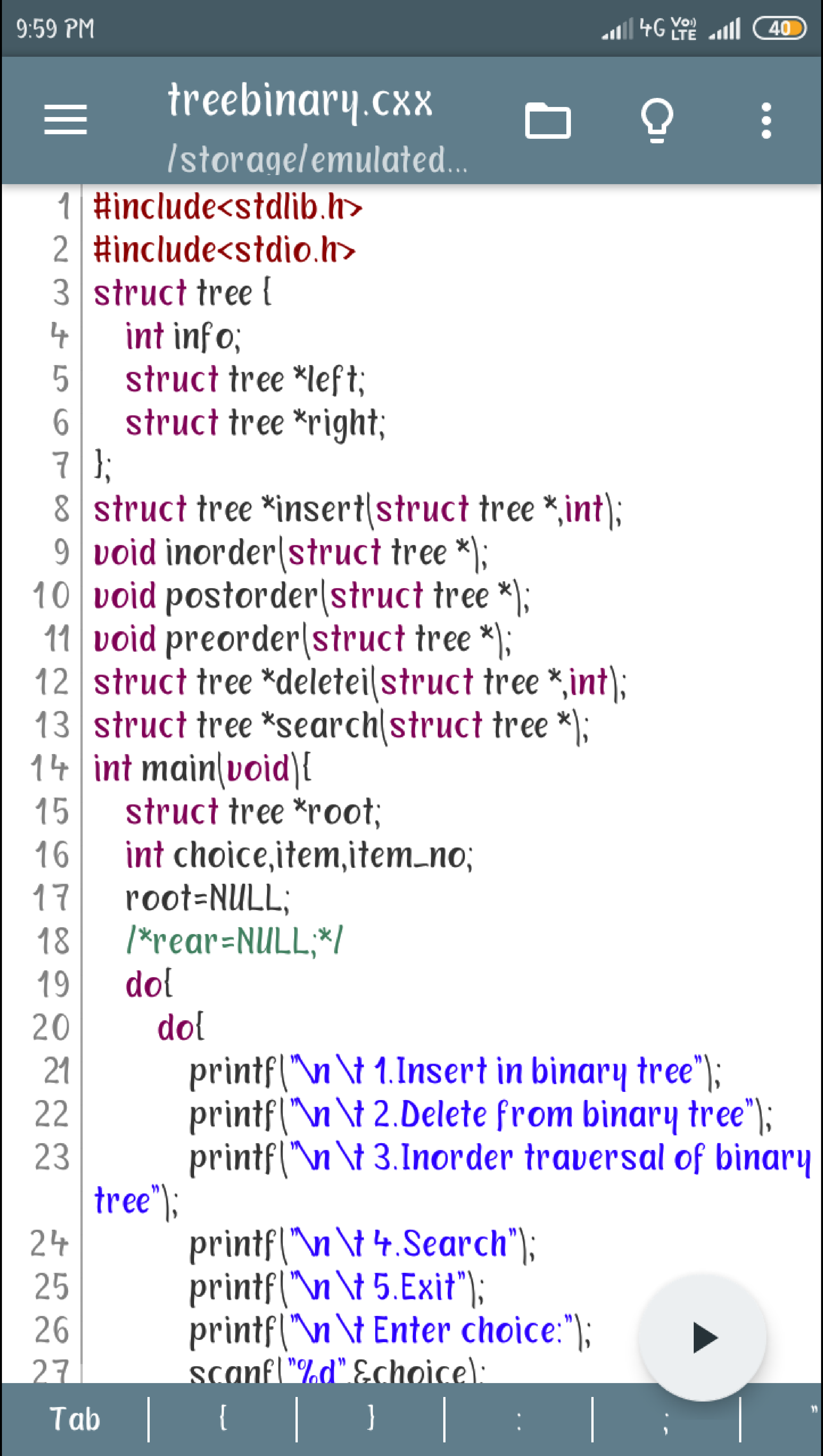
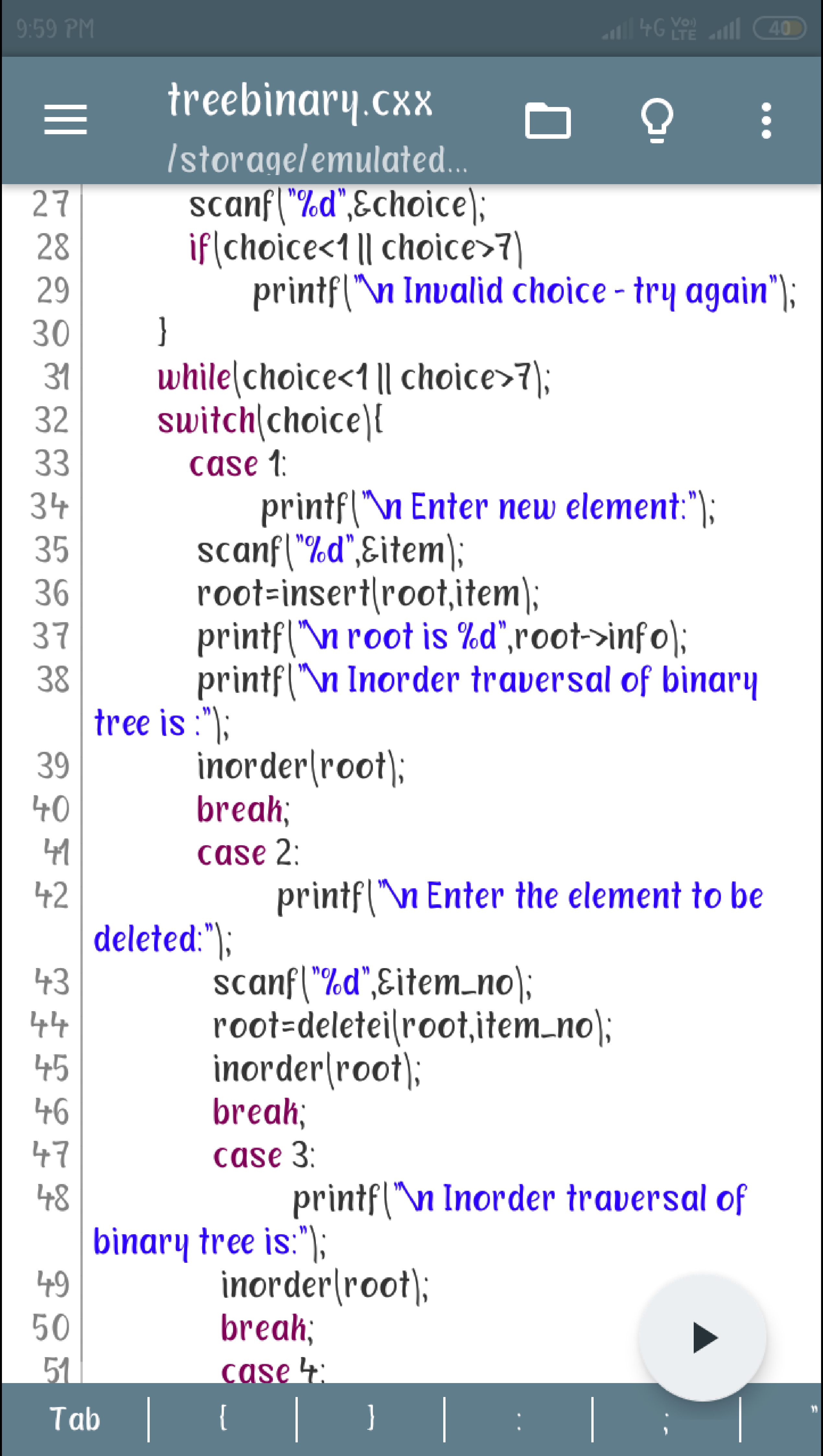
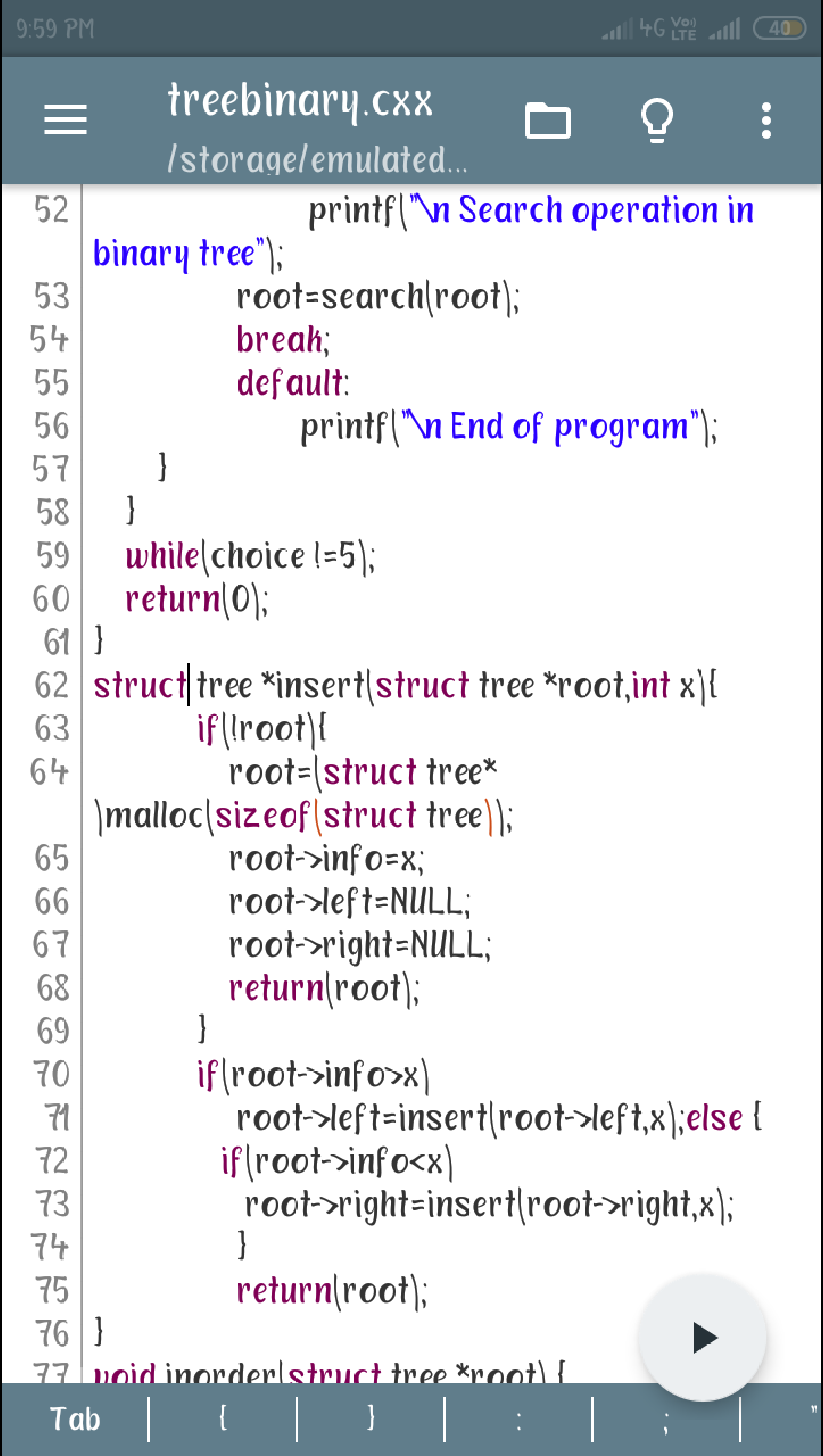


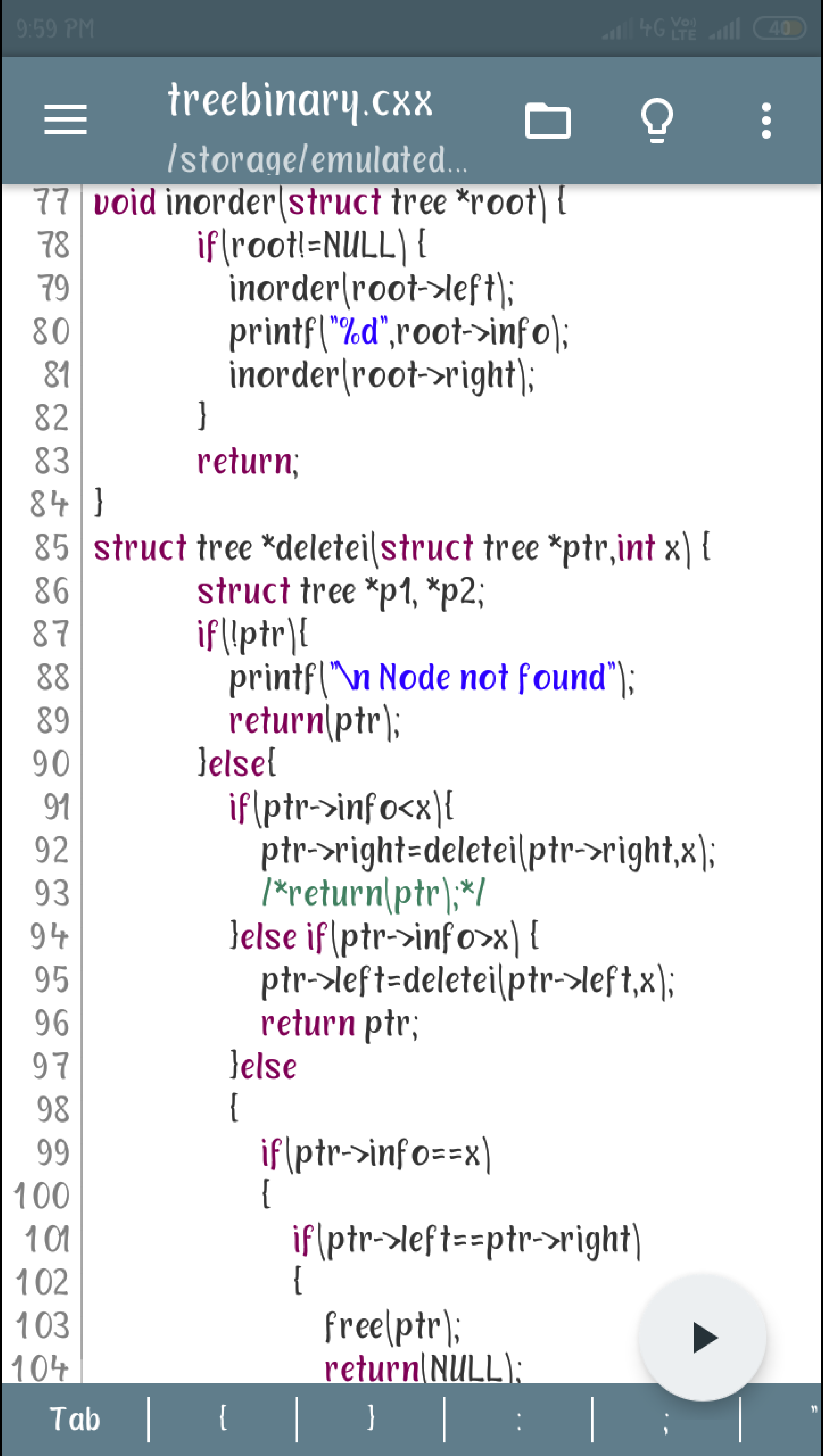
```
#include<stdlib.h>
#include<stdio.h>
struct tree {
   int info;
   struct tree *left;
   struct tree *right;
};
struct tree *insert(struct tree *,int);
void inorder(struct tree *);
void postorder(struct tree *);
void preorder(struct tree *);
struct tree *deletei(struct tree *,int);
struct tree *search(struct tree *);
int main(void){
   struct tree *root;
   int choice,item,item_no;
   root=NULL;
   /*rear=NULL;*/
   do{
      do{
         printf("\n \t 1.Insert in binary tree");
         printf("\n \t 2.Delete from binary tree");
         printf("\n \t 3.Inorder traversal of binary tree");
         printf("\n \t 4.Search");
         printf("\n \t 5.Exit");
         printf("\n \t Enter choice:");
         scanf("%d",&choice);
```



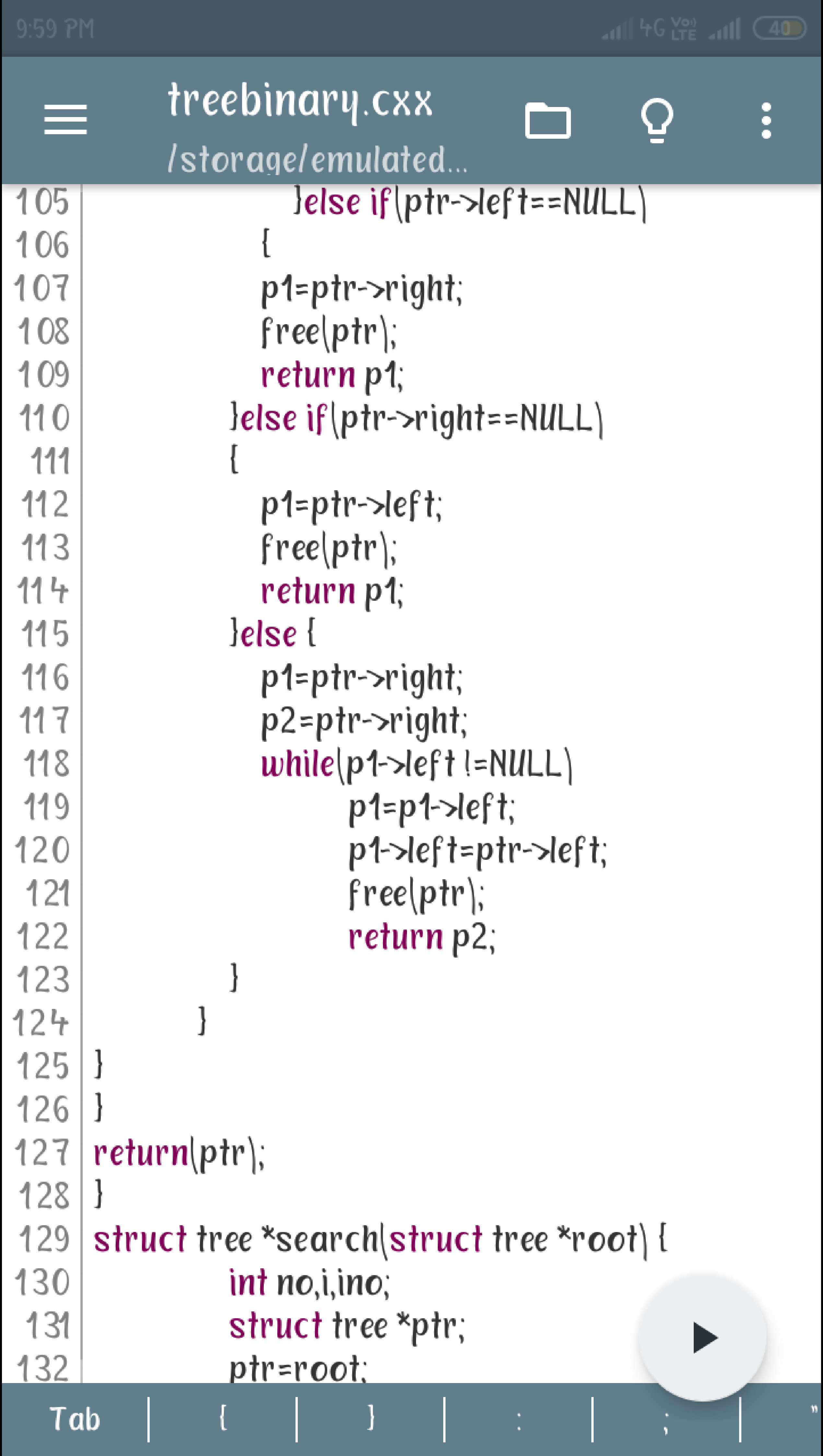
```
        scanf("%d",&choice);
        if(choice<1 || choice>7)
            printf("\n Invalid choice - try again");
      }
      while(choice<1 || choice>7);
      switch(choice){
        case 1:
             printf("\n Enter new element:");
         scanf("%d",&item);
         root=insert(root,item);
         printf("\n root is %d",root->info);
         printf("\n Inorder traversal of binary tree is :");
         inorder(root);
         break;
         case 2:
              printf("\n Enter the element to be deleted:");
          scanf("%d",&item_no);
          root=deletei(root,item_no);
          inorder(root);
          break;
          case 3:
               printf("\n Inorder traversal of binary tree is:");
           inorder(root);
           break;
           case 4:
```



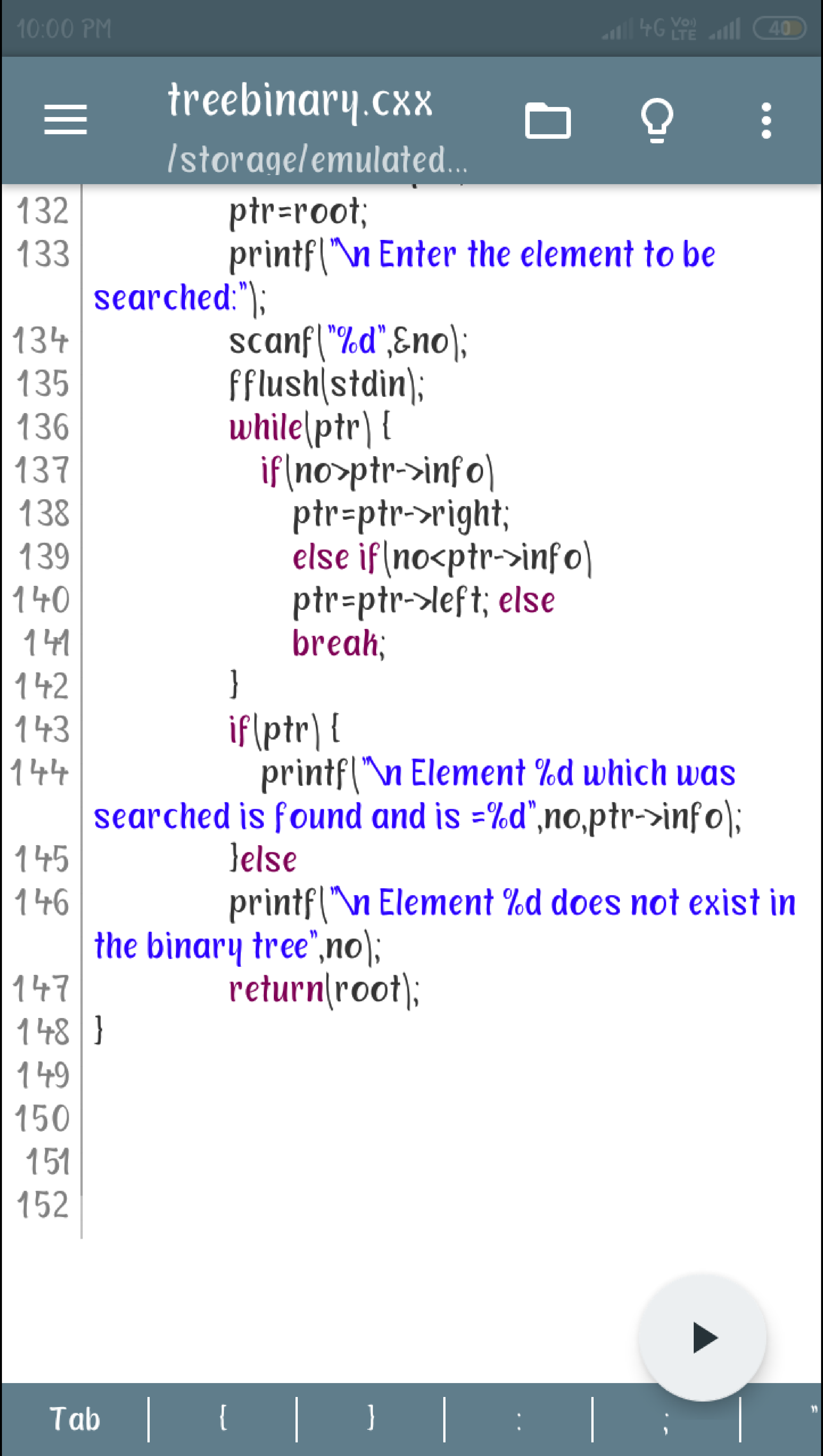
```
                printf("\n Search operation in binary tree");
            root=search(root);
            break;
            default:
                printf("\n End of program");
        }
    }
    while(choice !=5);
    return(0);
}
struct tree *insert(struct tree *root,int x){
        if(!root){
            root=(struct tree*)malloc(sizeof(struct tree));
            root->info=x;
            root->left=NULL;
            root->right=NULL;
            return(root);
        }
        if(root->info>x)
            root->left=insert(root->left,x);else {
          if(root->info<x)
            root->right=insert(root->right,x);
            }
            return(root);
}
void inorder(struct tree *root) {
```



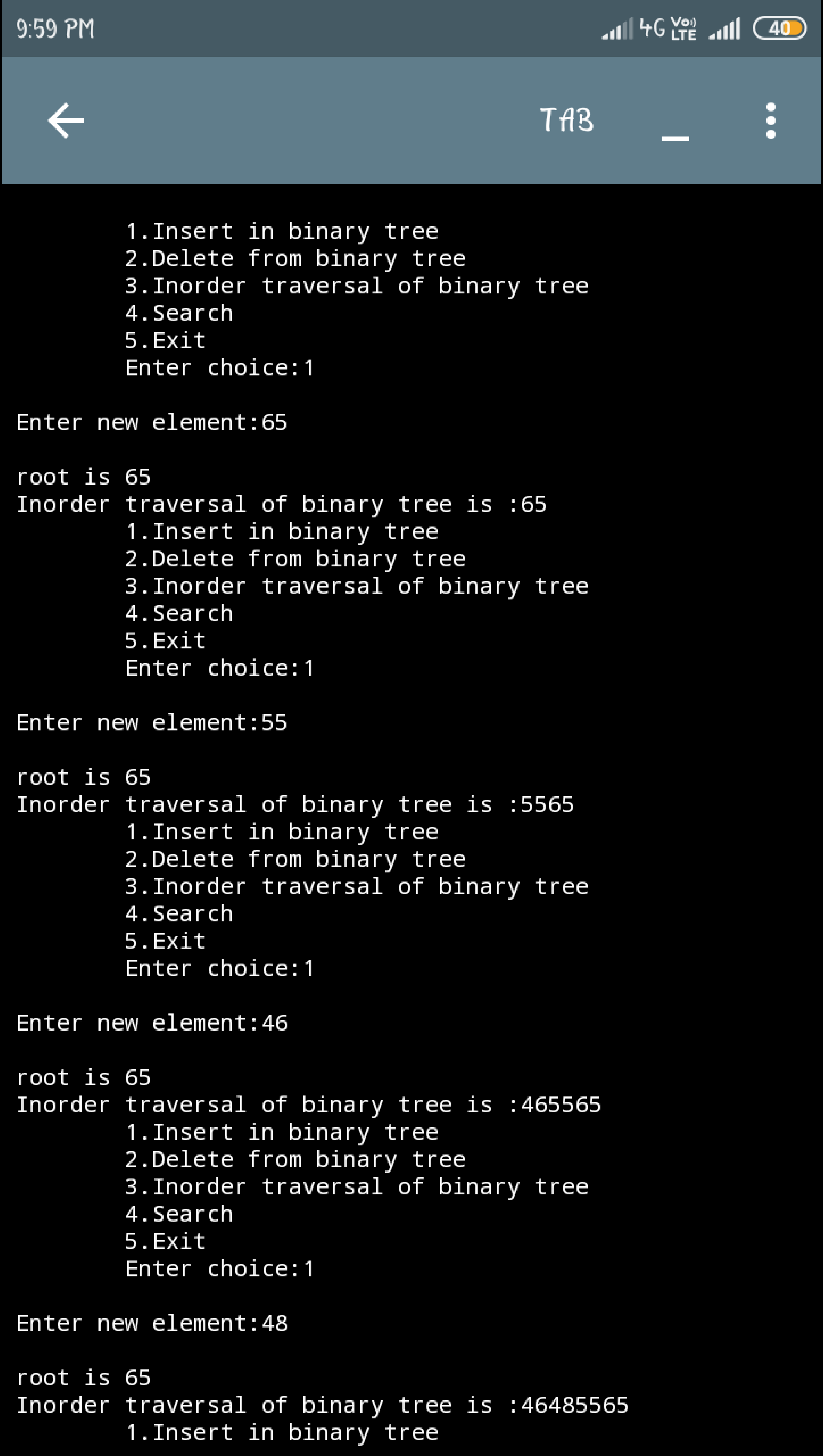
```
void inorder(struct tree *root) {
      if(root!=NULL) {
        inorder(root->left);
        printf("%d",root->info);
        inorder(root->right);
      }
      return;
}
struct tree *deletei(struct tree *ptr,int x) {
      struct tree *p1, *p2;
      if(!ptr){
        printf("\n Node not found");
        return(ptr);
      }else{
        if(ptr->info<x){
          ptr->right=deletei(ptr->right,x);
          /*return(ptr);*/
        }else if(ptr->info>x) {
          ptr->left=deletei(ptr->left,x);
          return ptr;
        }else
        {
          if(ptr->info==x)
          {
            if(ptr->left==ptr->right)
            {
              free(ptr);
              return(NULL);
```



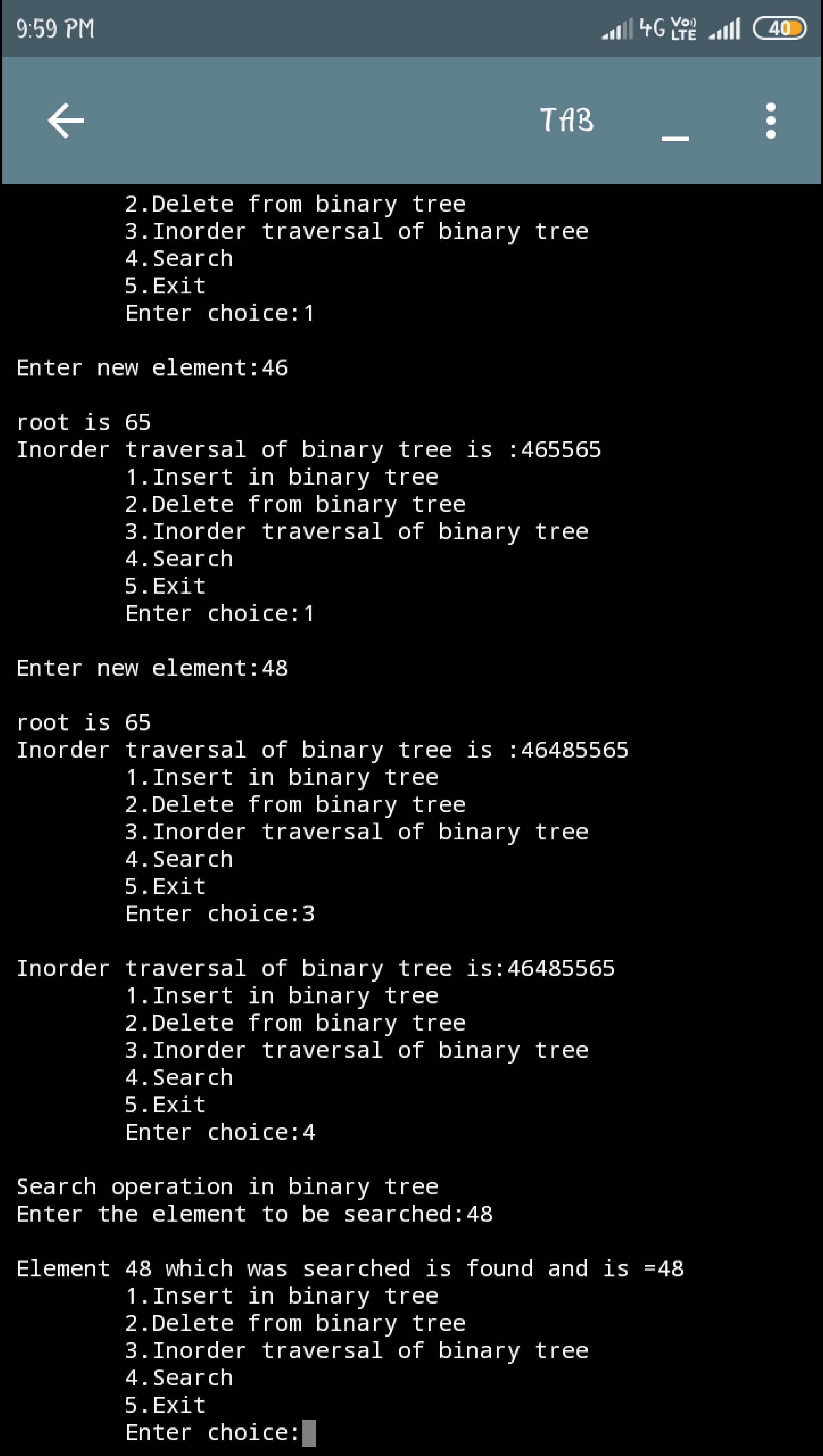
```
            }else if(ptr->left==NULL)
          {
          p1=ptr->right;
          free(ptr);
          return p1;
        }else if(ptr->right==NULL)
        {
          p1=ptr->left;
          free(ptr);
          return p1;
        }else {
          p1=ptr->right;
          p2=ptr->right;
          while(p1->left !=NULL)
                p1=p1->left;
                p1->left=ptr->left;
                free(ptr);
                return p2;
        }
      }
}
}
return(ptr);
}
struct tree *search(struct tree *root) {
        int no,i,ino;
        struct tree *ptr;
        ptr=root;
```



```
        ptr=root;
        printf("\n Enter the element to be searched:");
        scanf("%d",&no);
        fflush(stdin);
        while(ptr) {
          if(no>ptr->info)
            ptr=ptr->right;
            else if(no<ptr->info)
            ptr=ptr->left; else
            break;
        }
        if(ptr) {
          printf("\n Element %d which was searched is found and is =%d",no,ptr->info);
        }else
        printf("\n Element %d does not exist in the binary tree",no);
        return(root);
}
```



```
        1.Insert in binary tree
        2.Delete from binary tree
        3.Inorder traversal of binary tree
        4.Search
        5.Exit
        Enter choice:1

Enter new element:65

root is 65
Inorder traversal of binary tree is :65
        1.Insert in binary tree
        2.Delete from binary tree
        3.Inorder traversal of binary tree
        4.Search
        5.Exit
        Enter choice:1

Enter new element:55

root is 65
Inorder traversal of binary tree is :5565
        1.Insert in binary tree
        2.Delete from binary tree
        3.Inorder traversal of binary tree
        4.Search
        5.Exit
        Enter choice:1

Enter new element:46

root is 65
Inorder traversal of binary tree is :465565
        1.Insert in binary tree
        2.Delete from binary tree
        3.Inorder traversal of binary tree
        4.Search
        5.Exit
        Enter choice:1

Enter new element:48

root is 65
Inorder traversal of binary tree is :46485565
        1.Insert in binary tree
```



```
        2.Delete from binary tree
        3.Inorder traversal of binary tree
        4.Search
        5.Exit
        Enter choice:1

Enter new element:46

root is 65
Inorder traversal of binary tree is :465565
        1.Insert in binary tree
        2.Delete from binary tree
        3.Inorder traversal of binary tree
        4.Search
        5.Exit
        Enter choice:1

Enter new element:48

root is 65
Inorder traversal of binary tree is :46485565
        1.Insert in binary tree
        2.Delete from binary tree
        3.Inorder traversal of binary tree
        4.Search
        5.Exit
        Enter choice:3

Inorder traversal of binary tree is:46485565
        1.Insert in binary tree
        2.Delete from binary tree
        3.Inorder traversal of binary tree
        4.Search
        5.Exit
        Enter choice:4

Search operation in binary tree
Enter the element to be searched:48

Element 48 which was searched is found and is =48
        1.Insert in binary tree
        2.Delete from binary tree
        3.Inorder traversal of binary tree
        4.Search
        5.Exit
        Enter choice:
```