ABALOS, JHO RAVEN
CPE21S1
FINAL EXAM PART 1 - DATA STRUCTURE AND ALGORITHMS

INPUT:

```
class BinaryTree:
    def __init__(self):
        self.root = None

    class Node:
        def __init__(self, data):
            self.data = data
            self.left = None
            self.right = None

    def newNode(self, data):
        return self.Node(data)

    def inorder(self, root):
        if root:
            self.inorder(root.left)
            print(root.data, end=" ")
            self.inorder(root.right)

    def insert(self, root, data):
        if root is None:
            return self.newNode(data), True
        if data < root.data:
            root.left, inserted = self.insert(root.left, data)
        elif data > root.data:
            root.right, inserted = self.insert(root.right, data)
        else:
            inserted = False
        return root, inserted

    def search(self, root, data):
        if root is None or root.data == data:
            return root is not None

        if data < root.data:
            return self.search(root.left, data)
        else:
            return self.search(root.right, data)
```

```
    def delete(self, root, data):
        if root is None:
            return root, False

        if data < root.data:
            root.left, deleted = self.delete(root.left, data)
        elif data > root.data:
            root.right, deleted = self.delete(root.right, data)
        else:
            # Node with only one child or no child
            if root.left is None:
                return root.right, True
            elif root.right is None:
                return root.left, True

            # Node with two children: Get the inorder successor
            root.data = self.min_value_node(root.right).data
            # Delete the inorder successor
            root.right, deleted = self.delete(root.right, root.data)

        return root, deleted

    def min_value_node(self, node):
        current = node
        while current.left is not None:
            current = current.left
        return current

    def display(self):
        if self.root is None:
            print("The tree is empty.")
        else:
            print("Nodes of the tree are: ", end="")
            self.inorder(self.root)
            print()


if __name__ == "__main__":
    tree = BinaryTree()

    while True:
        print("Please choose an action.")
        print("[1] Insert data into a tree.")
        print("[2] Display all data from the tree.")
```

```
print("[3] Search for data in the tree.")
print("[4] Delete data from the tree.")
print("[5] Exit.")

op1 = int(input())
if op1 == 1:
    counter = 0
    while counter < 10:
        try:
            data = int(input("Enter data: "))
            tree.root, inserted = tree.insert(tree.root, data)
            if not inserted:
                print(f"The data {data} already exists in the tree.")
            counter += 1
        except ValueError:
            print("Invalid input. Please enter an integer.")
    else:
        print("The tree is full.")
elif op1 == 2:
    tree.display()
elif op1 == 3:
    data = int(input("Enter data to search: "))
    if tree.search(tree.root, data):
        print(f"The data {data} is present in the tree.")
    else:
        print(f"The data {data} is not present in the tree.")
elif op1 == 4:
    data = int(input("Enter data to delete: "))
    tree.root, deleted = tree.delete(tree.root, data)
    if deleted:
        print(f"The data {data} has been deleted from the tree.")
    else:
        print(f"The data {data} is not present in the tree.")
elif op1 == 5:
    print("Thank you.")
    break
else:
    print("You have entered an invalid input.")

op2 = input("\nWould you like to try again? (Y/N): ")
if op2.lower() != 'y':
    print("Thank you.")
    break
```

OUTPUT:

```
Please choose an action.
[1] Insert data into a tree.
[2] Display all data from the tree.
[3] Search for data in the tree.
[4] Delete data from the tree.
[5] Exit.
1
Enter data: 1
Enter data: 2
Enter data: 3
Enter data: 4
Enter data: 5
Enter data:
Invalid input. Please enter an integer.
Enter data: 67
Enter data: 7
Enter data: 8
Enter data:
Invalid input. Please enter an integer.
Enter data: 9
Enter data: 10
The tree is full.

Would you like to try again? (Y/N): Y
Please choose an action.
[1] Insert data into a tree.
[2] Display all data from the tree.
[3] Search for data in the tree.
[4] Delete data from the tree.
[5] Exit.
2
Nodes of the tree are: 1 2 3 4 5 7 8 9 10 67

Would you like to try again? (Y/N): Y
Please choose an action.
[1] Insert data into a tree.
[2] Display all data from the tree.
[3] Search for data in the tree.
[4] Delete data from the tree.
[5] Exit.
3
Enter data to search: 10
The data 10 is present in the tree.

Would you like to try again? (Y/N): Y
Please choose an action.
[1] Insert data into a tree.
[2] Display all data from the tree.
[3] Search for data in the tree.
[4] Delete data from the tree.
[5] Exit.
4
Enter data to delete: 67
The data 67 has been deleted from the tree.
```

```
Enter data:
Invalid input. Please enter an integer.
Enter data: 9
Enter data: 10
The tree is full.

Would you like to try again? (Y/N): Y
Please choose an action.
[1] Insert data into a tree.
[2] Display all data from the tree.
[3] Search for data in the tree.
[4] Delete data from the tree.
[5] Exit.
2
Nodes of the tree are: 1 2 3 4 5 7 8 9 10 67

Would you like to try again? (Y/N): Y
Please choose an action.
[1] Insert data into a tree.
[2] Display all data from the tree.
[3] Search for data in the tree.
[4] Delete data from the tree.
[5] Exit.
3
Enter data to search: 10
The data 10 is present in the tree.

Would you like to try again? (Y/N): Y
Please choose an action.
[1] Insert data into a tree.
[2] Display all data from the tree.
[3] Search for data in the tree.
[4] Delete data from the tree.
[5] Exit.
4
Enter data to delete: 67
The data 67 has been deleted from the tree.

Would you like to try again? (Y/N): Y
Please choose an action.
[1] Insert data into a tree.
[2] Display all data from the tree.
[3] Search for data in the tree.
[4] Delete data from the tree.
[5] Exit.
2
Nodes of the tree are: 1 2 3 4 5 7 8 9 10

Would you like to try again? (Y/N): N
Thank you.


...Program finished with exit code 0
Press ENTER to exit console.
```