



Construção de uma árvore binária de busca
1.º passo: criar a raiz
2.º passo: inserir os nós

Construção de uma árvore binária de busca
3.º passo: percorrer a árvore

```
int main() {  
    int n;  
    cin >> n;  
    if (n == 0) return 0;  
    int *arr = new int[n];  
    for (int i = 0; i < n; i++)  
        arr[i] = i + 1;  
    // Construção da árvore  
    // 1.º passo: criar a raiz  
    // 2.º passo: inserir os nós  
    // 3.º passo: percorrer a árvore  
    // ...  
}
```

Caro Siles de Brumpe Amoroso
Turma: 41
matrículo: 21.1.4111

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

```
int** cria_matriz(int m){  
    int** matriz;  
    matriz = malloc(m * sizeof(int));  
    for(int i=0; i<m; i++){  
        matriz[i] = malloc(m * sizeof(int));  
    }  
    for(int i=0; i<m; i++){  
        for(int j=0; j<m; j++){  
            scanf("%d", &matriz[i][j]);  
        }  
    }  
    return matriz;  
}
```

```
void libera_matriz(int** A, int m){  
    for(int i=0; i<m; i++){  
        free(A[i]);  
    }  
    free(A);  
}
```

```
int main(){  
    int m;  
    scanf("%d", &m);  
    int** matriz = cria_matriz(m);  
    int aux=0;  
    int menor=999999;
```

```

for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        if (matriz[i][j] < menor) {
            cur = matriz[i][j];
            menor = matriz[i][j];
            matriz[i][j] = matriz[i][i];
            matriz[i][i] = cur;
        }
    }
}

```

```

printf("\n");
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        printf("%d", matriz[i][j]);
    }
    printf("\n");
}
desaloca_matriz(matriz, n);
return 0;

```