Java Assignment-2

1) What do you mean by a Data structure?

- Data structures is a way of organizing and storing the data in a computer so that it can be accessed and modified efficiently. The main idea is to reduce the space and time complexities of different tasks.

2) What are some of the applications of DS?

- Following are common DS with their respective applications:

Arrays: Execution of matrices and vectors, Dynamic memory allocation, Pointer container, Control tables.
Stack: Evaluation of expressions, Backtracking, Runtime memory management, Arrangement of books in a library.
Queue: Here, the data sent need not be received at the same rate at which it was sent. A certain system resource is to be shared between different processes.
Linked-List: Representation of sparse matrices, Non-contiguous data storage, Implementation of non-binary tree or other data structures, Dynamic memory management, Equalizing parenthesis, Symbol tables.
Graph: Computer networking, Problem solutions involving 'Depth-First' search or 'Breadth-First' search algorithms, Representation of matrices, Study of molecular interactions in Chemistry.
Tree: Representation of data lists, Quickly accessible data storage, Representation of hierarchal data, Routing of algorithms.
Hash-table: Unique data representation, Implementation of caches, Array association, Locating entries in a table, Representation of objects, Database indexing.

3) What are the advantages of a Linked list over an array?

- Size of the list doesn't need to be mentioned at the beginning of the program, certainly dynamic memory allocation and deallocation.
- As the linked list doesn't have a size limit, we can go on adding new nodes (elements) and increasing the size of the list to any extent.
- Linked Lists are faster in operating times, and since they are assigned memory dynamically, they are good to manage memory space.
- Insertion/deletion of an element at beginning in a linked list is O (1) operation while in array it is O(n).

4) **Write the syntax in C to create a node in the singly linked list.**

```
#include <stdio.h>
#include <stdlib.h>
struct node
{
int data;
struct node *next;
};
int main ()
{
   struct node *prev, *head, *p;
   return 0;
}
```

5) **What is the use of a doubly-linked list when compared to that of a singly linked list?**

- Doubly linked list allows element two-way traversal, whereas Singly linked list allows traversal elements only in one way.
- Singly linked list is generally used for implementation of stacks, but doubly linked list can be used to implement stacks as well as heaps and binary trees.
- In doubly linked list, the traversal can be done using the previous node link or the next node link. In singly linked list, the traversal can be done using the next node link only.
- A doubly linked list contains an extra pointer, typically called *previous pointer*, together with next pointer and data which are there in singly linked list.

6) **What is the difference between an Array and Stack?**

Stack:

- Stack is an ordered collection of items.
- Stack is a dynamic object whose size is constantly changing as items are pushed and popped.
- Stack may contain different data types.

Array:

- Array is an ordered collection of items.
- Array is a static object i.e. no of item is fixed and is assigned by the declaration of the array.
- It contains same data types.

- Although both are the most efficient ways for storing and accessing data and you can certainly implement a stack with an array with the exception of working principle and access control. A stack is a basic representation of collection of items in a data structure where the items are arranged in a particular order so that they can be inserted and removed from one end only, which is from the top of the stack in a LIFO or FILO order.

An array is a static object where the number of items is fixed and unlike stacks, items in an array can be added and removed from either end regardless of the order.

7) What are the minimum number of Queues needed to implement the priority queue?

- **TWO are required.**
- One queue is used for the actual storing of data, and the other one is used for storing the priorities.

8) What are the different types of traversal techniques in a tree?

- **Depth First Traversal**: Traverse depth wise. It includes these three-

- **Inorder:** In case of binary search trees, Inorder traversal gives nodes in non- decreasing order. It follows following pattern while traversing (Left child, Root, Right Child). It will first go to the left most of sub tree and then to root node of sub tree and then to right child of sub tree.
- **Preorder:** Preorder traversal is used to create a copy of the tree. Preorder traversal is also used to get prefix expression on of an expression tree. It follows following pattern while traversing (Root, Left child, Right Child). It will first give root node than go to the left of sub tree and then to right child of sub tree.
- **Postorder:** Postorder traversal is used to delete the tree. Postorder traversal is also useful to get the postfix expression of an expression tree. It follows following pattern while traversing (Left child, Right Child, Root). It will first go to the left most of sub tree and then to right child of sub tree than to root node of sub tree.

- **Breath First Traversal*:*** Traverse breadth wise.

9) Why it is said that searching a node in a binary search tree is efficient than that of a simple binary tree?

- BINARY TREE is unordered, whereas in BINARY SEARCH TREE the left subtree has elements less than the nodes element and the right subtree has elements greater than the nodes element which makes the searching of an particular node will be easy.

10) What are the applications of Graph DS?

Graph is powerful and versatile data structure that easily allow to you represent real life relationships between different type of data nodes.

- **Facebook**: Each user is represented as a vertex and two people are friends when there is an edge between two vertices. Similarly, friend suggestion also uses graph theory concept.

- **Google Maps**: Various locations are represented as vertices and the roads are represented as edges and graph theory is used to find shortest path between two nodes.
- **Recommendations on e-commerce websites**: The "Recommendations for you" section on various e-commerce websites uses graph theory to recommend items of similar type to user's choice.

11) Can we apply Binary search algorithm to a sorted Linked list?

- NO, we can't apply binary search algorithm directly on linked list because here memory is not contiguous, so middle element can't be found in o (1). we can use two pointers for finding the middle element and then apply binary search on it. But the complexity will not be o (log n) because we can't find middle element in o (1).

12) **When can you tell that a Memory Leak will occur?**

- The memory leak occurs, when a piece of memory which was previously allocated by the programmer. Then it is not deallocated properly by programmer. That memory is no longer in use by the program. So that place is reserved for no reason. That's why this is called the memory leak.

13) How will you check if a given Binary Tree is a Binary Search Tree or not?

Binary Search Trees follow the following properties: -

- All nodes in the left subtree of a node have values less than the node's value.
- All nodes in the right subtree of a node have values greater than the node's value.
- Both left and right subtrees are also binary search trees.

14) Which data structure is ideal to perform recursion operation and why?

- **Stack**, if you call a function or method recursively, the compiler uses the call stack to save each iteration variables (the state of the iteration) so the code can continue from where it stopped when called the recursive method.

15) What are some of the most important applications of a Stack?

The various applications of stack are:

- Expression Evaluation like infix to postfix.
- Reverse String.
- Conversation of decimal to binary numbers.
- Parsing
- Simulation of recursion
- Function call

16) Convert the below given expression to its equivalent Prefix and Postfix notations.

Steps to convert infix expression to prefix

- First, reverse the given infix expression.
- Scan the characters one by one.
- If the character is an operand, copy it to the prefix notation output.
- If the character is a closing parenthesis, then push it to the stack.
- If the character is an opening parenthesis, pop the elements in the stack until we find the corresponding closing parenthesis.
- If the character scanned is an operator
- If the operator has precedence greater than or equal to the top of the stack, push the operator to the stack.
- If the operator has precedence lesser than the top of the stack, pop the operator and output it to the prefix notation output and then check the above condition again with the new top of the stack.
- After all the characters are scanned, reverse the prefix notation output.

25) How to find the shortest path between two vertices?

Dijkstra's Algorithm:
The most common solution for this problem is Dijkstra's algorithm which updates the shortest path between the current node and all of its neighbors.

After updating the distance of all of the neighbors it moves to the node with the lowest distance and repeats the process with all unvisited neighbors. This process continues until the entire graph has been visited.

Step 1:
Graph needs to be setup so that we can record the required values. On any edge we have the distance between the two nodes it connects. On any node we have its shortest distance from the starting node.Let's set the value on every node to positive infinity, and set the value on the starting node to zero.

Step 2:

Look at all nodes directly adjacent to the starting node. The values carried by the edges connecting the start and these adjacent nodes are the shortest distances to each respective node.

Record these distances on the node - overwriting infinity - and also cross off the nodes, meaning that their shortest path has been found.

Step 3:
Select one of the nodes which has had its shortest path calculated, we'll call this our pivot. Look at the nodes adjacent to it (we'll call these our destination nodes) and the distances separating them.

For every destination node:

If the value in the pivot plus the edge value connecting it totals less than the destination node's value, then update its value, as a new shorter path has been found. If all routes to this destination node have been explored, it can be crossed off.
Step 4:
Repeat step 2 until all nodes have been crossed off. We now have a graph where the values held in any node will be the shortest distance to it from the start

------ END -------