Atividade 12

Exercício 1

Passo	MST	Aresta	A	B	C	D	E	F
Inicial	{A}	-	0	4	2	∞	∞	∞
1	{A,C}	A-C (2)	0	4	0	∞	7	∞
2	{A,C,B}	A-B(4)	0	0	0	3	6	∞
3	{A,C,B,D}	B-D(3)	0	0	0	0	1	4
4	{A,C,B,D,E}	D-E(1)	0	0	0	0	0	4
5	{A,C,B,D,E,F}	E-F(4)	0	0	0	0	0	0

Arestas da MST: A-C, A-B, B-D, D-E, E-F
Peso: 14

Exercício 2

Ordem	Aresta	Peso	Adicionar?	Conjuntos
1	D-E	1	sim	{D,E}
2	A-C	2	sim	{A,C}, {D,E}
3	B-D	3	sim	{A,C}, {B,D,E}
4	A-B	4	sim	{A,B,C,D,E} {F}
5	E-F	4	sim	{A,B,C,D,E,F}
6	B-C	5	não	{A,B,C,D,E,F}
7	B-E	6	não	{A,B,C,D,E,F}
8	C-E	7	não	{A,B,C,D,E,F}
9	D-F	8	não	{A,B,C,D,E,F}

obs: mesmo peso total

Exercício 3

Passo	Aresta	Peso	Find(u)	Find(v)	Ciclo?	Union?	MST?
Inicial	-	-	-	-	-	-	{}
1	1-2	1	1	2	não	sim	{1-2}
2	2-3	2	1	3	não	sim	{1-2, 2-3}
3	1-4	3	1	4	não	sim	{1-2, 2-3, 1-4}
4	2-5	4	1	5	não	sim	{1-2, 2-3, 1-4, 2-5}
5	3-6	5	1	6	não	sim	{1-2, 2-3, 1-4, 2-5, 3-6}
6	4-5	6	1	1	sim	não	{1-2, 2-3, 1-4, 2-5, 3-6}
7	5-6	7	1	1	sim	não	{1-2, 2-3, 1-4, 2-5, 3-6}
8	1-5	8	1	1	sim	não	{1-2, 2-3, 1-4, 2-5, 3-6}

Para eu lembrar:

find(u): em qual conjunto o primeiro vértice da aresta pertence (na linha 4: find[2]: 1)

find(u): em qual conjunto o segundo vértice da aresta pertence (na linha 4: find[5]: 5)

pois {1,2, 2-3, 1-4, 2-5} {5}

Exercício 4

1) Prim pois possui poucas vértices e muitas arestas.

Passo	MST	Aresta	SP	BH	Cur	Por	Bra	Rio
Inicial	{SP}	-	0	52	38	∞	95	45
1	{SP, Cur}	SP-Cur(38)	0	52	0	42	95	45
2	{SP, Cur, Por}	Cur-Por(42)	0	52	0	0	85	45
3	{SP, Cur, Por, Rio}	SP-Rio(45)	0	40	0	0	85	0
4	{SP, Cur, Por, Rio, BH}	Rio-BH(40)	0	0	0	0	68	0
5	{SP, Cur, Por, Rio, BH, Bra}	BH-Bra(68)	0	0	0	0	0	0

Custo total: 38+42+45+40+68 = 233

Exercício 5:

Critério	Prim	Kruskal
Estratégia Principal	Cresce uma árvore a partir de um vértice inicial e vai escolhendo as arestas mais baratas (sem mudar as que já foram)	Seleciona as arestas em ordem crescente e adiciona-as se não fizerem um ciclo
Estrutura de Dados	Heap/Priority Queue	Union-find
Complexidade (com heap/UF)	O(E log V)	O(E log E)
Tipo de Grafo	Conexo e ponderado	Ponderado
Ordenação Necessária	Não ordena todas arestas	Ordena por peso (crescente)
Melhor para Grafos Densos	Sim	Não
Melhor para Grafos Esparsos	Não	Sim
Escolha do Vértice Inicial	Necessário	Não

Exercício 6:

1. **Unicidade da MST: Se todas as arestas têm pesos distintos, a MST é única? Justifique.**

Sim, porque não tem arestas com o mesmo peso.

2. **Ciclo com Aresta Máxima: Prove ou refute: Se adicionarmos uma aresta (u,v) com peso w a MST, criamos um ciclo. A aresta de maior peso neste ciclo tem peso w.**

Suponha que nesse ciclo exista uma aresta mais pesada que w.
Se isso fosse verdade, poderíamos trocar essa aresta pesada pela nova aresta (u,v)(u,v)(u,v), deixando a árvore mais leve, é impossível porque a MST já é mínima. Verdadeiro.

3. **Corte Mínimo: Explique como a propriedade do corte garante a correção do algoritmo de Prim.**

Porque ele sempre escolhe a aresta de menor peso naquele momento (entre os vértices mst).

4. **Número de Arestas: Quantas arestas possui a MST de um grafo conexo com n vértices?**

n-1 arestas.

Exercício 7

Caso 1: Árvore Geradora Máxima

Como você modificaria os algoritmos de Prim e Kruskal para encontrar uma árvore geradora *máxima* (maximum spanning tree)?

Em vez de priorizar a menor aresta, eu priorizaria a maior, ou seja, em Prim eu escolheria o maior peso e em Kruskal, faria a tabela em ordem decrescente.

Caso 2: MST com Restrições

Suponha que você precise construir uma MST, mas a aresta (B,C) deve obrigatoriamente fazer parte da solução. Como você adapta o algoritmo?

Eu começaria incluindo essa aresta (mesmo se não fosse o menor peso), em Prim eu começaria no vértice B e já iria para o C, já no Kruskal, a primeira linha de aresta seria b-c.

Caso 3: Floresta Geradora Mínima

Em um grafo desconexo com 3 componentes conexas, o que os algoritmos de Prim e Kruskal produzem?

Kruskal teria 3 árvores geradoras mínimas (msf) e em Prim produziria apenas a MST do vértice inicial ou teria que ter 2 vértices “iniciais”.