1.

The Big Omega notation defines the shortest run time of an algorithm, while Big-Theta notation accounts for the worst-case runtime. So, by defining 𝑓(𝑛) = 𝑂(𝑔(𝑛)) and 𝑡(𝑛) = 𝑂(𝑔(𝑛)), we are deciding that 𝑓(𝑛) and 𝑡(𝑛) are the worse runtime of the function 𝑔(𝑛). These definitions would make 𝑓(𝑛) = Ω(𝑡(𝑛)), true. Since both 𝑓(𝑛) and 𝑡(𝑛) are defined using the same function 𝑔(𝑛), all of their properties would hold the same, the maximum run time and the least run time would be equal, making the statement 𝑓(𝑛) = Ω(𝑡(𝑛)) true.

2.

public void reverse() {

Node curr = head;

Node next = head.getNext();

Node prev = null;

//Base case, if it is the last note, set it to be the head

if (curr.getNext() == null) {

head = curr;

curr.setNext() = prev;

return head;

}

Node next1 = curr.getNext();

curr.setNext = prev;

reverse(next1, curr);

return head;

}