1.

Algorithm: index(x,L) Requires: a number x and a list L Returns: the index number of x in the list, if x is one element in the list; otherwise -1.	Algorithm: linSearch(x,L) [sub-algorithm] Requires: a number x and a list L Returns: True if x occurs in L; False otherwise
1: if !(linSearch(x,L)) 2: return -1 3: elseif x==head(L) 4: return 1 5: else 6: return 1+index(x,tail(L)) 7: endif	1: if isEmpty(L) 2: return False 3: elseif x==head(L) 4: return True 5: else 6: return linSearch(x,tail(L)) 7: endif

Note: the sub-algorithm linSearch() (or binSearch) is called to check if x occurs in the list first.
In exams, if the question says, "you may call the algorithm/function …", then it means you can directly use the mentioned algorithm/function for computation without providing further detail (e.g., Q5).

2.

Algorithm: concat(L1,L2) Requires: two lists L1 and L2 Returns: one list with L1 attaching to front of L2
1: if isEmpty(L1) 2: return L2 3: else 4: return cons(head(L1),concat(tail(L1),L2)) 5: endif

3.

Algorithm: insert(x,i,L) Requires: a number x, a list L, and an index number i, $1 \leq i \leq length(L) + 1$, Returns: a list with x inserted as the i-th element
1: if i==1 2: return cons(x,L) 3: else 4: return cons(head(L),insert(x,i-1,tail(L))) 5: endif

4. Refer to Lecture 5 slides.

5.

```
Algorithm: subset(L1,L2)
Requires: two lists L1 and L2
Returns: True if L1 is a subset of L2; False otherwise
1:   if length(L1)>length(L2)
2:       return False
3:   else
4:       if isEmpty(L1)
5:           return True
6:       elseif !(linSearch(head(L1),L2))
7:           return False
8:       else
9:           return subset(tail(L1),L2)
10:      endif
11:  endif
```

6.

Target list	Comparison
[2,3,5,8,13,17,26,33,36,41]	36==2, False
[3,5,8,13,17,26,33,36,41]	36==3, False
[5,8,13,17,26,33,36,41]	36==5, False
[8,13,17,26,33,36,41]	36==8, False
[13,17,26,33,36,41]	36==13, False
[17,26,33,36,41]	36==17, False
[26,33,36,41]	36==16, False
[33,36,41]	36==33, False
[36,41]	36==36, True

9 times of comparisons.

7.

Target list	Middle element	Comparison
[2,3,5,8,13,17,26,33,36,41]	17	36>17
[26,33,36,41]	36	36==36

Use (N/2+1) to compute middle element index. 2 times of comparisons.

Target list	Middle element	Comparison
[2,3,5,8,13,17,26,33,36,41]	13	36>13
[17,26,33,36,41]	26	36>26
[33,36,41]	33	36>33
[36,41]	36	36==36

Use (N/2) to compute middle element index. 4 times of comparisons.

Note: both ways are correct.

In exams, you may use tables/diagrams to show such process more efficiently.

8. i: $O(n^3)$
   ii: $O(n^2)$
   iii: $O(n \log n)$
   iv: $O(n)$
   v: $O(\log n)$
   vi: $O(n^2 \log n)$

9. Algorithm (v). Because this function has the slowest growth rate with respect to n.

10.

Left list	Right list
[4,8,3,1,9]	[ ]
[8,3,1,9]	[4]
[3,1,9]	[4,8]
[1,9]	[3,4,8]
[9]	[1,3,4,8]
[ ]	[1,3,4,8,9]

11. Refer to Seminar 5 slides.