# CS 240
# Laboratory 8 Assignment
# Disassembly and Reverse Engineering

Analyze the X86 code for the C function ***test_prime***, and answer the questions below. Assume that the function has been invoked with the argument **num**= 7.

C function to test if a number is prime

```
int test_prime(long num) {

  for (long i =2; i <= num/2;++i)  {
    if (num % i == 0) {
       return  1;
    }
  }
  return  0;
}
```

Dump of assembler code produce by **gdb** for function **test_prime**

NOTE: the <+xx> on each line represents an offset from the starting address of the function.

```
0x0000000000400480 <+0>:     mov    %rdi,%rsi
0x0000000000400483 <+3>:     shr    $0x3f,%rsi
0x0000000000400487 <+7>:     add    %rdi,%rsi
0x000000000040048a <+10>:    sar    %rsi
0x000000000040048d <+13>:    cmp    $0x1,%rsi
0x0000000000400491 <+17>:    jle    0x4004d0 <test_prime+80>
0x0000000000400493 <+19>:    mov    %rdi,%rax
0x0000000000400496 <+22>:    shr    $0x3f,%rax
0x000000000040049a <+26>:    lea    (%rdi,%rax,1),%rdx
0x000000000040049e <+30>:    and    $0x1,%edx
0x00000000004004a1 <+33>:    mov    $0x2,%ecx
0x00000000004004a6 <+38>:    cmp    %rax,%rdx
0x00000000004004a9 <+41>:    jne    0x4004bf <test_prime+63>
0x00000000004004ab <+43>:    jmp    0x4004ca <test_prime+74>
0x00000000004004ad <+45>:    mov    %rdi,%rdx
0x00000000004004b0 <+48>:    mov    %rdi,%rax
0x00000000004004b3 <+51>:    sar    $0x3f,%rdx
0x00000000004004b7 <+55>:    idiv   %rcx
0x00000000004004ba <+58>:    test   %rdx,%rdx
0x00000000004004bd <+61>:    je     0x4004ca <test_prime+74>
0x00000000004004bf <+63>:    add    $0x1,%rcx
0x00000000004004c3 <+67>:    cmp    %rsi,%rcx
0x00000000004004c6 <+70>:    jle    0x4004ad <test_prime+45>
0x00000000004004c8 <+72>:    jmp    0x4004d0 <test_prime+80>
0x00000000004004ca <+74>:    mov    $0x1,%eax
0x00000000004004cf <+79>:    retq
0x00000000004004d0 <+80>:    mov    $0x0,%eax
0x00000000004004d5 <+85>:    retq
```

1. What is the starting address of **test_prime** in memory?
2. What register is the argument stored in when the assembler code begins execution?
3. What is the purpose of *shr $0x3f, %rsi?*

4. Where register holds *i* (with an initial value of 2) ?
5. What register holds *num* / 2?
6. What register holds *num % i*?
7. Circle and label the X86 statements that tests the condition in the **for** loop. Does this code come at the beginning of the assembler code, as it does in the C program?

8. Circle and label the X86 statements that divide *num* by ***i*** and check that the remainder is 0.
9. Circle and label the statements (there are two) that set the return value for the function.