COMP.2030 Assembly LAB 10

NAME _____

1. A C function declared as

   ```
   void pp (int *p){ … }
   ```

   is compiled to the MIPS code on the right.

   Write the C code that performs the same operation as the MIPS code labeled 'pp' to the right in **no more than two** C statements.

```
pp:
    pushq    %rbp
    movq     %rsp, %rbp
    movq     %rdi, -8(%rbp)
    cmpq     $0, -8(%rbp)
    je       .L1
    movq     -8(%rbp), %rax
    addq     $4, %rax
    movq     %rax, -8(%rbp)
.L1:
    popq     %rbp
    ret
```

```
void pp (int *p) {




}
```

2. A C function prob declared as

   ```
   void prob(int *p1, int *p2){ … }
   ```

   is compiled to the x86 code on the right.
   Write the C function prob in **ONE or TWO** C statements that does the same thing as the x86 code labeled 'prob'.

```
prob: pushq %rbp
      movq  %rsp, %rbp
      movq  %rdi, -8(%rbp)
      movq  %rsi, -16(%rbp)
      movq  -16(%rbp), %rax
      movl  (%rax), %edx
      movq  -8(%rbp), %rax
      movl  (%rax), %eax
      addl  %eax, %edx
      movq  -16(%rbp), %rax
      movl  %edx, (%rax)
      popq  %rbp
      ret
```

```
void prob(int *p1, int *p2){




}
```

3. For a function with the prototyped

```
long decode2(long x, long y, long z);
```

gcc generates the assembly code on the right.
Fill in C code below that does the same thing as the x86 assembly code labeled '`decode2`' using only arithmetic operations and conditions (i.e., no bitwise operations like &, ^, ~, or shifts >>, <<).

```
# x = %rdi, y = %rsi, z = %rdx
decode2:
        subq    %rdx, %rsi
        imulq   %rsi, %rdi
        movq    %rsi, %rax
        salq    $63, %rax
        sarq    $63, %rax
        xorq    %rdi, %rax
        ret
```

```
long decode2
  (long x, long y, long z) {
```