COMPUTER SCIENCE & INFORMATION TECHNOLOGY

Data Structures

Text Book: Theory with worked out Examples and Practice Questions
Data Structures
(Solutions for Text Book Practice Questions)

1. Arrays

01. Ans: 1010
   Sol: Loc. of A (i) = L_0 + (i-lb) * C
        Loc of A [0] = 1000+(0+5) × 2 = 1010

02. Ans: 1024 and 1024
   Sol: (i) By RMO, the loc. of
       A[i, j] = L_0 + [(i-b1) (u_2-b_2+1) +(j-b_2)]*C
       A [0, 5] = 1000 + [(0+2) × 5 + (5-3)] × 2
           = 1000+24 = 1024
   (ii) By CMO, the loc of
       A[i ,j] = L_0 +[(j-b_2) (u_1-b_1+1) +(i-b_1)]*C
       A [0, 5] = 1000 +[(5–3) × 5 + (0+2)] × 2
           = 1024

03. Ans: (a)
   Sol: In general
       RMO = L_0 + (i – 1) r_2 + (j – 1)
           = 100 + (i – 1) 15 + (j – 1)
           = 100 + 15 i – 15 + j – 1
           = 15 i + j + 84

04. Ans: (c)
   Sol: Lower triangular matrix
       \[
       \begin{bmatrix}
         \text{a} & 0 & 0 & \ldots & 0 \\
         \text{b} & \text{c} & 0 & \ldots & 0 \\
         \text{d} & \text{e} & \text{f} & 0 & 0 \\
         \text{g} & \text{h} & \text{i} & \text{j} & 0 \\
       \end{bmatrix}
       \]

05. Ans: (c)
   Sol: CMO:
   Storage:
   \[
   \begin{array}{cccccccccc}
   \text{a}_{11} & \text{a}_{21} & \text{a}_{31} & \text{a}_{41} | \text{a}_{22} & \text{a}_{32} & \text{a}_{42} | \text{a}_{33} & \text{a}_{43} | \text{a}_{44} \\
   0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9
   \end{array}
   \]
   Retrieval:
   \[
   \text{loc of } A[i,j] = L_0 + 2D + 1D
   \]
   \[
   = L_0 + [(j-1) \text{ cols} + (i-1)\text{ rows}]
   \]
   In each col., i/b = j.
   Loc. of \( A[i,j] = L_0 + [(j-1) \text{ cols} + (i-j)] \)
   In (j-1) cols
   The no. of elements is
   \[
   n + (n-1) + \ldots + (n-(j-1))
   \]
   \[
   = (j-1)n - [1 + 2 + \ldots + j - 2]
   \]
   \[
   = n(j-1) - \frac{(j-1)(j-2)}{2}
   \]
   loc. of \( A[i,j] \)
   \[
   = L_0 + \left[ n(j-1) - \frac{(j-1)(j-2)}{2} + (i-j) \right]
   \]

06. Ans: (d)
   Sol: RMO:
   Storage:
   \[
   \begin{array}{cccccccccc}
   \text{a}_{11} & \text{a}_{12} | \text{a}_{21} & \text{a}_{22} & \text{a}_{23} | \text{a}_{32} & \text{a}_{33} & \text{a}_{34} | \text{a}_{43} & \text{a}_{44} \\
   0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9
   \end{array}
   \]
Retrieval:
loc of \( A[i,j] \) = \( L_0 + 2D + 1D \)
\( = L_0 + \text{number of elements in (i-1) rows} \)
\( + (j - j/l) \)

Row    \( j/l \)
---    ---
4      3
3      2
2      1
except 1st row

\( i^{th} \) (i-1)
loc of \( A[i,j] \) = \( L_0 + [(3i - 4) + j - (i - 1)] \)
\( = L_0 + (2i + j - 3) \)

07. Ans: (a)
Sol: CMO:

Storage:
\[
\begin{array}{cccccc}
  a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\
  a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\
  a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\
  a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\
  a_{51} & a_{52} & a_{53} & a_{54} & a_{55}
\end{array}
\]

Retrieval:
loc of \( A[i,j] \) = \( L_0 + 2D + 1D \)
\( = L_0 + (j - 1) \text{cols} + (i - i/l) \)

Since i is Varying

Col    \( i/l \)
---    ---
4      3
3      2
2      1
except 1st column

\( j^{th} \) \( j-1 \)
\therefore \text{loc of } A[i,j] = L_0 + [3(j-1) - 1 + i - (j-1)]
\( = L_0 + [2j + i - 3] \)

08. Ans: (b)
Sol: Storage & Retrieval:

\[
\begin{array}{cccccc}
  a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\
  a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\
  a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\
  a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\
  a_{51} & a_{52} & a_{53} & a_{54} & a_{55}
\end{array}
\]

If \( i - j = 1 \)
loc of \( A[i,j] \) = \( L_0 + (i - i/l) \)
\( + (j - j/l) \)
i.e.,
loc of \( A[i,j] \) = \( L_0 + (n - 1) \)
\( + (i - 1) \)
or
\( (j - 1) \)

If \( i - j = 0 \)
loc of \( A[i,j] \) = \( L_0 + 2n - 1 \)
\( + (i - 1) \)
or
\( (j - 1) \)

If \( i - j = -1 \) // upper diagonal
loc of \( A[i,j] \) = \( L_0 + 2n - 1 \)
\( + (i - 1) \)
or
\( (j - 2) \)

09. Ans: (a)
Sol: A sample 5 \times 5 S-matrix is given below.

\[
\begin{array}{ccc}
  1 & 8 & 3 \\
  3 & 0 & 0 \\
  6 & 1 & 7 \\
  0 & 0 & 0 \\
  9 & 6 & 5
\end{array}
\]

The compact representation is
\[
[1,8,3,2,1, 6,1,7,4,3, 9,6,5,4,1, 3, 1] \]
10. Ans: 9
Sol: \(2n - 1 = 10 - 1 = 9\)

\[
\begin{array}{cccc}
1 & 2 & 3 & 4 \\
5 & 1 & 2 & 3 \\
6 & 5 & 1 & 2 \\
7 & 6 & 5 & 1 \\
\end{array}
\]

11. Ans: \(190900\)
Sol: \(n + (a - 1) (2n - a)\)
\[
\begin{align*}
&= 1000 + (101 - 1) (2.1000 - 101) \\
&= 1000 + 100 \times (2000 - 101) \\
&= 1000 + 100 \times 1899 \\
&= 1000 + 189900 = 190900
\end{align*}
\]

12. Ans: (a)
Sol: Square Band Matrix:
It is denoted by \(D_{n,a}\) where \(n\) is size of matrix, \(a\) is \((a-1)\)'s diagonals exists above and below the matrix diagonal.

\[
D_{5,4} = \begin{bmatrix}
11 & 12 & 13 & 14 & 15 \\
21 & 22 & 23 & 24 & 25 \\
31 & 32 & 33 & 34 & 35 \\
41 & 42 & 43 & 44 & 45 \\
51 & 52 & 53 & 54 & 55
\end{bmatrix}_{5 \times 5}
\]

Size:- \(N\) (diagonal elements)
\[
+ 2[N–1+N–2+N–3+...+N–(a–1)]
= N + 2[(a–1)N – (1+2+3+….+a–1)]
\]

Total no. of elements
\[
= n + 2[n–1 + n–2 + …… + n–(a–1)]
= n + 2 [(a–1) n – [1+2+ ……. + (a–1)]]
= n + 2 \left[ (a–1)n - \frac{a(a-1)}{2} \right]
= n+2n(a–1) - a (a–1) = n + (a–1) [2n–a]
\]

(b) \(D_{6,4}\)
In \(D_{6,4}\) → \(A\) \((5,4)\) in \(3^{rd}\) diagonal
In \(D_{6,5}\) → \(A\) \((5,4)\) in \(4^{th}\) diagonal

(i) \(i – j\) value is remained constant throughout the diagonal
(ii) As ‘a’ value changes, accordingly \(K\) value also changes.

\[
\therefore K\ value = a – (i – j)
\]

\[
\therefore \text{loc. of } A[i, j] = L_0 + \text{no. of elements in (k–1) dig} + 1D\ cross
\]
loc. of \(A[i, j] = L_0 + \text{no. of elements in (k–1) dig} + (i – i/b)\) or \((j– j/b)\)
but here \(i/b\) is varying diagonal by diagonal so prefer \(j/b\) which is always 1 in each diagonal
\[
= L_0 + \text{no. of elements in (k–1) dig} + (j–1)
\]
No. of elements in (k–1) diagonals is
\[
[\text{n–(a–1)}] + [\text{n–(a–1)+1}] + [\text{n–(a–1)+2}] + \ldots + [\text{n–a+(k–1)}]
\]
\[
\text{n–(a–1)} \rightarrow 1^{st} \text{ diagonal because it is (a–1)th diagonal}
\]
\[
(n–(a–1)+1) \rightarrow 2^{nd} \text{ diagonal}
\]
\[
(n–(a–1)+2) \rightarrow 3^{rd} \text{ diagonal}
\]
\[
n–a+(k–1) \rightarrow (k–1)^{th} \text{ term i.e. (k–1)th diagonal}
\]
\[
= (k – 1) \text{ (n–a) } + 1 + 2 + \ldots \ldots \ldots +k – 1
\]
\[
= (k – 1) \text{ (n–a) } + \frac{k(k–1)}{2}
\]

\[
\therefore \text{loc. of } A[i, j] = L_0 + \left[ (k – 1) (n–a) + \frac{k(k–1)}{2} + (j–1) \right]
\]
2. Stacks & Queues

01. (i) Ans: (a) (ii) Ans: (c)

Sol: Given array size m, say 9

Number of stacks n, say 3

\[ 0 \leq i < n \quad T[i] = B[i] = i \left[ \frac{m}{n} \right] - 1 \]

For i = 0

\[ T[0] = B[0] = 0 \left[ \frac{9}{3} \right] - 1 = 0 - 1 = -1 \]

For i = 1

\[ T[1] = B[1] = 1 \left[ \frac{9}{3} \right] - 1 = 3 - 1 = 2 \]

For i = 2


When i = 3 \Rightarrow B[3] = m - 1 = 9 - 1 = 8

(i) Push = overflow = size

(ii) POP = underflow = initial

\[
\begin{array}{cccc}
0 & 1 & 2 & 3 \\
T & 0 & 5 & 8 \\
B & 1 & 2 & 5 \\
\end{array}
\]

\[
\begin{array}{cc}
0 & 3 & 6 \\
1 & 4 & 7 \\
2 & 5 & 8 \\
\end{array}
\]

0th stack 1st stack 2nd stack

\[ \begin{array}{cccc}
T[0] = B[0] \\
\end{array} \]

overflow cases

\[ \begin{array}{cccc}
T[0] = B[0] \\
\end{array} \]

\[ \therefore T[i] = B[i + 1] \]

underflow cases

\[ \begin{array}{cccc}
T[0] = B[0] \\
\end{array} \]

\[ \therefore T[i] = B[i] \]
02. Ans: (b)
Sol: 

<table>
<thead>
<tr>
<th>Stack operation</th>
<th>Push (10)</th>
<th>Push (20)</th>
<th>Pop</th>
<th>Push (10)</th>
<th>Push (20)</th>
<th>Pop</th>
<th>Pop</th>
<th>Pop</th>
<th>Push (20)</th>
<th>Pop</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stack</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pop list</td>
<td></td>
<td></td>
<td>20</td>
<td></td>
<td></td>
<td>20</td>
<td>20</td>
<td>20</td>
<td></td>
<td>20</td>
</tr>
</tbody>
</table>

The sequence of popped out values ⇒ 20, 20, 10, 10, 20

03. Ans: (d)
Sol: 

An instance of array having two stacks is shown above. Stack 1 occupied from 1 to MAXSIZE and stack 2 occupied from MAXSIZE to 1. Above shown array is filled completely. So condition for ‘stack full’ is

Top 1 = Top 2 – 1

04. Ans: (c)
Sol: Stack S is

<table>
<thead>
<tr>
<th>S1</th>
<th>S2</th>
<th>S3</th>
<th>S4</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>–1</td>
<td>–3</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>–1</td>
<td>–3</td>
<td>–3</td>
</tr>
</tbody>
</table>

f (0) = 0

05. Ans: (b)
Sol: Stack insertion order ⇒ 1, 2, 3, 4, 5. The only possible output sequence 3, 4, 5, 2, 1

That occurs when

Push (1)
Push (2)
Push (3)
Pop (3) → 3

(∵ There is no constraint on the order of deletion operations)

Push (4)
Pop (4) → 3, 4
Push (5)
Pop (5) → 3, 4, 5
Pop (2) → 3, 4, 5, 2
Pop (1) → 3, 4, 5, 2, 1

Other remaining combinations are not possible
06. Ans: 321
Sol: invocation tail (3)
T (3) = 3
T (2) = 2
T (1) = 1
T (0) = stop
Output: 3, 2, 1

07. Ans: 1213121
Sol:

Output: 1213121

08.
Sol: (i) Ans: 41

(ii) Ans: 67
Sol: Number of calls for evaluating
f(n) = 2 × f(n+1) – 1
The total number of calls in
Fibonacci (8) = 2 f(9) – 1
= 2 × 34 – 1 = 68 – 1 = 67

(iii) Ans: 54

Additions = f(n+1) – 1
f(9) = f(10) – 1 = 55 – 1 = 54
09. 
Sol: \( \text{Ackerman}(m, n) = \begin{cases} 
    n + 1 & \text{if } m = 0 \\
    \text{Ackerman}(m-1, 1) & \text{if } n = 0 \\
    \text{Ackerman}(m-1, \text{Ackerman}(m, n-1)) & \text{otherwise} 
\end{cases} \)

(i) Ans: 9 
Sol: \( \text{Ackerman}(2, 3) = \text{Ackerman}(1, \text{Ackerman}(2, 2)) = \text{Ackerman}(1, 7) \)
\[ \begin{align*} 
    A(2, 2) &= A(1, A(2, 1)) = A(1, 5) = 7 \\
    A(2, 1) &= A(1, A(2, 0)) = A(1, 3) = 5 \\
    A(2, 0) &= A(1, 1) = 3 \\
    A(1, 1) &= A(0, A(1, 0)) = A(0, 2) = 2 + 1 = 3 \\
    A(1, 0) &= A(0, 1) = 2 \\
    A(0, 1) &= 1 + 1 = 2 \\
    A(1, 3) &= A(0, A(1, 2)) = A(0, 4) = 4 + 1 = 5 \\
    A(1, 2) &= A(0, A(1, 1)) = A(0, 3) = 3 + 1 = 4 \\
    A(1, 5) &= A(0, A(1, 4)) \\
        &= A(0, A(0, A(1, 3))) \\
        &= A(0, A(0, 5)) \\
        &= A(0, 6) = 6 + 1 = 7 \\
\end{align*} \)

\[ A(1, 7) = A(0, A(1, 6)) = A(0, 9) = 9 \]

\( \text{Acknowledgment}(2, 3) = 9 \)

(ii) Ans: 13 
Sol: \( \text{Ackerman}(2, 5) = \text{Ackerman}(1, \text{Ackerman}(2, 4)) \)
\[ \begin{align*} 
    A(2, 4) &= A(1, A(2, 3)) = A(1, A(1, 9)) \\
    A(2, 3) &= A(1, A(2, 2)) = A(1, A(1, 5)) \\
    A(2, 2) &= A(1, A(2, 1)) = A(1, 5) = 5 \\
    A(2, 1) &= A(1, A(2, 0)) = A(1, 3) = 3 \\
    A(2, 0) &= A(1, 1) = 3 \\
    A(1, 1) &= A(0, A(1, 0)) = A(0, 2) = 2 + 1 = 3 \\
    A(1, 0) &= A(0, 1) = 2 \\
    A(0, 1) &= 1 + 1 = 2 \\
    A(1, 9) &= A(0, A(1, 8)) \\
        &= A(0, A(0, A(1, 7))) \\
        &= A(0, A(0, 9)) \\
        &= A(0, 10) \\
        &= 11 \\
\end{align*} \)

10. 
Sol: (a) After \( N + 1 \) calls we have the first move. 
So after 4 calls we have the first move. 
(b) After total calls \( -1 \) calls, we have the last move. 
(c) Total moves \( 2^N - 1 = 7 \) 
(d) Total invocations \( 2^{N+1} - 1 = 2^4 - 1 = 15 \)

11. Ans: (b) 
Sol: Postfix expression \( A B C D + * F /+DE + \)
12. Ans: (a)
Sol: \( a = -b + c \times d/e + f \uparrow g \uparrow h - i \times j \)

Prefix: 
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]
\[ a = -b + c \times d/e + (\uparrow f \uparrow gh) - i \times j \]

13. Ans: (a)
Sol: Infix expression: \([a+(b\times c)]-(d\times e\times f)\)
Postfix expression: \(abc+def\)

14. Ans: (a)
Sol:

15. Ans: (c)
Sol: 10, 5, +, 60, 6, /, *, 8, –

16. Ans: (c)
Sol: (i) ab
(ii) b
(iii) byz
(iv) yz
Output is \( yz \)

17. Ans: (b)
Sol: 

18. Ans: (i) 322 and (ii) 326
Sol:
Until first ‘0’ is encountered, stack contains

\[
\begin{array}{c|c}
9 & \\
6 & \\
5 & \\
\end{array}
\]

So 5 + 6 + 9 = 20 is enqueued in Q \_2 @ loc 326

Until second ‘0’ is encountered, stack contains

\[
\begin{array}{c|c}
7 & \\
5 & \\
\end{array}
\]

So 5 + 7 = 12 is enqueued is Q \_2 @ loc 328

Then simply 2 and 6 are pushed in stack

\[
\begin{array}{c|c|c}
322 & 324 & \\
6 & 2 & \\
\end{array}
\]

So the location of 6 and 20 are 322 and 326

19. Ans (c)
Sol: Suppose that array contains

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td>c</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

First configuration:

Delete element

\[
\begin{array}{c|c|c|c|c|c}
1 & 2 & 3 & 4 & 5 & 6 \\
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td>x</td>
<td>y</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
\end{array}
\]

 enqueue (x) and enqueue (y):

\[
\begin{array}{c|c|c|c|c|c}
1 & 2 & 3 & 4 & 5 & 6 \\
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td>x</td>
<td>y</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
\end{array}
\]

20. Ans: (b)
Sol: The given recursive procedure simply reverses the order of elements in the queue. Because in every invocation the deleted element is stored in ‘i’ and when the queue becomes empty.

Then the insert ( ) function call will be executed from the very last invoked function call. So, the last deleted element will be inserted first and the procedure goes on

## 3. Linked Lists

01. Ans: (d)
Sol:

\[
\begin{array}{c|c|c|c|c|c|c}
\text{a} & \text{b} & \text{c} & \text{d} & \text{e} & \text{f} & \text{g} \\
\end{array}
\]

Print = d

02. Ans: (d)
Sol:

\[
\begin{array}{c|c|c|c|c|c|c}
\text{a} & \text{b} & \text{c} & \text{d} & \text{e} & \text{f} & \text{g} \\
\end{array}
\]

Print = d

03. Ans: (a)
Sol: while (P) or while (P!= Null)
while P is pointing to somebody
04. Ans: (d)  
Sol: Recursive routine for ‘Count’

```
10
9
8
7
6
5
4
3
2
1
0
```

No. of nodes = 4

05. Ans: (b)  
Sol: either causes a null pointer dereference or append list m to the end of list n.

06. Ans: (b)  
Sol:  
Before

```
10
9
8
7
6
5
4
3
2
1
0
```

After

```
10
9
8
7
6
5
4
3
2
1
0
```

07. Ans: (b)  
Sol: This is recursive routine for reversing a SLL.

08. Ans: (a)  
Sol:  
Before

```
10
9
8
7
6
5
4
3
2
1
0
```

After

```
10
9
8
7
6
5
4
3
2
1
0
```

concatenation of two single linked lists by choosing alternative nodes.

09. Ans: (d)  
Sol:  
Before

```
10
9
8
7
6
5
4
3
2
1
0
```

After

```
10
9
8
7
6
5
4
3
2
1
0
```

10. Ans: (a)  
Sol: Linked stack push () = insert front ()
11. **Sol:**

<table>
<thead>
<tr>
<th>Operation</th>
<th>Left most</th>
<th>Right most</th>
<th>Middle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insert</td>
<td>3</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Delete</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

12. **Ans:** (b)

**Sol:** Inserts to the left of middle node in doubly linked list.

13. **Ans:** (a)

**Sol:** Before reverse:

![Before reverse diagram]

After reverse:

![After reverse diagram]

### 4. Trees

01. **Ans:** (d)

**Sol:**
1. Traverse the left subtree in postorder.
2. Traverse the right subtree in postorder.
3. Process the root node

02. **Ans:** (e)

**Sol:**
1. Traverse the right subtree in postorder.
2. Traverse the left subtree in postorder.
3. Process the root node

03. **Ans:** (c)

**Sol:**

![Tree diagram]

$A(t) = a \ b \ c \ d \ f \ g \ e$
04. Ans: (b)
Sol:

\[
\begin{align*}
B(t) &= b \quad d \quad c \quad a \quad e \quad f \quad g \\
\end{align*}
\]

05. Ans: 5
Sol:

\[
\begin{align*}
\therefore \text{Totally 5 distinct trees possible} \\
\text{Note: The number of binary trees can be formulated with unlabeled nodes are} \\
\frac{2^n C_n}{n + 1}.
\end{align*}
\]

06. Ans: (c)
Sol: Preorder : A B C D E F G
In-order : B D C A F G E
Post-order: D C B G F E A

07. Ans: 3
Sol: Note: If pre-order is given, along with terminal node information & all right child information the unique pattern can be found. If post-order is given along with terminal information and all left child information the unique pattern can be identified.
08. Ans: 3
Sol:
```
   a
  / \
 b   c
/ \ / \  
d e f g
```

09. Ans: 4
Sol: Post 8 9 6 7 4 5 2 3 1
In 8 6 9 4 7 2 5 1 3

10. (a) Ans: 19
Sol: Leaf nodes (L) = Total nodes – internal nodes
L = In+1-I
L = I(n-1)+1
L = 20
I = ?
20 = I(2 – 1) +1
20 = I + 1
I = 19

10. (b) Ans: 199
Sol: L = I(n-1)+1
L = 200
200 = I + 1
I = 199

11. Ans: (b)
Sol:
```
L = 0
  /  
 L= 1  
     /  
   L= 2  
   /    
  L= 3  
 /    
L= 4  
```
Minimum = 3, Maximum = 14
12. Ans: 2 & 1
Sol:

13. Ans: (a)
Sol:

Before Swap

After Swap

14. Ans: (d)
Sol:

Parent of f, g, h is e. i.e. internal parenthesis has children of parent which is out of parenthesis.

15. Ans: 6
Sol: a (b, c (e (f, g, h)), d)
16. Ans: 4
Sol: Given are 3 trees

To get the converted binary tree of these given trees
→ parent is not given we have to assume virtual parent
Among siblings
- Keep the leftmost as it is,
- Cut and connect right siblings as shown in diagram
19. Ans: 6
Sol: Expanded as
$$((1+1) - (0 - 1)) + ((1 - 0) + (1+1))$$
$$= 3 + 3 = 6$$

20. Ans: -2
Sol: $$(0 + 0) - (1 - 0) + (0 - 1) + (0 + 0)$$
$$= -1 + (-1) = -2$$

21. Ans: 4
Sol:
```
40
  20
   10
  30
  50
  70
```

22. Ans: (b)
Sol: Preorder = 12, 8, 6, 2, 7, 9, 10, 16, 15, 19, 17, 20
Inorder = 2, 6, 7, 8, 9, 10, 12, 15, 16, 17, 19, 20
```
2, 6, 7
  8
5, 10
```
```
15, 16, 17, 19, 20
```

23. Ans: 4
Sol:
```
SUN
MON TUE THU FRI
SEX
```

Sol: 71, 65, 84, 69, 67, 83 insert into empty binary search tree
```
71
  65
  84
69
67
  83
```

25. Ans: 30

$$= 2, 7, 6, 10, 9, 8, 15, 17, 20, 19, 16, 12$$

.: Element in the lowest level is 67
26. Ans: (d)  
Sol: (a) 5 3 1 2 4 7 8 6  
(b) 5 3 1 2 6 4 8 7  
(c) 5 3 2 4 1 6 7 8  
(d) 5 3 1 2 4 7 6 8

27. Ans: 15  
Sol: 1. Jump right  
2. Go on descend left

28. Ans: 88  
Sol: 
\[
N(H) = \begin{cases} 
1 & \text{if } H = 0 \\
2 & \text{if } L = H = 1 \\
1 + N(H - 1) + N(H - 2) & \text{otherwise} 
\end{cases}
\]

\[
N(2) = 1 + N(1) + N(0) = 1 + 2 + 1 = 4 \\
N(3) = 1 + N(2) + N(1) = 1 + 4 + 2 = 7 \\
N(8) = 1 + N(7) + N(6) = 1 + 54 + 33 = 88
\]
29. Ans: 14
Sol: 21, 26, 30, 9, 4, 14, 28, 18, 15, 10, 2, 3, 7
30. Ans: 28
Sol:
Delete 2, 3

01. Ans: (b)
Sol: $V_8$ is pushed in for two times
Output: 1, 2, 4, 6, 3, 7  step back

02. Ans: 2
Sol: Sequence of exploration
V₅→V₂→V₁→V₃→V₆→V₈→V₇→V₄
Sequence of stack contents
Not pushed vertices are → V₄, V₇
Vertices are pushed in more than once
→ V₁, V₂, V₃, V₆, V₅

03. Ans: 3
Sol: Sequence of exploration
V₈→V₄→V₂→V₁→V₃→V₆→V₇→V₅
Sequence of stack contents
Not pushed vertices are → V₆, V₇, V₅
Vertices are not pushed in more than once
→ V₁, V₄, V₈

04. Ans: (a)
Sol: (a) invalid  (b) valid
(c) valid  (d) valid

05. Ans: (c)
Sol: (a) valid  (b) valid
(c) invalid  (d) valid
06. Ans: 19
Sol:

Maximum possible recursion depth = 19
(The dashed link ‘nodes’ are explored while stepping backward.)

07. Ans: 8
Sol:

Minimum possible recursion depth = 8
(The dashed link ‘nodes’ are explored while stepping backward.)

08. Ans: (d)
Sol:

09. Ans: (d)

10. Ans: (d)
6. Hashing

01. Ans: (d)
Sol:  
\[
\begin{array}{c}
1 \\
\end{array}
\quad \text{to} \quad \begin{array}{c}
1000 \\
\end{array}
\]

02. Ans: (a)
Sol: 
\[
\begin{array}{c|c|c|c}
0 & 28 & 14 & 10 \\
1 & 20 & 8 & 3 \\
2 & 12 & & \\
3 & & & \\
4 & & & \\
5 & 5 & 15 & 33 \\
6 & & & \\
7 & & & \\
8 & 17 & & \\
\end{array}
\]

\[
\frac{3+1+1+1+2+1}{9} = 1 \text{ (average)}
\]

03. Ans: 80
Sol: 
\[
\text{Elements} = 2000
\]
\[
\text{Load factor} = \frac{\text{elements}}{\text{slots}}
\]
\[
= \frac{2000}{25} = 80
\]

04. Ans: (b)
Sol: 
Hash function
\[
h (x) = (3x + 4) \mod 7
\]
\[
h (1) = (3+4) \mod 7 = 0
\]
\[
h (3) = (9 + 4) \mod 7 = 6
\]
\[
h (8) = (24 +4) \mod 7 = 0
\]
\[
h (10) = (30 +4) \mod 7 = 6
\]
Assume Linear probing for collision resolution
The table will be like

05. Ans: (d)
Sol: After inserting all keys, the hash table is

<table>
<thead>
<tr>
<th>Key</th>
<th>Loc</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>43</td>
<td>10</td>
<td>8</td>
<td>36</td>
<td>92</td>
<td>87</td>
<td>11</td>
<td>4</td>
<td>71</td>
</tr>
<tr>
<td>36</td>
<td>3</td>
<td>4</td>
<td>10</td>
<td>0</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>92</td>
<td>11</td>
<td>13</td>
<td>36</td>
<td>2</td>
<td>4</td>
<td>71</td>
<td>14</td>
<td>43</td>
</tr>
<tr>
<td>87</td>
<td>11</td>
<td>13</td>
<td>36</td>
<td>2</td>
<td>4</td>
<td>71</td>
<td>14</td>
<td>43</td>
</tr>
</tbody>
</table>

Last element is stored at the position 7

06. Ans: (c)
Sol: Resultant hash table.
In linear probing, we search hash table sequentially starting from the original location. If a location is occupied, we check the next location. We wrap around from the last table location to the first table location if necessary.
07. Ans: (c)
Sol:

```
×  ×  ✓  ×
A  B  C  D
0
1
2 42 42 42 42
3 52 23 23 23
4 34 34 34 23
5 23 52 52 34
6 46 33 46 46
7 33 46 33 52
8
9
```

08. Ans: (c)
Sol: Case (I): To store 52

<table>
<thead>
<tr>
<th>Variable part</th>
<th>Fixed part</th>
</tr>
</thead>
<tbody>
<tr>
<td>42 23 34</td>
<td>52 46 33</td>
</tr>
<tr>
<td>42 34 23</td>
<td>52 46 33</td>
</tr>
<tr>
<td>23 42 34</td>
<td>52 46 33</td>
</tr>
<tr>
<td>23 34 42</td>
<td>52 46 33</td>
</tr>
<tr>
<td>34 42 23</td>
<td>52 46 33</td>
</tr>
<tr>
<td>34 23 42</td>
<td>52 46 33</td>
</tr>
</tbody>
</table>

3! = 6

Case (II): To store 33

```
<table>
<thead>
<tr>
<th>Variable part</th>
<th>Fixed part</th>
</tr>
</thead>
<tbody>
<tr>
<td>42 23 34</td>
<td>52 46 33</td>
</tr>
<tr>
<td>23 42 34</td>
<td>52 46 33</td>
</tr>
</tbody>
</table>
```

3! = 24

Since 46 is not getting collided with any other key, it can be moved to the variable part.

Case (I) & Case (II) are mutually exclusive

Case (I) + Case (II) = 24 + 6 = 30

Total 30 different insertion sequences.