**Exercises for lesson 1: Big-Oh, lists, stacks and queues**

**Exercise 1.1**

Order the following by their O(⋅) ranking:

1. 7
2. 2
3. 5
4. 8
5. 4
6. 9
7. 1
8. 3
9. 10
10. 6

**Exercise 1.2**

Which of the following are true?

1. T
2. T
3. F
4. F
5. T
6. T

**Exercise 1.3**

What is the complexity of the following algorithm?

algorithm(*m*, *n*):  
 *r* = 1 // 1  
 for *i* from 1 to *n*: // 1 assignment, n increment, n+1 checks = 2n+2  
 *r* = *r \* m* // n multiplication, n assignment = 2n  
 return *r* // 1 return

Sum: 1 + 2n + 2 + 2n + 1 = 4n + 3 = O(n)

What does the algorithm compute?

Raises *m* to the power of *n*.

**Exercise 1.4**  
Write pseudocode for an algorithm to determine if a number is a prime. What is the time complexity of your algorithm?

isPrime(n):

if n <= 2:

return n == 2