INTERVIEW QUESTIONS

DSA

JAVA EDITION

EXPLAIN THE DIFFERENCE BETWEEN AN ARRAY AND A LINKED LIST.

- ARRAY IS A DATA STRUCTURE THAT STORES A FIXED-SIZE SEQUENTIAL COLLECTION OF ELEMENTS OF THE SAME TYPE.
- IT ALLOWS FOR INDEX-BASED ACCESS FOR ELEMENTS, WHERE EACH ELEMENT IS IDENTIFIED BY ITS INDEX IN THE ARRAY.
- LINKED LIST IS A DATA STRUCTURE THAT CONSISTS OF A SEQUENCE OF NODES, WHERE EACH NODE CONTAINS A REFERENCE TO THE NEXT NODE IN THE SEQUENCE.
- LINKED LISTS ARE USUALLY PREFERRED OVER ARRAYS WHEN DEALING WITH LARGE AMOUNTS OF DATA AS IT CAN GROW OR SHRINK DYNAMICALLY, WHILE ARRAYS HAVE A FIXED SIZE.
- LINKED LISTS ALSO ALLOW FOR FASTER INSERTION AND DELETION OF ELEMENTS COMPARED TO ARRAYS.

DESCRIBE THE PROCESS OF IMPLEMENTING A STACK USING AN ARRAY.

- TO IMPLEMENT A STACK USING AN ARRAY, YOU WOULD NEED TO KEEP TRACK OF THE TOP OF THE STACK USING A VARIABLE, AND PUSH AND POP ELEMENTS BY INCREMENTING OR DECREMENTING THIS VARIABLE.
- WHEN PUSHING AN ELEMENT, YOU WOULD ADD IT TO THE ARRAY AT THE CURRENT TOP OF THE STACK AND INCREMENT THE TOP VARIABLE.
- WHEN POPPING AN ELEMENT, YOU WOULD REMOVE THE ELEMENT FROM THE ARRAY AT THE CURRENT TOP OF THE STACK AND DECREMENT THE TOP VARIABLE.

HOW WOULD YOU IMPLEMENT A BINARY SEARCH TREE IN JAVA?

- TO IMPLEMENT A BINARY SEARCH TREE IN JAVA, YOU WOULD NEED TO CREATE A NODE CLASS THAT HAS A LEFT AND RIGHT CHILD, AND A VALUE.
- YOU WOULD THEN NEED TO CREATE A BST CLASS THAT HAS A ROOT NODE AND METHODS FOR INSERTING, SEARCHING AND DELETING NODES.
- WHEN INSERTING A NEW NODE, YOU WOULD COMPARE ITS VALUE TO THE VALUE OF THE CURRENT NODE AND RECURSIVELY CALL THE INSERT METHOD ON THE LEFT OR RIGHT CHILD ACCORDINGLY.

WHAT IS THE TIME AND SPACE COMPLEXITY OF BUBBLE SORT? HOW CAN IT BE IMPROVED?

- THE TIME COMPLEXITY OF BUBBLE SORT IS O(N^2) AS IT COMPARES EACH ELEMENT WITH EVERY OTHER ELEMENT.
- THE SPACE COMPLEXITY IS O(1) AS IT ONLY REQUIRES ONE ADDITIONAL MEMORY FOR SWAPPING ELEMENTS.
- TO IMPROVE BUBBLE SORT, WE CAN USE THE TECHNIQUE CALLED "COCKTAIL SORT" WHICH IS A BIDIRECTIONAL VERSION OF BUBBLE SORT.

HOW CAN YOU REVERSE A SINGLY LINKED LIST IN JAVA?

- TO REVERSE A SINGLY LINKED LIST IN JAVA, YOU WOULD NEED TO ITERATE THROUGH THE LIST AND FOR EACH NODE, CHANGE THE NEXT POINTER TO POINT TO THE PREVIOUS NODE.
- YOU WOULD ALSO NEED TO KEEP TRACK OF THE PREVIOUS NODE AS YOU ITERATE THROUGH THE LIST.

DESCRIBE THE USE AND IMPLEMENTATION OF A HASH TABLE IN JAVA.

- A HASH TABLE IS A DATA STRUCTURE THAT USES A HASH FUNCTION TO MAP KEYS TO THEIR VALUES. IN JAVA, A HASH TABLE CAN BE IMPLEMENTED USING THE HASHMAP CLASS.
- THE HASHMAP CLASS USES THE HASHCODE() METHOD TO CALCULATE THE INDEX OF THE ARRAY, WHERE THE VALUE SHOULD BE STORED AND THE EQUALS() METHOD TO CHECK FOR EQUAL KEYS.

HOW CAN YOU IMPLEMENT A DEPTH-FIRST SEARCH ALGORITHM IN JAVA?

- TO IMPLEMENT A DEPTH-FIRST SEARCH ALGORITHM IN JAVA, YOU WOULD NEED TO USE RECURSION. YOU WOULD START AT THE ROOT NODE AND RECURSIVELY CALL THE DEPTH-FIRST SEARCH METHOD ON EACH CHILD NODE UNTIL YOU REACH A LEAF NODE.

EXPLAIN THE DIFFERENCE BETWEEN A STATIC AND DYNAMIC ARRAY IN JAVA.

- A STATIC ARRAY HAS A FIXED SIZE THAT IS DETERMINED AT THE TIME OF CREATION, WHEREAS A DYNAMIC ARRAY CAN GROW OR SHRINK IN SIZE AS ELEMENTS ARE ADDED OR REMOVED.
- IN JAVA, A STATIC ARRAY CAN BE CREATED USING THE NEW KEYWORD AND THE SIZE OF THE ARRAY, WHEREAS A DYNAMIC ARRAY CAN BE IMPLEMENTED USING THE ARRAYLIST CLASS.

WHAT IS THE TIME AND SPACE COMPLEXITY OF QUICKSORT? HOW CAN IT BE IMPROVED?

- THE TIME COMPLEXITY OF QUICKSORT IS O(N*LOG(N)) ON AVERAGE AND O(N^2) IN THE WORST CASE.
- THE SPACE COMPLEXITY IS O(LOG(N)) ON AVERAGE AND O(N) IN THE WORST CASE.
- IT CAN BE IMPROVED BY USING "MEDIAN OF THREE" PIVOT SELECTION TECHNIQUE AND RANDOM PIVOT SELECTION TECHNIQUE.

DESCRIBE THE USE AND IMPLEMENTATION OF A PRIORITY QUEUE IN JAVA.

- A PRIORITY QUEUE IS A DATA STRUCTURE THAT STORES ELEMENTS IN ORDER OF PRIORITY, SO THAT THE ELEMENT WITH THE HIGHEST PRIORITY IS ALWAYS AT THE FRONT.
- IN JAVA, A PRIORITY QUEUE CAN BE IMPLEMENTED USING THE PRIORITYQUEUE CLASS.
- THIS CLASS STORES THE ELEMENTS IN A PRIORITY ORDER, BASED ON THEIR NATURAL ORDERING OR BASED ON A PROVIDED COMPARATOR.