QUESTIONS

Q1. How does searching a value in a Binary Search Tree work?

Q2. What happens if we search for a value that is not present in the Binary Search Tree?

Q3. Which of the following is false about a binary search tree?

a) The left child is always smaller than its parent

b) The right child is always larger than its parent

c) The left and right sub-trees should also be binary search trees

d) Inorder sequence gives decreasing order of elements

Q4. What is the speciality about the inorder traversal of a binary search tree?

a) It traverses in a non increasing order

b) It traverses in an increasing order

c) It traverses in a random fashion

d) It traverses based on priority of the node

Q5. How will you find the minimum element in a binary search tree?

a)

```
public void min(Tree root)
{
        while(root != null)
        {
           root = root.left();
```

```
        }
      System.out.println(root.data());
}
b)
public void min(Tree root)
{
        while(root.left() != null)
        {
                root = root.left();
        }
        System.out.println(root.data());
}
c)
public void min(Tree root)
{
        while(root.right() != null)
        {
                root = root.right();
        }
        System.out.println(root.data());
}
d)
public void min(Tree root)
{
        while(root != null)
        {
                root = root.right();
        }
        System.out.println(root.data());
}
```

Q6. Suppose the numbers 7, 5, 1, 8, 3, 6, 0, 9, 4, 2 are inserted in that order into an initially empty binary search tree. The binary search tree uses the usual ordering on natural numbers. What is the in-order traversal sequence of the resultant tree?

a) 7 5 1 0 3 2 4 6 8 9

b) 9 8 6 4 2 3 0 1 5 7

c) 0 1 2 3 4 5 6 7 8 9

d) 0 2 4 3 1 6 5 9 8 7

Q7. The following numbers are inserted into an empty binary search tree in the given order: 10, 1, 3, 5, 15, 12, 16. What is the height of the binary search tree (the height is the maximum distance of a leaf node from the root)?

a) 2

b) 3

c) 4

d) 6

Q8. Consider the following code snippet in C. The function print() receives root of a Binary Search Tree (BST) and a positive integer k as arguments.

```
// A BST node
struct node {
   int data;
   struct node *left, *right;
};

int count = 0;

void print(struct node *root, int k)
{
```

```
  if (root != NULL && count <= k)
  {
     print(root->right, k);
     count++;
     if (count == k)
      printf("%d ", root->data);
    print(root->left, k);
  }
}
```

What is the output of print(root, 3) where root represent root of the following BST?

a) 10

b) 16

c) 20

d) 20 10

Q9. While inserting the elements 71, 65, 84, 69, 67, 83 in an empty binary search tree (BST) in the sequence shown, the element in the lowest level is

a) 65

b) 67

c) 69

d) 83

Q10. Suppose that we have numbers between 1 and 90 in a binary search tree and want to search for the number 54. Which of the following sequences CANNOT be the sequence of nodes examined?

a) {10, 78, 65, 44, 60, 58, 54}

b) {90, 15, 68, 34, 66, 48, 54}

c) {9, 85, 47, 68, 43, 57, 54}

d) {79, 16, 72, 56, 18, 53, 54}

Q11. How do you delete a node from a binary search tree in C in the following cases?

Node to be deleted is a leaf node.

A. Remove the node

Node to be removed has one child.

A. Move the child to the node and delete the node

Q12. A Binary Search Tree (BST) stores values in the range 40 to 560. Consider the following sequence of keys.

I) 81, 537, 105, 439, 296, 376, 305

II) 76, 98, 126, 199, 242, 383, 482

III) 144, 258, 520, 386, 345, 270, 307

IV) 556, 169, 507, 395, 465, 402, 270

Suppose the BST has been unsuccessfully searched for key 273. Which all of the above sequences list nodes in the order in which we could have encountered them in the search?

a) II and III only

b) I and III only

c) III and IV only

d) III only

Questions:

Q1. In the following code snippet, what is "x"?

char (*x) (char*);

- character pointer
- pointer to char
- pointer to function
- pointer to char pointer

Q2. Boolean datatype only needs 0 or 1 value which can be represented by a single bit. What is the memory size of boolean data type in C?

- 1 byte
- 1 bit
- 2 bit
- 16 byte

Q3. Suppose a global variable and local variable have the same name. Is it is possible to access a global variable from a block where local variables are defined?

Q4. Differentiate between calloc() and malloc()

Q5. What are the valid places where the programmer can apply Break Control Statement?

Q6. Can a C program be compiled or executed in the absence of a main()?

Q7. Where can we not use &(address operator in C)?

- on register variables
- on functions
- on pointer data type variables
- on char data type variables

Q8. What is the difference between declaring a header file with < > and " "?

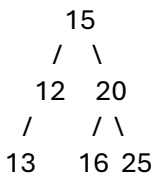
- #include"..." is memory efficient

- #include"..." searches in current directory and #include \<...> searches in standard fixed directories

- Both are same

- #include \<...> searches in current directory and #include"..." searches in standard fixed directories

Q9: Is the following binary tree a BST?



Q10: Which keyword can be used to make a variable unchangeable/read-only?

- final

- readonly

- constant

- const