Prática de Teste Estrutural - Fluxo de Controle

1. Gerar **casos de teste** para executar **todos os nós** e **todos os arcos** do programa identifier.
2. Aplicar o **Critério de McCabe** para o programa *identifier* e gerar os **caminhos básicos**. Gere casos de teste a partir dos requisitos de teste gerados pelos caminhos básicos ou marque os caminhos não executáveis.
3. Gerar o GFC para a função.

```
void insercao(int a[], int size){
    int i, j, aux;
    for (i=1; i<size; i++){
        aux = a[i];
        j = j-1;
        while (j>0 && a[j] > aux){
            a[j+1] = a[j];
            j--;
        }
        a[j+1] = aux;
    }
}
```

4. Gerar o GFC para a função.

```
public static void heapsort(int n, double ra[]) {
    int l, j, ir, i;
    double rra;

    l = (n >> 1) + 1;
    ir = n;
    for (;;) {
        if (l > 1) {
            rra = ra[--l];
        } else {
            rra = ra[ir];
            ra[ir] = ra[1];
            if (--ir == 1) {
                ra[1] = rra;
                return;
            }
        }
        i = l;
        j = l << 1;
        while (j <= ir) {
            if (j < ir && ra[j] < ra[j + 1]) {
                ++j;
            }
            if (rra < ra[j]) {
                ra[i] = ra[j];
                j += (i = j);
            } else {
                j = ir + 1;
            }
        }
```

```
        ra[i] = rra;
    }
}
```

5. Gerar o GFC para a função.

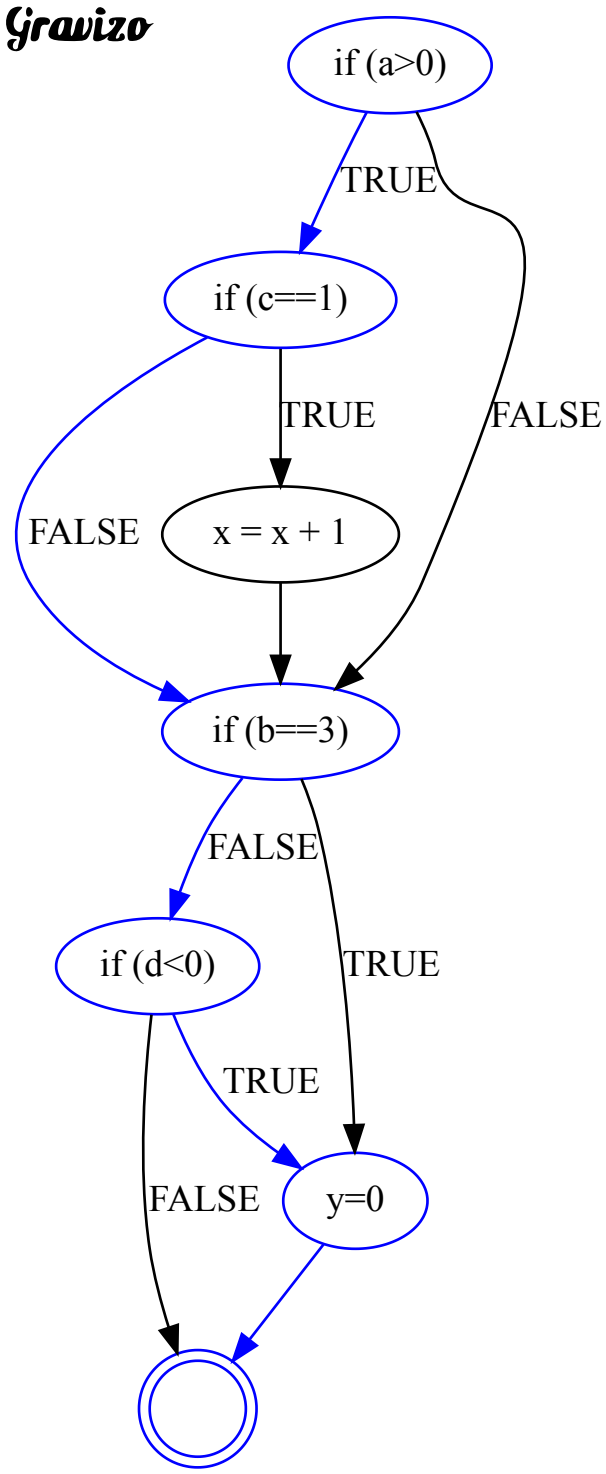
```
void quicksort(int a[], int lo0, int hi0)
{
    int lo = lo0;
    int hi = hi0;
    int mid;

        if (hi0 > lo0) {
        mid = a[(lo0 + hi0) / 2];
        while (lo <= hi) {
            while ((lo < hi0) && (a[lo] < mid)) {
                ++lo;
            }
            while ((hi > lo0) && (a[hi] > mid)) {
                --hi;
            }
            if (lo <= hi) {
                int T = a[lo];
                a[lo] = a[hi];
                a[hi] = T;
                        ++lo;
                --hi;
            }
        }
        if (lo0 < hi) {
            quicksort(a, lo0, hi);
        }
        if (lo < hi0) {
            quicksort(a, lo, hi0);
        }
    }
}
```

Dica: para gerar os GFCs, você pode usar o editor de texto markdown Typora (https://typora.io/) que permite a inserção de grafos na linguagem Dot (https://graphviz.org/doc/info/lang.html). O Typora renderiza os grafos usando o Gravizo (http://www.gravizo.com/). Olhe o exemplo a seguir:

```
![Alt text](https://g.gravizo.com/svg?
  digraph G {
    graph [layout = dot, rankdir = TD];
    1 [label="if (a>0)", color="blue"];
    2 [label="if (c==1)", color="blue"];
    3 [label="x = x + 1"];
    4 [label="if (b==3)", color="blue"];
    5 [label="if (d<0)", color="blue"];
    6 [label="y=0", color="blue"];
    7 [label="", shape="doublecircle", color="blue"];
    1 -> 2 [label="TRUE", color="blue"];
    2 -> 3 [label="TRUE"];
```

```
1 -> 4 [label="FALSE"];
2 -> 4 [label="FALSE", color="blue"]
3 -> 4
4 -> 5 [label="FALSE", color="blue"]
4 -> 6 [label="TRUE"]
5 -> 6 [label="TRUE", color="blue"]
5 -> 7 [label="FALSE"]
6 -> 7 [color="blue"]
})
```



Dica 2: Você também pode usar diretamente o GraphViz que processa a linguagem linguagem Dot (https://graphviz.org/doc/info/lang.html) para gerar as figuras.