CS 69/169: Lab Assignment, Week 01 Day 01

For this lab assignment, you will be taking the provided assembly code listing (see next page) and writing the equivalent in pseudocode followed by a few questions about the listing. The pseudocode doesn't have to compile but it should be understandable to someone who knows C.

Deliverable: Write up your results in a document. We won't force you to a single style, but professionalism in reporting is a good habit to learn as a reverse engineer. Generally this means a summary of what you found, followed by a section with technical details. For formatting, it is typical to use a monospaced font for any disassembly and consistent indentation. Syntax highlighting might be handy. You may wish to consider LaTeX or Markdown.

Questions to consider:

1. What is the return value if the argument to the `sub_1161` function is `0x50`? `0x200`?
2. Were there any instructions that you hadn't seen before?[1] Find them in the Intel instruction manual and write down a brief explanation.

[1] No matter how many years you've spent looking at disassembly, there's *always* something new, each time you look at a new platform or compiler. Compilers change their optimizations all the time, CPU makers introduce new instructions and features, new vulnerabilities lead to new mitigations. Expect to use a search engine a lot to look up instructions, followed by a look in the CPU instruction set manual.

Listing

The following listing was generated using `objdump -M intel -D <somefile>`. The first column is the instrution virtual address. The next column is the machine code and the final column is the disassembled instruction.

```
0000000000001149 <sub_1149>:
    1149: f3 0f 1e fa           endbr64
    114d: 55                    push   rbp
    114e: 48 89 e5              mov    rbp,rsp
    1151: 89 7d fc              mov    DWORD PTR [rbp-0x4],edi
    1154: 89 75 f8              mov    DWORD PTR [rbp-0x8],esi
    1157: 8b 55 fc              mov    edx,DWORD PTR [rbp-0x4]
    115a: 8b 45 f8              mov    eax,DWORD PTR [rbp-0x8]
    115d: 01 d0                 add    eax,edx
    115f: 5d                    pop    rbp
    1160: c3                    ret


0000000000001161 <sub_1161>:
    1161: f3 0f 1e fa           endbr64
    1165: 55                    push   rbp
    1166: 48 89 e5              mov    rbp,rsp
    1169: 48 83 ec 08           sub    rsp,0x8
    116d: 89 7d fc              mov    DWORD PTR [rbp-0x4],edi
    1170: 81 7d fc ff 00 00 00  cmp    DWORD PTR [rbp-0x4],0xff
    1177: 7f 11                 jg     118a <sub_1161+0x29>
    1179: 8b 45 fc              mov    eax,DWORD PTR [rbp-0x4]
    117c: be ad de 00 00        mov    esi,0xdead
    1181: 89 c7                 mov    edi,eax
    1183: e8 c1 ff ff ff        call   1149 <sub_1149>
    1188: eb 0f                 jmp    1199 <sub_1161+0x38>
    118a: 8b 45 fc              mov    eax,DWORD PTR [rbp-0x4]
    118d: be 0d d0 00 00        mov    esi,0xd00d
    1192: 89 c7                 mov    edi,eax
    1194: e8 b0 ff ff ff        call   1149 <sub_1149>
    1199: c9                    leave
    119a: c3                    ret
```