USN □□□□□□□□□□

Sub:	**DATA STRUCTURES AND APPLICATIONS**					Sub Code:	**BCS304**
Date:		Duration:	**90 minutes**	Max Marks:	**50**	Sem/Sec: III A,B,C	

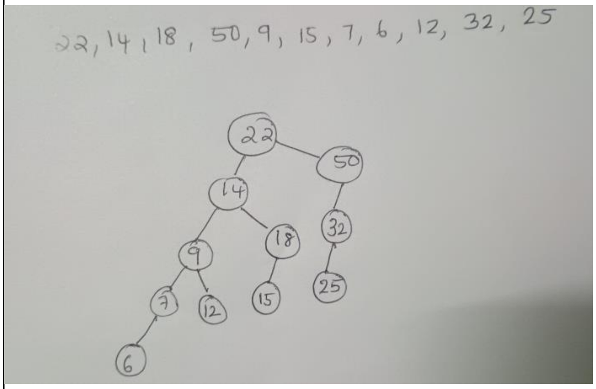
Scheme and Solutions

1 a

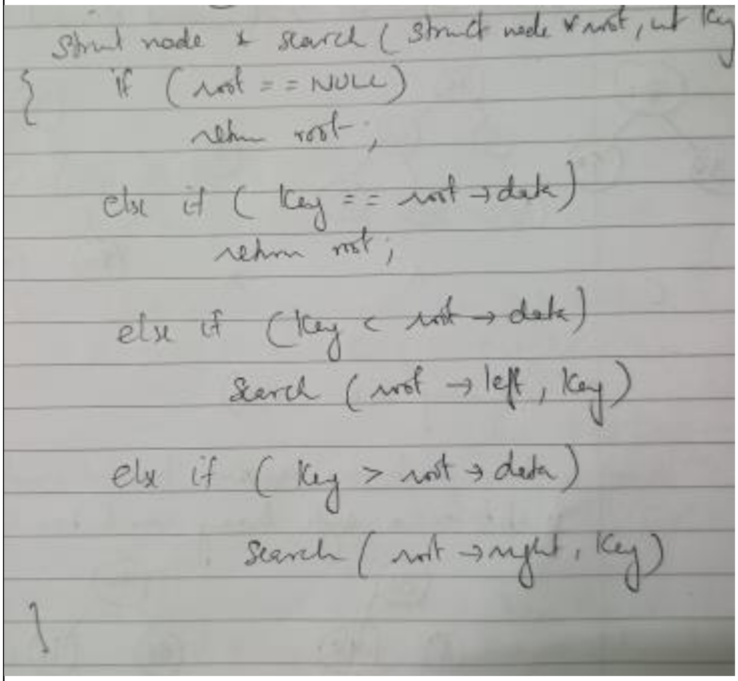
Construct a binary search tree for the inputs 22, 14, 18, 50, 9, 15, 7, 6, 12, 32, 25 also write a function in C to search an item in the BST.

Answer:

Construction of BST-3M(step wise)



Search an item in the BST-3M



```
Struct node * search ( struct node *root, int key
{
        if (root == NULL)
            return root;
       else if ( key == root -> data)
            return root;
       else if (key < root -> data)
              search (root -> left, key)
       else if ( key > root -> data)
              search ( root -> right, key)
}
```

b

Explain winner tree and looser tree with suitable examples.

Answer:

Winner Tree Explanation with example-2M

Looser Tree Explanation with example-2M

2

a

Construct a binary tree by using the following in-order and pre-order traversal.

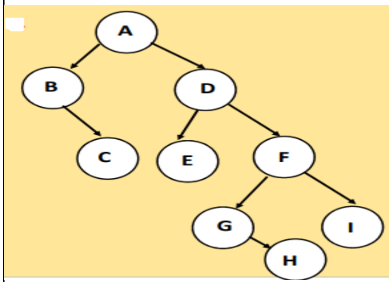
In-order: BCAEDGHFI

Pre-order: ABCDEFGHI

Also perform the post order traversal of the tree.

Answer:

Construction of Binary Tree-3M



Postorder-1M CBEHGIFDA

b

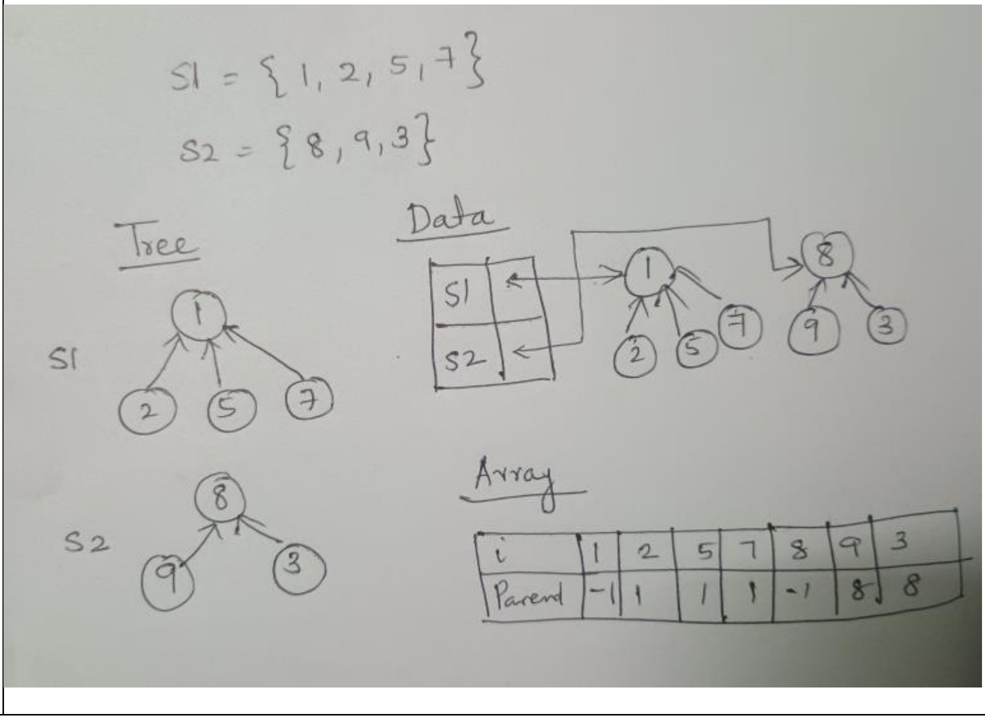
Demonstrate the tree, data, and array representation for the disjoint sets,

S1= {1,2,5,7}

S2= {8,9,3}. Also write algorithm for simple union () and simple find().

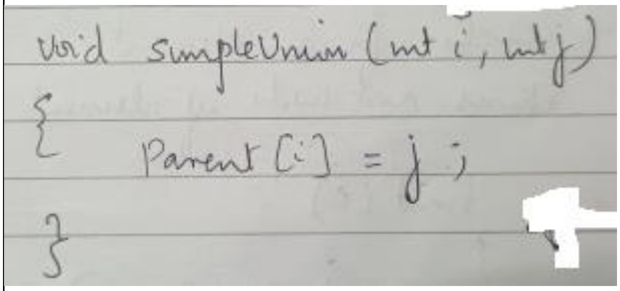
Answer:

of tree, data,Representation array -1M,2M,1M



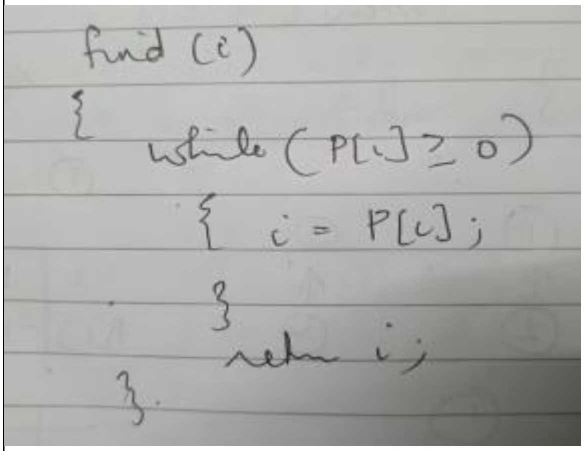
i	1	2	5	7	8	9	3
Parent	-1	1	1	1	-1	8	8

simple union()- 1M



```
void simpleUnion (int i, int j)
{
      Parent [i] = j ;

}
```

simple find()- 1M

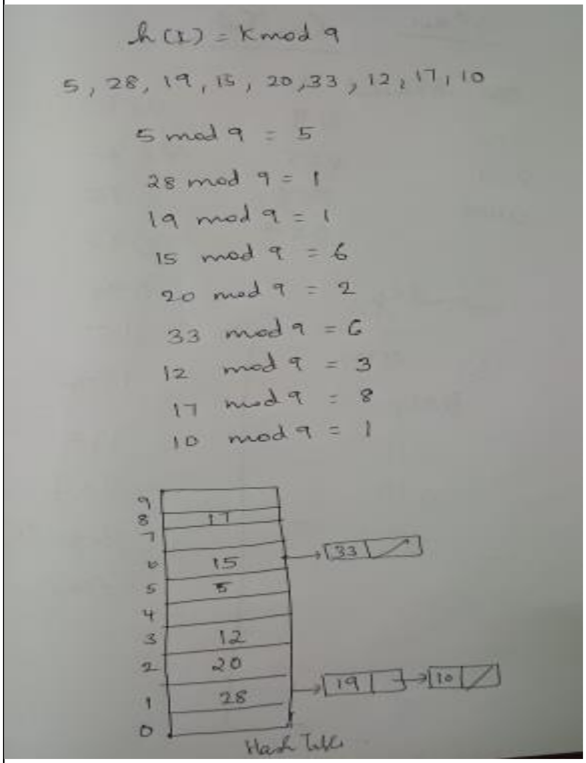


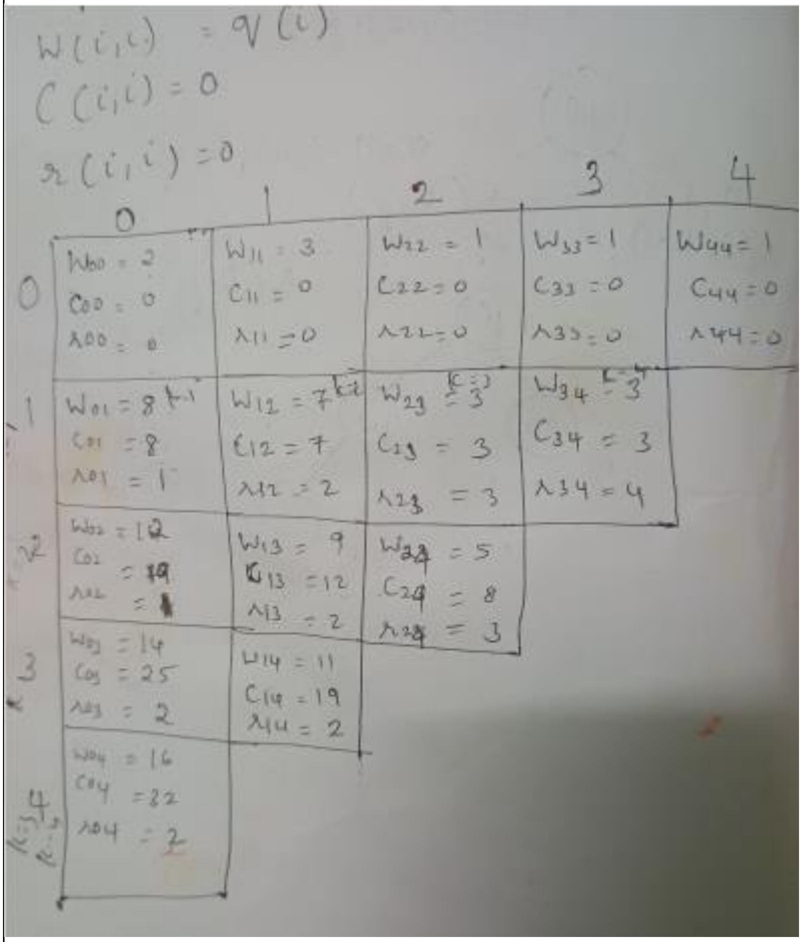
```
find (i)
{
    while ( P[i] ≥ 0 )
      { i = P[i] ;
      }
      return i ;
}.
```

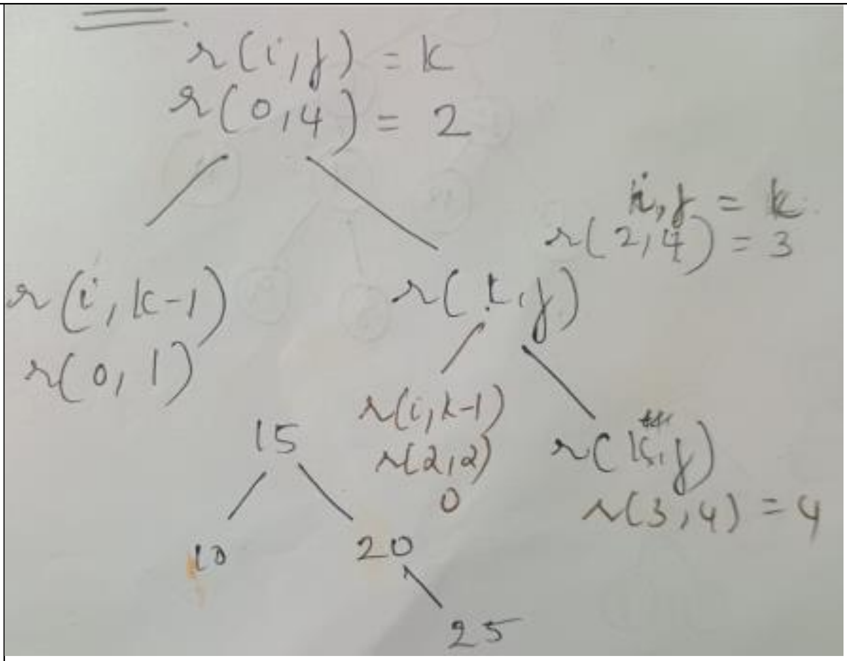
3 a

```
#include <stdio.h>


// Function to insert an element at a specific position in an array
void insertAtPosition(int arr[], int *n, int position, int element) {
    if (position < 0 || position > *n) {
        printf("Invalid position!\n");
        return;
    }


    for (int i = *n; i > position; i--) {
        arr[i] = arr[i - 1];
    }
    arr[position] = element;
    (*n)++; // Increase array size
}


// Function to delete an element from a specific position in an array
void deleteAtPosition(int arr[], int *n, int position) {
    if (position < 0 || position >= *n) {
        printf("Invalid position!\n");
        return;
    }

    for (int i = position; i < *n - 1; i++) {
        arr[i] = arr[i + 1];
    }
    (*n)--; // Decrease array size
}

// Function to insert an element at the end of the array
void insertAtEnd(int arr[], int *n, int element, int maxSize) {
    if (*n >= maxSize) {
        printf("Array is full!\n");
        return;
    }

    arr[*n] = element;
    (*n)++; // Increase array size
}

// Function to delete an element from the end of the array
void deleteAtEnd(int *n) {
    if (*n <= 0) {
        printf("Array is empty!\n");
        return;
    }

    (*n)--; // Decrease array size
}

// Main function to demonstrate the above functions
int main() {
    int arr[10] = {1, 2, 3, 4, 5}; // Array with initial elements
    int n = 5; // Current size of the array
    int maxSize = 10; // Maximum size of the array

    printf("Original Array: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");

    // Insert at position
    insertAtPosition(arr, &n, 2, 99);
    printf("After Inserting 99 at position 2: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");

    // Delete at position
    deleteAtPosition(arr, &n, 2);
    printf("After Deleting element at position 2: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");

    // Insert at end
    insertAtEnd(arr, &n, 77, maxSize);
    printf("After Inserting 77 at the end: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");

    // Delete at end
```

		deleteAtEnd(&n);
		printf("After Deleting element from the end: ");
		for (int i = 0; i < n; i++) {
		printf("%d ", arr[i]);
		}
		printf("\n");
		return 0;
		}

4	a	Given a hash table with 9 slots. The hash function is h(k)=k mod 9. The collision is overcome by chaining. The following keys are inserted in the order. 5,28,19,15,20,33,12,17,10. Develop the corresponding hash table. **Answer:**  h(k) = kmod 9 5, 28, 19, 15, 20, 33, 12, 17, 10 5 mod 9 = 5 28 mod 9 = 1 19 mod 9 = 1 15 mod 9 = 6 20 mod 9 = 2 33 mod 9 = 6 12 mod 9 = 3 17 mod 9 = 8 10 mod 9 = 1 9 8 7 6 5 4 3 2 1 0 17 15 33 5 12 20 28 19 10 Hash Table
	b	Explain the following by taking suitable examples, a) Linear Probing b) Quadratic Probing c) Folding Method **Answer:** **Linear Probing technique with example-2M** **Quadratic Probing technique with example-2M** **Folding technique with example- 2M**
5	a	Explain dynamic hashing using directories with the help of an example. **Answer:**

<table>
<tr><td></td><td></td><td><b>Dynamic hashing using Directories 2M</b>
-Importance of directory and buckets
-Increasing depth of the directory.
<b>Example 3M</b></td></tr>
<tr><td></td><td>b</td><td>Differentiate between height biased and weight biased leftist tree with examples.
<b>Answer:</b>
<b>Height biased leftist tree 2.5M</b>
<b>Weight biased leftist tree 2.5M</b></td></tr>
<tr><td>6</td><td>a</td><td>What is the need for an optimal BST. Find the optimal BST for n=4,
Keys are 10,15,20, 25.
p1, p2, p3, p4 =3,3,1,1
q0, q1, q2, q3, q4 =2,3,1,1,1
<b>Answer:</b>
<b>Need for BST-2M</b>
<b>Problem-8M</b>

W(i,i) = q(i)
C(i,i) = 0
r(i,i) = 0
<table>
<tr><th></th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th></tr>
<tr><td>0</td><td>W00 = 2
C00 = 0
r00 = 0</td><td>W11 = 3
C11 = 0
r11 = 0</td><td>W22 = 1
C22 = 0
r22 = 0</td><td>W33 = 1
C33 = 0
r33 = 0</td><td>W44 = 1
C44 = 0
r44 = 0</td></tr>
<tr><td>1</td><td>W01 = 8 k=1
C01 = 8
r01 = 1</td><td>W12 = 7 k=2
C12 = 7
r12 = 2</td><td>W23 = 3 k=3
C23 = 3
r23 = 3</td><td>W34 = 3 k=4
C34 = 3
r34 = 4</td><td></td></tr>
<tr><td>2</td><td>W02 = 12
C02 = 19
r02 = 1</td><td>W13 = 9
C13 = 12
r13 = 2</td><td>W24 = 5
C24 = 8
r24 = 3</td><td></td><td></td></tr>
<tr><td>3</td><td>W03 = 14
C03 = 25
r03 = 2</td><td>W14 = 11
C14 = 19
r14 = 2</td><td></td><td></td><td></td></tr>
<tr><td>4
k=3,
k=2</td><td>W04 = 16
C04 = 32
r04 = 2</td><td></td><td></td><td></td><td></td></tr>
</table></td></tr>
</table>

		 r(i,j) = k r(0,4) = 2 i,j = k r(2,4) = 3 r(i, k-1) r(0, 1) r(k,j) r(i,k-1) r(2,2) 0 r(k,j) r(3,4) = 4 15 10 20 25

CI **CCI** **HOD**

--All the Best--