLibraryA
0070100C  │. FF15 68768800      CALL DWORD PTR DS:[887668]
00701012  │. 61                 POPAD
00701013  └.^ E9 6C98F1FF       JMP 0061A884
00701018   . 65 72 61 2E 64 6C 6C                                          ra.dll"
00701020   . 90
00701021   . 90
00701022   . 90
00701023   . 90
00701024   . 90
00701025   . 90
00701026   . 90
00701027   . 90
00701028   . 90
00701029  ┌$ 55
0070102A  │. 8BEC
0070102C  │. 51
0070102D  │. 8B45 08
00701030  │. 50
00701031  │. E8 6A000700
00701036  │. 83C4 04
00701039  │. 8945 FC
0070103C  │. 837D FC 00
00701040  │.v 75 04
00701042  │. 33C0
00701044  │.v EB 10
00701046  │> 8B4D 0C
00701049  │. 51
0070104A  │. 8B55 FC
0070104D  │. 52
0070104E  │. E8 69000700
00701053  │. 83C4 08
00701056  │> 8BE5
00701058  │. 5D
00701059  └. C3
0070105A  ┌. 55
0070105B  │. 8BEC
0070105D  │. 83EC 20
00701060  │. A1 00064000
00701065  │. 8945 FC
00701068  │. 8B0D 04064000
0070106E  │. 894D F8
00701071  │. C745 F0 00000000
00701078  │.v EB 09
0070107A  │> 8B55 F0
0070107D  │. 83C2 01
00701080  │. 8955 F0
00701083  │> 8B45 F0
00701086  │. 6BC0 14
00701089  │. 8B4D FC
0070108C  │. 833C01 00
```

```
Address   Hex dump
0063B850  10 00 00 00 16 00 00 00 1
0063B860  15 00 00 00 15 00 00 00 0
0063B870  14 00 00 00 0A 00 00 00 13 00 00 00 05 00 00 00
0063B880  17 00 00 00 05 00 00 00 16 00 00 00 05 00 00 00
0063B890  FF FF FF FF 00 00 00 00 00 00 00 00 6D 56 C5 3F  ''''        mV'?
0063B8A0  00 00 04 42 00 00 E6 42 00 00 16 43 CD CC 4C 3F    ◆B  'B  C''L?
0063B8B0  CD CC CC 3E CD CC 4C 3E 33 33 33 33 33 33 D3 3F  '''>''L>333333'?
0063B8C0  9A 99 99 99 99 99 99 3F 33 33 73 3F 00 00 00 3F  'ЩЩЩЩЩЩ?33s?   ?
```

```
*(Byte *)0x63B878 = 0x0F; //replace Chain Lightning with Magic Arrow
```