ADS CCEE Quiz 2

Total points 25/40

The respondent's email (**harshal.tarmale.cmaug25@gmail.com**) was recorded on submission of this form.

0 of 0 points

PRN: *

250840320073

Name *

Harshal Vilas Tarmale

MCQ

25 of 40 points

✓ The degree of a node in a tree is: * 1/1

- (●) A) Number of its children ✓
- ( ) B) Number of its ancestors
- ( ) C) Number of its siblings
- ( ) D) None of these

✓ A skewed binary tree is similar to: * 1/1

- (●) A) A linked list ✓
- ( ) B) A complete binary tree
- ( ) C) A full binary tree
- ( ) D) A balanced tree

✓ What is the time complexity of the following code? * 1/1

```
for (int i = 0; i < n; i++) {
   for (int j = 0; j < Math.sqrt(n); j++) {
      System.out.println(i + j);
   }
}
```

- ( ) A) O(n)
- (●) B) O(n√n) ✓
- ( ) C) O(n²)
- ( ) D) O(log n)

✗ Which traversal is used in Depth First Search (DFS)? * 0/1

- ○ A) Inorder, Preorder, Postorder
- ○ B) Level Order
- ○ C) Breadth First
- ◉ D) None of the above ✗

Correct answer

- ◉ A) Inorder, Preorder, Postorder

✗ What is the time complexity? * 0/1

```
for (int i = 1; i < n; i = i * 2) {
    for (int j = 0; j < i; j++) {
        System.out.println(i + j);
    }
}
```

- ○ A) O(n)
- ○ B) O(n log n)
- ◉ C) O(n^2) ✗
- ○ D) O(log n)

Correct answer

- ◉ A) O(n)

✓ What is the output of the following code? * 1/1

```
ArrayList<Integer> list = new ArrayList<>();
for(int i=1;i<=5;i++){
   list.add(i*i);
}
int val = list.get(2) + list.get(4);
System.out.println(val);
```

- ○ 20
- ○ 25
- ◉ 34 ✓
- ○ 13

✗ Which recursion type executes last statement first while unwinding? * 0/1

- ○ a) Tail recursion
- ○ b) Head recursion
- ◉ c) Nested recursion ✗
- ○ d) Indirect recursion

Correct answer

- ◉ b) Head recursion

✗ What is the time complexity of the following code? * 0/1

```
for (int i = 0; i < n; i += 2) {
    System.out.println(i);
}
```

- A) O(1)
- B) O(n)
- C) O(n/2) ✗
- D) O(log n)

Correct answer

- B) O(n)

✓ What is the output of the following code? * 1/1

```
int arr[] = {1,2,3,4,5};
int sum = 0;
for(int i=0;i<arr.length;i+=2){
    sum += arr[i];
}
System.out.println(sum);
```

- 15
- 6
- 9 ✓
- 4

✓ The maximum number of nodes in a BST of height h is: * 1/1

- A) h
- B) 2^h - 1 ✓
- C) h^2
- D) $\log_2 h$

✗ Node curr = head; * 0/1

```
while(curr.next != null) curr = curr.next;

while(curr != null) {
   System.out.print(curr.data + " ");
   curr = curr.prev;
}
```

If list = 1 ⇔ 2 ⇔ 3, output is:

- a) 1 2 3
- b) 3 2 1
- c) 2 1 3
- d) Error ✗

Correct answer

- b) 3 2 1

✓ Q3. Find the time complexity: * 1/1

```
for (int i = 1; i < n; i = i * 2) {
    System.out.println(i);
}
```

- ○ A) O(n)
- ◉ B) O(log n) ✓
- ○ C) O(n log n)
- ○ D) O(n²)

✓ What is the time complexity of the following code? * 1/1

```
for (int i = 1; i < n; i *= 2) {
    for (int j = n; j > 0; j /= 2) {
        System.out.println(i + j);
    }
}
```

- ○ A) O(n)
- ○ B) O(log n)
- ◉ C) O(log² n) ✓
- ○ D) O(n log n)

✓ What is the primary goal of using data structures in programming? * 1/1

- ○ A) To reduce the need for comments in code
- ○ B) To make code execution faster
- ◉ C) To organize and manage data efficiently ✓
- ○ D) To simplify code development

✗ In a binary tree, the maximum number of nodes at level l is: * 0/1

- ○ A) l
- ◉ B) 2^l ✗
- ○ C) 2^(l-1)
- ○ D) l²

Correct answer

- ◉ C) 2^(l-1)

✓ In a binary search tree (BST), the left child is always: * 1/1

- ◉ A) Smaller than parent ✓
- ○ B) Greater than parent
- ○ C) Equal to parent
- ○ D) None of these

✓ The degree of a tree is: * 1/1

- ○ A) Number of root nodes
- ◉ B) Maximum degree of any node ✓
- ○ C) Number of leaf nodes
- ○ D) Number of internal nodes

✓ Consider the following Java code snippet for a doubly linked list: * 1/1

```
public void traverseList() {
   Node prev = null;
   Node current = head;
   Node next = null;
   while (current != null) {
      next = current.next;
      current.next = prev;
      prev = current;
      current = next;
   }
   head = prev;
}
```

What does this code do?

- ○ A) Inserts a new node at the end of the list
- ◉ B) Reverses the order of nodes in the list ✓
- ○ C) Deletes the first node in the list
- ○ D) Searches for a specific node in the list

✓ What is the time complexity of the following code? * 1/1

```
for (int i = 0; i < n; i++) {
   System.out.println(i);
}
```

- ○ A) O(1)
- ◉ B) O(n) ✓
- ○ C) O(n²)
- ○ D) O(log n)

✗ The maximum number of nodes in a binary tree of height h is: * 0/1

- ○ A) h
- ○ B) 2^h - 1
- ◉ C) 2^(h+1) ✗
- ○ D) h²

Correct answer

- ◉ B) 2^h - 1

✓ In a tree, the topmost node is called the: * 1/1

- ○ A) Parent
- ◉ B) Root ✓
- ○ C) Leaf
- ○ D) Child

✗ Time complexity of the code: * 0/1

```
for (int i = n; i > 0; i = i / 2) {
    for (int j = 0; j < i; j++) {
        System.out.println(i + j);
    }
}
```

- ○ A) O(n)
- ○ B) O(n log n)
- ○ C) O(n²)
- ◉ D) O(log² n) ✗

Correct answer

- ◉ A) O(n)

✗ What is the output of the following code? * 0/1

```
class A {
 static void m( ){ }
}
class B extends A {
   @Override
   static void m( ){ }
}
```

- (●) a) Compiles ✗
- ( ) b) Compilation error (@Override invalid)
- ( ) c) Runtime error
- ( ) d) Works but warns

Correct answer

- (●) b) Compilation error (@Override invalid)

✓ What is the output of the following code? * 1/1

```
int n = -3;
int[] a = new int[n];
System.out.println(a.length);
```

- ( ) a) Prints -3
- ( ) b) Prints 0
- ( ) c) ArrayIndexOutOfBoundsException
- (●) d) NegativeArraySizeException ✓

✓ Which is true for linked list vs arrays? * 1/1

- (●) a) Arrays allow random access, linked lists don't ✓
- ( ) b) Arrays save memory, linked lists waste memory
- ( ) c) Linked lists are always faster
- ( ) d) Arrays don't support insertion

✓ What is time complexity? * 1/1

- ( ) A) The amount of time it takes to write code
- ( ) B) The number of statements in a program
- (●) C) The amount of time an algorithm takes to run as a function of its input size ✓
- ( ) D) The space used by an algorithm

✓ What is the time complexity of the following code? * 1/1

```
for (int i = 0; i < n; i++) {
   for (int j = 0; j < i * i; j++) {
      System.out.println(i + j);
   }
}
```

- A) $O(n^2)$
- B) $O(n^3)$ ✓
- C) $O(n^4)$
- D) $O(n \log n)$

✗ A complete binary tree is: * 0/1

- A) All levels filled except possibly the last, filled left to right
- B) All levels filled completely ✗
- C) Only root and leaf nodes present
- D) Skewed tree

Correct answer

- A) All levels filled except possibly the last, filled left to right

✓ In a full binary tree of height h, number of leaf nodes is: * 1/1

- (●) A) 2^h ✓
- ( ) B) 2^(h-1)
- ( ) C) h
- ( ) D) h²

✗ Postorder traversal follows: * 0/1

- ( ) A) Root → Left → Right
- (●) B) Left → Root → Right ✗
- ( ) C) Left → Right → Root
- ( ) D) Right → Root → Left

Correct answer

- (●) C) Left → Right → Root

✓ What is the time complexity? * 1/1

```
for (int i = 0; i < n; i++) {
   for (int j = 0; j < i; j++) {
      System.out.println(i + j);
   }
}
```

- A) O(n)
- B) O(n log n)
- C) O(n²) ✓
- D) O(n³)

✓ What is a common approach to implementing backtracking algorithms? * 1/1

- Using Loop
- Using Stack
- Using Recursion ✓
- Using Dynamic Programing

✗ The height of a tree is the length of: * 0/1

- ○ A) Shortest path from root to a leaf
- ○ B) Longest path from root to a leaf
- ○ C) Number of internal nodes
- ◉ D) None of these ✗

Correct answer

- ◉ B) Longest path from root to a leaf

✗ public class Main * 0/1

```
{
   public static void main(String[] args) {
String[] s = new String[1];
System.out.println(s[0].length());
   }
}
```

- ○ a) Prints 0
- ○ b) Prints null
- ○ c) NullPointerException
- ◉ d) Compilation error ✗

Correct answer

- ◉ c) NullPointerException

✓ Which of the following is NOT a type of binary tree? * 1/1

- ○ A) Full binary tree
- ○ B) Complete binary tree
- ○ C) Extended binary tree
- ◉ D) Circular binary tree ✓

✗ What is the output of the following code? * 0/1

```
int[][] m = new int[2][];
m[0] = new int[]{1,2};
int x = m[1][0];
System.out.println(x);
```

- ◉ a) Prints 0 ✗
- ○ b) Prints garbage value
- ○ c) NullPointerException
- ○ d) ArrayIndexOutOfBoundsException

Correct answer

- ◉ c) NullPointerException

✓ What is the time complexity of the following code? * 1/1

```
for (int i = 0; i < n; i++) {
   for (int j = i; j > 0; j /= 2) {
      System.out.println(i + j);
   }
}
```

- ○ A) O(n)
- ◉ B) O(n log n) ✓
- ○ C) O(n²)
- ○ D) O(log² n)

✗ Which traversal of BST gives elements in sorted order? * 0/1

- ○ A) Preorder
- ○ B) Inorder
- ◉ C) Postorder ✗
- ○ D) Level order

Correct answer

- ◉ B) Inorder

✓ A tree is a ___ data structure. * 1/1

- A) Linear
- B) Non-linear ✓
- C) Both A & B
- D) None of the above

✓ What is the time complexity of the following code? * 1/1

```
for (int i = 0; i < n; i++) {
   for (int j = 0; j < n; j++) {
      for (int k = 0; k < n; k++) {
         System.out.println(i + j + k);
      }
   }
}
```

- A) O(n)
- B) O(n^2)
- C) O(n^3) ✓
- D) O(log n)