Integrantes: Deisyre Hernandez
Angela Tovar

Informe Estructura de Datos

Código

```
#include <iostream>
#include <string>

struct Nodo {
    int valor;
    Nodo* izquierdo;
    Nodo* derecho;

    Nodo(int val) : valor(val), izquierdo(nullptr), derecho(nullptr) {}
};

Nodo* insertar(Nodo* raiz, int valor) {
    if (raiz == nullptr) {
        return new Nodo(valor);
    }
    if (valor < raiz->valor) {
        raiz->izquierdo = insertar(raiz->izquierdo, valor);
    }
    else if (valor > raiz->valor) {
        raiz->derecho = insertar(raiz->derecho, valor);
    }
    return raiz;
}
```

Integrantes: Deisyre Hernandez
Angela Tovar

```
bool buscar(Nodo* raiz, int valor) {
    if (raiz == nullptr) {
        return false;
    }
    if (valor == raiz->valor) {
        return true;
    }
    else if (valor < raiz->valor) {
        return buscar(raiz->izquierdo, valor);
    }
    else {
        return buscar(raiz->derecho, valor);
    }
}

void imprimirEnOrden(Nodo* raiz) {
    if (raiz != nullptr) {
        imprimirEnOrden(raiz->izquierdo);
        std::cout << raiz->valor << " ";
        imprimirEnOrden(raiz->derecho);
    }
}

void imprimirPreOrden(Nodo* raiz) {
    if (raiz != nullptr) {
        std::cout << raiz->valor << " ";
        imprimirPreOrden(raiz->izquierdo);
```

Integrantes: Deisyre Hernandez
Angela Tovar

```
        imprimirPreOrden(raiz->derecho);
    }
}

void imprimirPosOrden(Nodo* raiz) {
    if (raiz != nullptr) {
        imprimirPosOrden(raiz->izquierdo);
        imprimirPosOrden(raiz->derecho);
        std::cout << raiz->valor << " ";
    }
}

Nodo* encontrarMinimo(Nodo* raiz) {
    while (raiz->izquierdo != nullptr) {
        raiz = raiz->izquierdo;
    }
    return raiz;
}

Nodo* eliminar(Nodo* raiz, int valor) {
    if (raiz == nullptr) {
        return raiz;
    }

    if (valor < raiz->valor) {
        raiz->izquierdo = eliminar(raiz->izquierdo, valor);
    }
    else if (valor > raiz->valor) {
```

Integrantes: Deisyre Hernandez
Angela Tovar

```
        raiz->derecho = eliminar(raiz->derecho, valor);
    }
    else {

        if (raiz->izquierdo == nullptr) {
            Nodo* temp = raiz;
            raiz = raiz->derecho;
            delete temp;
        }
        else if (raiz->derecho == nullptr) {
            Nodo* temp = raiz;
            raiz = raiz->izquierdo;
            delete temp;
        }
        else {
            Nodo* temp = encontrarMinimo(raiz->derecho);
            raiz->valor = temp->valor;
            raiz->derecho = eliminar(raiz->derecho, temp->valor);
        }
    }
    return raiz;
}

void imprimirArbol(Nodo* raiz, int espacio = 0) {
    if (raiz == nullptr) {
        return;
    }
```

Integrantes: Deisyre Hernandez
Angela Tovar

```
    espacio += 10;

    imprimirArbol(raiz->derecho, espacio);

    std::cout << std::endl;
    for (int i = 10; i < espacio; i++) {
        std::cout << " ";
    }
    std::cout << raiz->valor << "\n";

    imprimirArbol(raiz->izquierdo, espacio);
}

int main() {
    Nodo* raiz = nullptr;
    int valor;

    std::cout << "Ingrese números para construir el árbol binario (ingrese 'q' para salir):" << std::endl;

    while (true) {
        std::string entrada;
        std::cin >> entrada;

        if (entrada == "q") {
            break; // Salir del bucle si se ingresa "q"
        }
```

Integrantes: Deisyre Hernandez
Angela Tovar

```
        valor = std::stoi(entrada);


        raiz = insertar(raiz, valor);
    }


    std::cout << "Árbol completo:" << std::endl;
    imprimirArbol(raiz);


    std::cout << "Árbol en orden: ";
    imprimirEnOrden(raiz);
    std::cout << std::endl;


    std::cout << "Árbol en preorden: ";
    imprimirPreOrden(raiz);
    std::cout << std::endl;


    std::cout << "Árbol en posorden: ";
    imprimirPosOrden(raiz);
    std::cout << std::endl;


    std::cout << "Ingrese un número para buscar en el árbol: ";
    std::cin >> valor;
    if (buscar(raiz, valor)) {
        std::cout << "El número " << valor << " se encuentra en el árbol." << std::endl;
    }
    else {
        std::cout << "El numero " << valor << " no se encuentra en el árbol." <<
std::endl;
```

Integrantes: Deisyre Hernandez
Angela Tovar

```
    }

    std::cout << "Ingrese un número para eliminar del árbol: ";
    std::cin >> valor;
    raiz = eliminar(raiz, valor);

    std::cout << "Árbol después de eliminar " << valor << " en orden: ";
    imprimirEnOrden(raiz);
    std::cout << std::endl;

    return 0;
}
```

Resultado