

Cosc 69.16: Reverse Engineering

Prishita Dharampal

Problem 1.

```
00000000000011a0 <f>:
11a0: 48 8b 16      mov     rdx,QWORD PTR [rsi]
11a3: 48 03 17      add     rdx,QWORD PTR [rdi]
11a6: 48 89 17      mov     QWORD PTR [rdi],rdx
11a9: 48 8b 06      mov     rax,QWORD PTR [rsi]
11ac: 48 29 d0      sub     rax,rdx
11af: 48 89 06      mov     QWORD PTR [rsi],rax
11b2: 48 01 07      add     QWORD PTR [rdi],rax
11b5: 48 f7 1e      neg     QWORD PTR [rsi]
11b8: c3            ret
```

For your convenience, here is the same function written in AT&T assembly style:

```
00000000000011a0 <f>:
11a0: 48 8b 16      mov     (%rsi),%rdx
11a3: 48 03 17      add     (%rdi),%rdx
11a6: 48 89 17      mov     %rdx,(%rdi)
11a9: 48 8b 06      mov     (%rsi),%rax
11ac: 48 29 d0      sub     %rdx,%rax
11af: 48 89 06      mov     %rax,(%rsi)
11b2: 48 01 07      add     %rax,(%rdi)
11b5: 48 f7 1e      negq    (%rsi)
11b8: c3            ret
```

1. What does this function do?
2. What does this function really do? What is its most likely intended effect? Assume the behavior that is the simplest to describe (though not on all values).
3. When does the simplest behavior break? Give concrete examples.
4. **[Poll]** What looks better to you in this example: AT&T assembly style or Intel assembly style?

Solution.

- 1.
- 2.
- 3.
- 4.