Section 4
Assembly and GDB

1. Assembly to C: What does the following code do?

```
iii)         movl      (%rdi), %eax
             leal      (%eax,%eax,2), %eax
             addl      %eax, %eax
             andl      %esi, %eax
             subl      %esi, %eax
             ret
```

(14au midterm)

(((*x) * 6) & y) - y

2. C to Assembly:

Given the following C function:

```
long happy(long *x, long y, long z) {
  if (y > z)
    return z + y;
  else
    return *x;
}
```

Write **x86-64** bit assembly code for this function here. Comments are not required but could help for partial credit. We are not judging you on the efficiency of your code, just the correctness. It is fine to leave off the size suffixes if you prefer to (e.g. b, w, l, q).

(15au midterm)

```
happy:
        cmp %rdx, %rsi    # y:z
        jle  .else
        leaq (%rsi, %rdx), %rax      # y > z    %rax = z + y
        ret
.else:
        movq (%rdi), %rax  # y <= z %rax = *x
        ret
```

Also fine to swap the if and else clauses:

```
happy:
        cmp %rdx, %rsi       #y:z
        jg  .else
        movq (%rdi), %rax  # y <= z  %rax = *x ret
.else:
        leaq (%rsi, %rdx), %rax      # y > z   %rax = z + y
        ret
```