Pratibh
CSDS-B

Date: / /
Page No.

Data Structures 2019-20

Section-A

1

(a) We can represent it either by array representation or linked list representation

(b) → Traversal
→ Insertion
→ Deletion
→ Search

(c) Memory management
Expression evaluation & system parsing

(d) Tail recursion is using the recursive function as the last statement of the function, which means when nothing is left to do after coming back from the recursive call, it is tail recursion.

(e) In Priority queue each element has a priority associated with it & served according to its priority. Elements with higher priority are served before elements with lower priority

Application :-
→ CPU scheduling in OS.
→ Dijkstra's shortest path algorithm in graph theory
→ Heap sort algorithm

(f) It repeatedly steps through the list to be sorted, compares each pair of adjacent items and swaps them if they are in the wrong order.

(g) It is a subset of the edges of a connected, undirected graph that connects all the vertices together, without any cycles and with minimum possible total edge weight.

Applications :-
→ Network design
→ Cluster analysis
→ Approximation Algo
→ Image segmentation
→ Image processing

(h) Adjacency matrix is a 2-D matrix of size n x n, where n is the no. of vertices of graph
Each cell in matrix represents an edge between two vertices.
Time complexity - O(1)
Space complexity - $O(n^2)$

Adjacency list is a collection of lists, one for each vertex in graph
Each list contains neighbours of the vertex
Time complexity - O(n)
Space complexity - O(n+e)

(h) Extended binary tree includes an additional element called sentinel which is used as a placeholder for certain operations.

In Strictly Binary tree each node can have either two children or more

In Full binary tree every node has either 0 or 2 children

In complete Binary tree every level is completely filled and all the leaves are left-justified.

(j) It is a binary tree in which each node contains a link called a thread that allows for easy traversal of the tree without need for a stack or recursion.

Section-B

2

(a) Merits of arrays :-

→ In array, accessing of elements is easy by using index number.

→ 2D array is used to represent Matrices.

→ It can be used to store multiple values of similar type.

Demerits :-

→ Size is fixed

→ Insertion & deletion is not easy

→ Allocating more memory than requirement leads to wastage of memory space.

```
#Include <stdio.h>
# Include <stdlib.h>
int main(){
    int arr1[] = {1, 4, 6, 8, 9};
    int arr2[] = {2, 3, 5, 7, 10};
    int len = sizeof(arr2)/sizeof(arr3[0]) + sizeof(arr2)/sizeof(arr2[0])
    int arr3[len];
for(int i=0; j=0, k=0; k<len;){
    if(arr1[i] < arr2[j])
        arr3[k++] = arr1[j++];
    else
        arr3[k++] = arr2[j++];
```

```
printf("Sorted array : ");
for(int k=0; k<len; k++) {
    printf("%d", arr3[k]);
}
return 0;
}
```

(c)

Linear search	Binary search
• It is based on sequential approach	• It is based on divide and conquer approach.
• It is used for small size data sets.	• It is used for large size data sets.
• It can be implemented both on single & multidimensional array.	• It can be implemented only on a multidimensional array.
• Elements don't need to be sorted for linear search.	• Elements must be sorted.
• It is less efficient	• It is more efficient

11, 22, 80, 33, 40, 44, 55, 60, 66, 77, 80, 88, 99

$n = 13$

$s = 11$

$e = 99$

$mid = \frac{99+11}{2} = 55$

$40 < 55$

$e = mid - 1 = 55 - 1 = 54$

$40 < 54$

$e = mid - 1 = 53$

$40 < 53$

$e = 53 - 1 = 52$

$40 < 52$

$e = 52 - 1 = 50$

49

48

47

46

45

44

$40 < 44$

$e = mid - 1 = 43$

42

41

$e = 41 - 1 = 40$

$\boxed{e = 40}$

Section-C

3
(B) Time complexity :-
It is the measure of amount of time an algorithm takes to run as a function of the size of input.

Space complexity :-
It is a measure of amount of memory an algorithm takes up as a function of size of input

Big O notation :-
It is used to express the upper bound of the running time of an algorithm. It expresses the running time as a function of the input size

Asymptotic notation.
It is used to express the behaviour of a function as the input size approaches infinity. The most commonly used asymptotic notation are big O, big omega & big theta.

4.(a)

(i)

Iteration	Recursion
• In this the repeated execution of set of instruction.	• It is a process of calling a function itself.
• The format of iteration includes initialization, condition & increment/decrement of variable.	• This is a termination condition is specified.
• It is applied to loops.	• It is applied to functions
• It is faster than recursion	• It is slower.
• Its time complexity is lower	• It time complexity is higher.
• It uses less memory as compared to recursion	• It uses more memory.

(ii)

```
void towerofhanoi(int n, char from_rod, char to_rod,
                  char aux_rod)
{
  if(n==0){
    return;
  }
  towerofhanoi(n-1, from_rod, aux_rod, to_rod);
  cout<<"move disk "<<n<<" from rod "<<from_rod<<" to rod "<<
      to_rod<<endl;
  towerofRanoi(n-1, aux_rod, to_rod, from_rod);
}
```

7

(b) A B-Tree is a type of data structure in which each node has at most two children, which are referred to as the left child & the right child. The topmost node in the tree is called the root node. Each node in a binary tree has a value & the structure of the tree is defined by a set of rules that determine how values of the nodes are related to one another.