```c
#include <stdio.h>
#include <stdlib.h>

struct arvore {
  int num;
  struct arvore *esq;
  struct arvore *dir;
};
typedef struct arvore Arvore;

void inserir(Arvore **pRaiz, int numero) {
  if (*pRaiz == NULL) {
    *pRaiz = (Arvore *)malloc(sizeof(Arvore));
    (*pRaiz)->esq = NULL;
    (*pRaiz)->dir = NULL;
    (*pRaiz)->num = numero;
  } else {
    if (numero < (*pRaiz)->num)
      inserir(&(*pRaiz)->esq, numero);
    else
      inserir(&(*pRaiz)->dir, numero);
  }
}
void imprimirPreOrdem(Arvore *raiz) {
  if (raiz != NULL) {
    printf("%d ", raiz->num);
    imprimirPreOrdem(raiz->esq);
    imprimirPreOrdem(raiz->dir);
  }
}
void imprimirInOrdem(Arvore *raiz) {
  if (raiz != NULL) {
    imprimirInOrdem(raiz->esq);
    printf("%d ", raiz->num);
    imprimirInOrdem(raiz->dir);
  }
}
void imprimirPosOrdem(Arvore *raiz) {
  if (raiz != NULL) {
    imprimirPosOrdem(raiz->esq);
    imprimirPosOrdem(raiz->dir);
    printf("%d ", raiz->num);
  }
}
int busca(Arvore *raiz, int valor) {
  if (raiz == NULL) {
    printf("Valor nao encontrado");
    return 0;
  } else {
    if (valor < raiz->num)
      busca(raiz->esq, valor);
    else if (valor > raiz->num)
      busca(raiz->dir, valor);
    else
```

```
      printf("Valor encontrado");
    return 1;
  }
}
void removerNo(Arvore **pRaiz, int numero) {
  if (*pRaiz == NULL) {
    return;
  } else if (numero < (*pRaiz)->num) {
    removerNo(&(*pRaiz)->esq, numero);
  } else if (numero > (*pRaiz)->num) {
    removerNo(&(*pRaiz)->dir, numero);
  } else {
    removerNo(&(*pRaiz)->esq, -1);
    removerNo(&(*pRaiz)->dir, -1);
    free(*pRaiz);
    *pRaiz = NULL;
  }
}
int alturaArvore(Arvore *raiz) {
  if (raiz == NULL) {
    return -1;
  } else {
    int alturaEsq = alturaArvore(raiz->esq);
    int alturaDir = alturaArvore(raiz->dir);
    if (alturaEsq > alturaDir) {
      return alturaEsq + 1;
    } else {
      return alturaDir + 1;
    }
  }
}

int main() {
  Arvore *raiz = NULL;
  int opcao, valor;

  while(1){
    printf("\n\nMenu de opções\n");
    printf("1 - Inserir na árvore\n");        // a)
    printf("2 - Imprimir a árvore\n");        // e),f),g)
    printf("3 - Buscar na árvore\n");         // b)
    printf("4 - Remover um nó da árvore\n");  // d)
    printf("5 - Altura da árvore\n");         // c)
    printf("0 - Sair\n");
    printf("Escolha uma opção: ");
    scanf("%d", &opcao);
    switch (opcao){
      case 1:
        printf("Digite um valor: ");
        scanf("%d", &valor);
        inserir(&raiz, valor);
        break;
      case 2:
        printf("\nPré-ordem: ");
```

```
        imprimirPreOrdem(raiz);
        printf("\nIn-ordem: ");
        imprimirInOrdem(raiz);
        printf("\nPós-ordem: ");
        imprimirPosOrdem(raiz);
        break;
      case 3:
        printf("Digite um valor a ser buscado: ");
        scanf("%d", &valor);
        busca(raiz, valor);
        break;
      case 4:
        printf("Digite um valor a ser removido: ");
        scanf("%d", &valor);
        removerNo(&raiz, valor);
        break;
      case 5:
        printf("A altura da árvore é: %d", alturaArvore(raiz));
        break;
      case 0:
        exit(0);
      default:
        printf("Opção inválida");
    }
  }
  return 0;
}
```