1.

Algorithm: merge(L1,L2) Requires: two sorted lists L1 and L2 Returns: a merged and sorted list of L1 and L2
1: if isEmpty(L1) 2: return L2 3: elseif isEmpty(L2) 4: return L1 5: elseif value(L1)<value(L2) 6: return cons(value(L1),merge(tail(L1),L2)) 7: else 8: return cons(value(L2),merge(L1,tail(L2))) 9: endif

Trace merge([2,5,10,15],[3,6,7]) backtracking
L1=[2,5,10,15] L2=[3,6,7]
Line 1,3 False. Line 5 True. Line 6: return cons(2,merge([5,10,15],[3,6,7])) =[2,3,5,6,7,10,15]
L1=[5,10,15] L2=[3,6,7]
Line 1,3,5 False. Line 8: return cons(3,merge([5,10,15],[6,7])) =[3,5,6,7,10,15]
L1=[5,10,15] L2=[6,7]
Line 1,3 False. Line 5 True. Line 6: return cons(5,merge([10,15],[6,7])) =[5,6,7,10,15]
L1=[10,15] L2=[6,7]
Line 1,3,5 False. Line 8: return cons(6,merge([10,15],[7])) =[6,7,10,15]
L1=[10,15] L2=[7]
Line 1,3,5 False. Line 8: return cons(7,merge([10,15],[ ])) =[7,10,15]
L1=[10,15] L2=[ ]
Line 1 False. Line 3 True. Line 4: return [10,15]

Note: steps in blue colour are the minimal elements you need to include in the tracing process.

Time complexity is $O(n)$. Because total number of comparisons of two lists' values is linearly related to the size of the merged list.

2. Refer to Lecture 6 slides .
In general, there are about $\log_2 n$ levels of splitting.
We will introduce two functions to round decimal numbers to integers.

Q3 and Q4 will be further explained in Seminar 6.

5. Demonstration with L1=[24,26,22,29,66,11,20,27,21,23,25,28].
Pivots are highlighted on each partition step.

[22,11,20,21,23]	[**24**]	[26,29,66,27,25,28]	partition
[11,20,21] [**22**] [23]		[25] [**26**] [29,66,27,28]	partition
[**11**] [20,21]		[27,28][**29**][66]	partition
[**20**][21]		[**27**][28]	partition
[20,21]		[27,28]	merge
[11,20,21]		[27,28,29,66]	merge
[11,20,21,22,23]		[25,26,27,28,29,66]	merge
[11,20,21,22,23,24,25,26,27,28,29,66]			merge