Examine the C code on the left and the corresponding X86 code on the right for the function `test_prime`

Answer the questions below assuming that `test_prime` is called with `num = 7`

### C code for function `test_prime`

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

### Disassembly of X86 code produced by `gdb` for function `test_prime`

```assembly
0x0000000000400478 : mov %rdi, -0x18(%rsp)
0x000000000040047c : movq $0x2, -0x8(%rsp)
0x0000000000400484 : jmp 0x4004a9
```

### Questions

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. Circle and label the statements (there are two) that set the return value for the function.

4. Circle and label the X86 statements that test the condition in the `for` loop. Describe how `num/2` is calculated in this code:

5. Circle and label the X86 statements that implement testing the conditional for the `if` statement in the body of the loop. Look up the `idivq` X86 instruction, and explain how the `num%2` is accomplished with the given code: