***Sai Kiran Varma Pingili.***

[***Saikiranvarma1212@gmail.com***](mailto:Saikiranvarma1212@gmail.com)

***+91 6304 586 369***

**Global Trend Programming Profile Assessment Questions**

Question No: 4.

public String serialize (TreeNode root) {

if (root == null) return "";

Deque<TreeNode> dq = new LinkedList<> ();

StringBuilder sb = new StringBuilder ();

dq.offer(root);

while(!dq.isEmpty()) {

TreeNode node = dq.poll();

if(node == null) {

sb.append("#,");

}

else {

sb.append(node.val).append(",");

dq.offer(node.left);

dq.offer(node.right);

}

}

return sb.toString();

}

public TreeNode deserialize (String data) {

if(data == "") return null;

Deque<TreeNode> dq = new ArrayDeque<>();

StringBuilder sb = new StringBuilder(data);

int comma = sb.indexOf(",");

String str = sb.substring(0, comma);

sb.delete(0, comma + 1);

TreeNode root = new TreeNode(Integer.parseInt(str));

dq.offer(root);

while(!dq.isEmpty()) {

TreeNode node = dq.poll();

comma = sb.indexOf(",");

str = sb.substring(0, comma);

sb.delete(0, comma+1);

if (!str.equals("#")) {

TreeNode leftNode = new TreeNode(Integer.parseInt(str));

node.left = leftNode;

dq.offer(leftNode);

}

comma = sb.indexOf(",");

str = sb.substring(0, comma);

sb.delete(0, comma + 1);

if (!str.equals("#")) {

TreeNode rightNode = new TreeNode(Integer.parseInt(str));

node.right = rightNode;

dq.offer(rightNode);

}

}

return root;

}

Question No: 6.

public boolean isValidBrackets(String s) {

if(s.length() == 0)

return true;

if(s.length() %2 != 0)

return false;

Stack <Character> stack = new Stack<>();

for(int i=0; i<s.length(); i++) {

char c = s.charAt(i);

if(stack.isEmpty() && (c == ')' || c == '}' || c == ']'))

return false;

if(c == '(' || c == '{' || c == '[')

stack.push(c);

else if((c == ')' && stack.peek() == '(') || (c == ']' && stack.peek() == '[') || (c == '}' && stack.peek() == '{'))

stack.pop();

else

return false;

}

return stack.isEmpty();

}

Question No: 7.

public static int maxArea(int[] height) {

int maxArea = 0;

int left = 0;

int right = height.length - 1;

while (left < right) {

// Calculate the area

int width = right - left;

int currHeight = Math.min(height[left], height[right]);

int currArea = width \* currHeight;

// Update maxArea if the current area is greater

maxArea = Math.max(maxArea, currArea);

// Move the pointer pointing to the shorter line

if (height[left] < height[right]) {

left++;

} else {

right--;

}

}

return maxArea;

}

Question No: 8.

public static int findKLargestNumber(int[] nums, int k) {

PriorityQueue<Integer> pq = new PriorityQueue<>(k);

for (int i = 0; i < k; i++) {

pq.add(nums[i]);

}

for (int i = k; i < nums.length; i++) {

if (nums[i] > pq.peek()) {

pq.poll();

pq.add(nums[i]);

}

}

return pq.peek();

}

Question No: 10.

public static boolean isPalindrome(String str) {

if (str == null || str.isEmpty()) {

return false;

}

int end = str.length() - 1;

String s = str.toLowerCase();

int start = 0;

while (start < end) {

if (!isValid(s.charAt(start))) {

start++;

continue;

}

if (!isValid(s.charAt(end))) {

end--;

continue;

}

if (s.charAt(start) != s.charAt(end)) {

return false;

}

start++;

end--;

}

return true;

}

public static boolean isValid(char ch) {

return (ch >= 'a' && ch <= 'z') || (ch >= '0' && ch <= '9');

}