Instituto Federal de Educação, Ciência e Tecnologia do maranhão
Algoritmos e Estruturas de Dados ll
Franciele Alves da Silva (20231SI0012)

Relatório

main:

```
#include "ATV3.h"
#include <locale.h>

int main() {setlocale(LC_ALL, "");

    Grafo* gr = cria_Grafo(20, 20, 1);
    le_e_constroi_grafo("grafo_modificado.txt", gr);

    printf("Algoritmo de Prim:\n");
    Prim(gr, 0);

    printf("\nAlgoritmo de Kruskal:\n");
    Kruskal(gr);

    libera_Grafo(gr);
    return 0;
}
```

Árvores mínimas:

Algoritmo de Prim

```
Algoritmo de Prim:
Árvore Mínima
0 - 2 (Peso: 100)
0 - 3 (Peso: 100)
2 - 4 (Peso:  5)
8 - 5 (Peso: 23)
9 - 6 (Peso: 97)
4 - 7 (Peso:  4)
6 - 8 (Peso: 52)
7 - 9 (Peso:  4)
4 - 10 (Peso:  5)
5 - 11 (Peso: 11)
0 - 12 (Peso: 100)
0 - 13 (Peso: 100)
16 - 14 (Peso: 61)
13 - 15 (Peso: 33)
12 - 16 (Peso: 26)
12 - 17 (Peso: 48)
0 - 18 (Peso: 100)
18 - 19 (Peso: 46)
0 - 20 (Peso: 100)
```

Algoritmo de Kruskal

```
Algoritmo de Kruskal:
Árvore Mínima
7 - 9 (Peso:  4)
4 - 7 (Peso:  4)
4 - 10 (Peso:  5)
2 - 4 (Peso:  5)
5 - 11 (Peso: 11)
5 - 8 (Peso: 23)
12 - 16 (Peso: 26)
13 - 15 (Peso: 33)
18 - 19 (Peso: 46)
12 - 17 (Peso: 48)
6 - 8 (Peso: 52)
14 - 16 (Peso: 61)
6 - 9 (Peso: 97)
19 - 20 (Peso: 112)
3 - 11 (Peso: 113)
13 - 14 (Peso: 165)
1 - 3 (Peso: 443)
```

Conclusão

Os dois algoritmos conseguem gerar árvores mínimas corretamente, mas seus comportamentos diferem na forma como adicionam as arestas à árvore. O algoritmo de Prim é mais eficiente em casos onde se deseja expandir gradualmente a árvore a partir de um vértice inicial, enquanto o algoritmo de Kruskal pode ser mais útil em casos onde a prioridade é adicionar as menores arestas possíveis, independentemente da conectividade imediata.