# ICS Homework 9

November 20, 2019

1. For a C function having the general structure

```
typedef long long unsigned u64;
u64 foo(u64 x) {
        return x?foo(x-1)*x:x;
        /* or */
        /* return x?x*foo(x-1):0; */
}
```

GCC generates the following assembly code:

```
foo:
    pushq   %rbx
    movq    %rdi, %rbx
    testq   %rdi, %rdi
    jne     .L4
.L2:
    movq    %rbx, %rax
    popq    %rbx
    ret
.L4:
    leaq    -1(%rdi), %rdi
    call    foo
    imulq   %rax, %rbx
    jmp     .L2
```

Please fill in the missing expressions in the C code shown above.

2. Suppose $x_S$, the address of integer arrays $S$, and long integer index $i$ are stored in registers `%rdx` and `%rcx`, respectively. For each of the following expressions, give its type, a formula for its value, and an assembly-code implementation. The result should be stored in register `%rax` if it is a pointer and the appropriate portion of register `%rax` if it has data type `short`, `int`, or `long`.

| Expression | Type | Value | Assembly code |
|---|---|---|---|
| S[5] | int | M[$x_S$+20] | movl 20(%rdx), %eax |
| S+5 | int * | $x_S$+20 | leaq 20(%rdx), %rax |
| &S[i] | int * | $x_S$+4*i | leaq (%rdx,%rcx,4), %rax |
| ((long *)S)[i] | long | M[$x_S$+8*i] | movq (%rdx,%rcx,8), %rax |
| *((short *)&S[i]) | short | M[$x_S$+4*i] | movw (%rdx,%rcx,4), %ax |

3. Consider the following source code, where `X`, `Y` and `Z` are constants declared with `#define`:

```
typedef long long unsigned u64;
u64 P[X][Y][Z];
u64 Q[Y][Z][X];
u64 bar(u64 i, u64 j, u64 k) {
        return P[i][j][k] + Q[i][j][k];
}
```

In compiling this program, GCC generates the following assembly code:

```
bar:
    leaq    (%rsi,%rsi,4), %rax
    leaq    (%rdi,%rdi,4), %rcx
    leaq    (%rax,%rcx,4), %r8
    addq    %rdx, %r8
    leaq    (%rsi,%rsi,2), %rcx
    movq    %rdi, %rax
    salq    $4, %rax
    subq    %rdi, %rax
    addq    %rcx, %rax
    addq    %rax, %rdx
    movq    Q(,%rdx,8), %rax
    addq    P(,%r8,8), %rax
    ret
```

Use your reverse engineering skills to determine the values of `X`, `Y` and `Z` based on this assembly code.

```
#define X 3
#define Y 4
#define Z 5
```