PrintPlayerStats

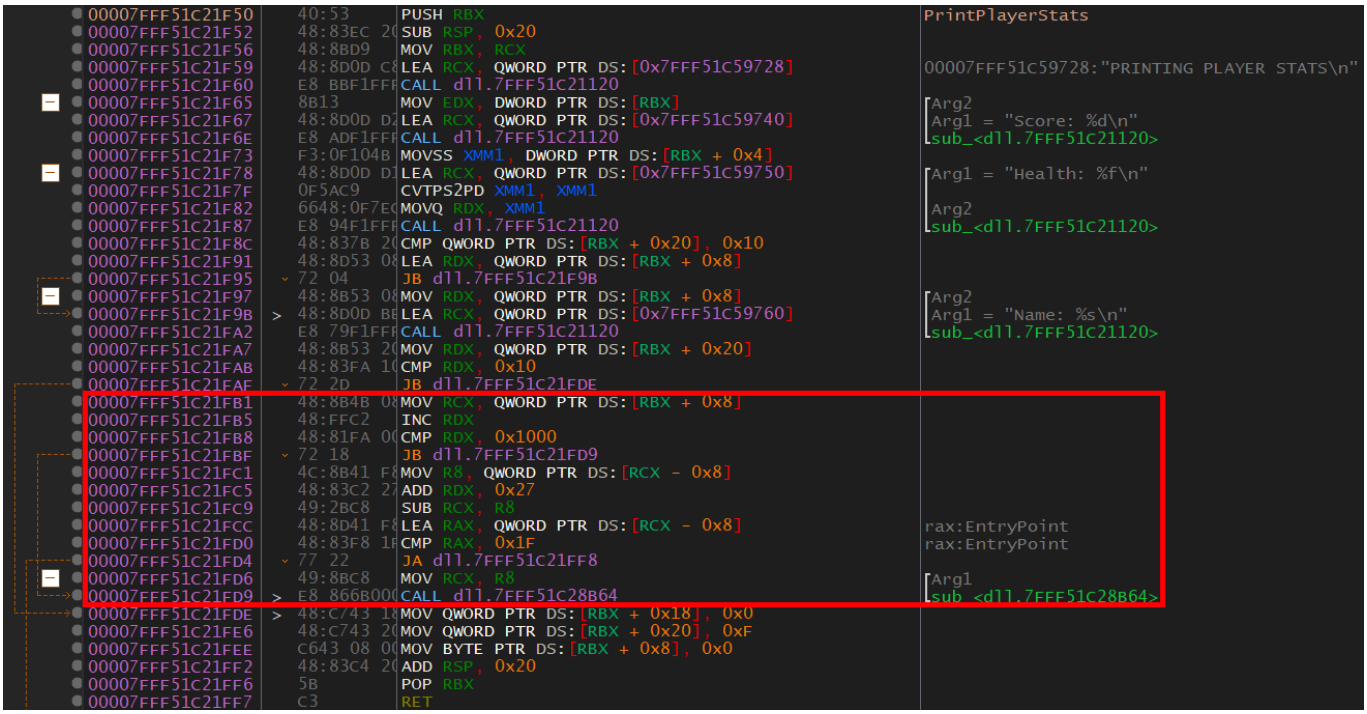
Before we use `InitializePlayer` in our own program, let's see what other information we can find about the class. To find more information, let's look at `PrintPlayerStats`.

```
00007FFF51C21F50  40:53        PUSH RBX                                  PrintPlayerStats
00007FFF51C21F52  48:83EC 20   SUB RSP, 0x20
00007FFF51C21F56  48:8BD9      MOV RBX, RCX
00007FFF51C21F59  48:8D0D C8   LEA RCX, QWORD PTR DS:[0x7FFF51C59728]     00007FFF51C59728:"PRINTING PLAYER STATS\n"
00007FFF51C21F60  E8 BBF1FFFF  CALL dll.7FFF51C21120
00007FFF51C21F65  8B13         MOV EDX, DWORD PTR DS:[RBX]               Arg2
00007FFF51C21F67  48:8D0D D2   LEA RCX, QWORD PTR DS:[0x7FFF51C59740]     Arg1 = "Score: %d\n"
00007FFF51C21F6E  E8 ADF1FFFF  CALL dll.7FFF51C21120                     sub_<dll.7FFF51C21120>
00007FFF51C21F73  F3:0F104B    MOVSS XMM1, DWORD PTR DS:[RBX + 0x4]
00007FFF51C21F78  48:8D0D D1   LEA RCX, QWORD PTR DS:[0x7FFF51C59750]     Arg1 = "Health: %f\n"
00007FFF51C21F7F  0F5AC9       CVTPS2PD XMM1, XMM1
00007FFF51C21F82  6648:0F7E0   MOVQ RDX, XMM1                            Arg2
00007FFF51C21F87  E8 94F1FFFF  CALL dll.7FFF51C21120                     sub_<dll.7FFF51C21120>
00007FFF51C21F8C  48:837B 20   CMP QWORD PTR DS:[RBX + 0x20], 0x10
00007FFF51C21F91  48:8D53 08   LEA RDX, QWORD PTR DS:[RBX + 0x8]
00007FFF51C21F95  72 04        JB dll.7FFF51C21F9B
00007FFF51C21F97  48:8B53 08   MOV RDX, QWORD PTR DS:[RBX + 0x8]         Arg2
00007FFF51C21F9B  48:8D0D B8   LEA RCX, QWORD PTR DS:[0x7FFF51C59760]     Arg1 = "Name: %s\n"
00007FFF51C21FA2  E8 79F1FFFF  CALL dll.7FFF51C21120                     sub_<dll.7FFF51C21120>
00007FFF51C21FA7  48:8B53 20   MOV RDX, QWORD PTR DS:[RBX + 0x20]
00007FFF51C21FAB  48:83FA 10   CMP RDX, 0x10
00007FFF51C21FAF  72 2D        JB dll.7FFF51C21FDE
00007FFF51C21FB1  48:8B4B 08   MOV RCX, QWORD PTR DS:[RBX + 0x8]
00007FFF51C21FB5  48:FFC2      INC RDX
00007FFF51C21FB8  48:81FA 00   CMP RDX, 0x1000
00007FFF51C21FBF  72 18        JB dll.7FFF51C21FD9
00007FFF51C21FC1  4C:8B41 F8   MOV R8, QWORD PTR DS:[RCX - 0x8]
00007FFF51C21FC5  48:83C2 27   ADD RDX, 0x27
00007FFF51C21FC9  49:2BC8      SUB RCX, R8
00007FFF51C21FCC  48:8D41 F8   LEA RAX, QWORD PTR DS:[RCX - 0x8]         rax:EntryPoint
00007FFF51C21FD0  48:83F8 1F   CMP RAX, 0x1F                            rax:EntryPoint
00007FFF51C21FD4  77 22        JA dll.7FFF51C21FF8
00007FFF51C21FD6  49:8BC8      MOV RCX, R8                               Arg1
00007FFF51C21FD9  E8 866B00    CALL dll.7FFF51C28B64                     sub_<dll.7FFF51C28B64>
00007FFF51C21FDE  48:C743 18   MOV QWORD PTR DS:[RBX + 0x18], 0x0
00007FFF51C21FE6  48:C743 20   MOV QWORD PTR DS:[RBX + 0x20], 0xF
00007FFF51C21FEE  C643 08 00   MOV BYTE PTR DS:[RBX + 0x8], 0x0
00007FFF51C21FF2  48:83C4 20   ADD RSP, 0x20
00007FFF51C21FF6  5B           POP RBX
00007FFF51C21FF7  C3           RET
```

This function is actually quite simple, all it does is print information about a `Player`. I want you to try to reverse this function on your own. I challenge you to figure out what the purpose is of each line of code.

I do want to let you know something before you start. There is some extra code after the final `printf()` call. If you do not follow the `JB` after the final `printf()` call, execution goes into some memory freeing code. Feel free to reverse this if you want, but you can ignore it.

Here is the code I'm talking about (in the red box):



Anyways, good luck and have fun! We'll be reversing one more thing and then we will implement this `Player` class in our own code.