# PROVA DE ESTRUTURAS DE DADOS II

| INSTITUTO FEDERAL Catarinense Campus Blumenau | | NOTA |
|---|---|---|
| | Curso: Bacharelado em Ciência da Computação | 4,5 |
| | Professor: Ricardo de la Rocha Ladeira | |
| | Aluno (a): Giovani Zanella da Maia | |

**ORIENTAÇÕES**

- *Questões de múltipla escolha contêm apenas uma alternativa correta.*
- *Questões discursivas devem estar na forma mais completa possível.*
- *Respostas rasuradas não serão consideradas.*

**1)** (0,5) Desenhe um grafo cubo $Q_4$.

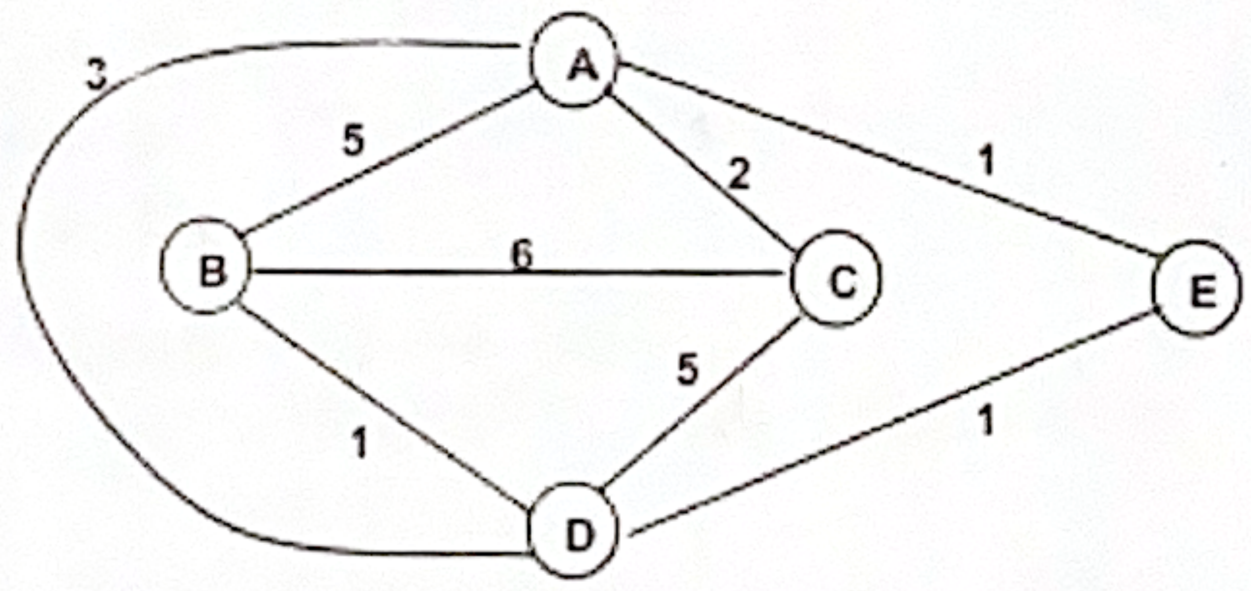
**2)** (1,0) Para cada um dos grafos abaixo, diga se é Euleriano, Hamiltoniano e justifique (comprove) sua resposta.

a) (0,5) Grafo completo $K_{50}$.

b) (0,5) Grafo bipartido completo $K_{20,21}$.

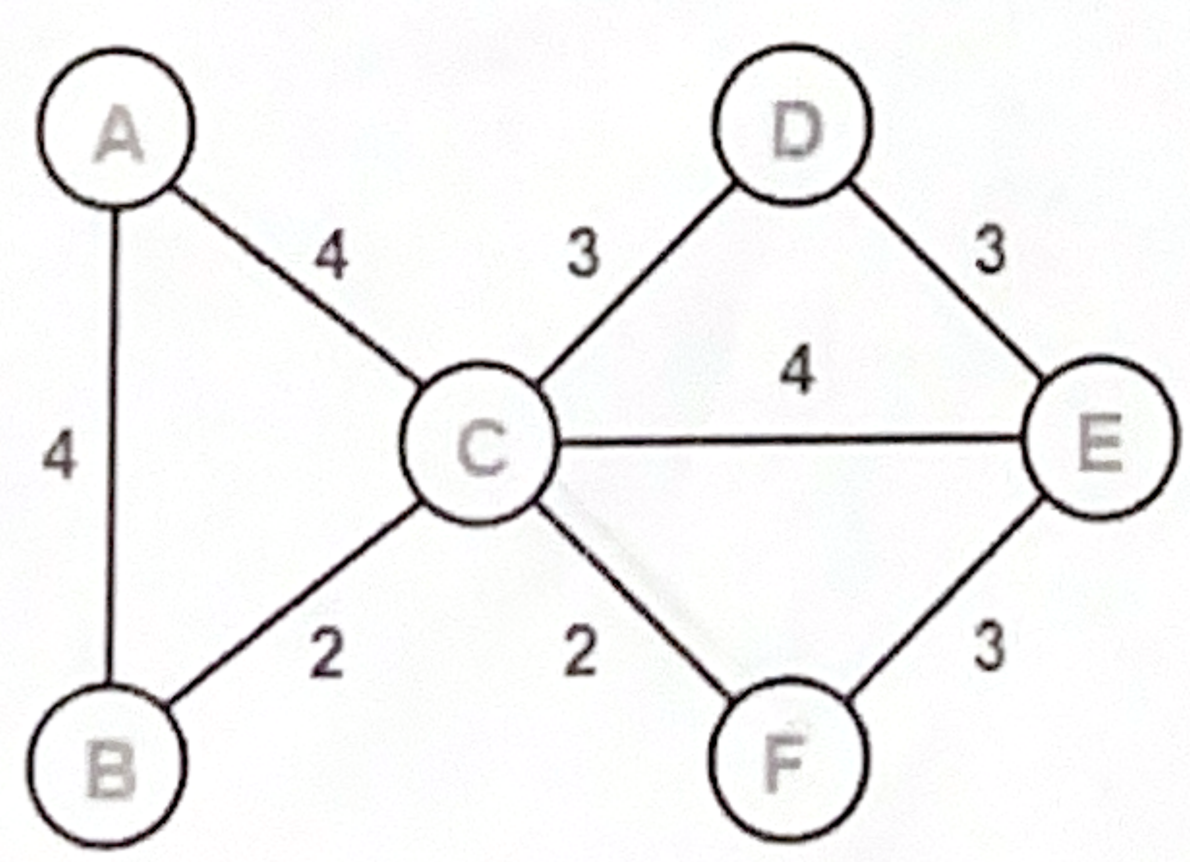
0,1

**3)** (1,0) Encontre a solução do *problema do carteiro chinês* para o grafo do problema abaixo. Mostre o(s) caminho(s) artificial(is) e calcule o custo total, isto é, faça a *eulerização* do grafo com custo mínimo.




**4)** (1,0) Utilize o Algoritmo de Kruskal para gerar a Árvore Geradora Mínima (AGM, ou *Minimum Spanning Tree* – MST) do grafo a seguir. Liste as arestas selecionadas e seus respectivos pesos em ordem crescente, detalhando o processo de inclusão de cada aresta. Ao final, apresente o custo total da MST.

CX




**5)** (1,0) Resolva o mesmo problema da questão 4, mas utilizando o algoritmo de Prim.

**6)** (1,0) Explique, com o máximo de detalhes, o funcionamento do algoritmo de Floyd através de um exemplo.

CX

**7)** (1,0) O problema *Número de Ilhas* é descrito da seguinte forma: dado um *grid* de dimensões *m×n*, representado por uma matriz onde cada elemento pode ser *1*, indicando uma massa de terra, ou *0*, representando o oceano. O objetivo é determinar a quantidade de ilhas, ou seja, o número de agrupamentos contíguos de células com valor 1. Para isso, consideram-se como adjacentes apenas as células vizinhas horizontal e verticalmente (não são considera-das conexões diagonais).

```
let ilhas = [
[1, 1, 0, 0, 1, 0, 1],
[0, 1, 0, 1, 1, 0, 1],
[0, 0, 0, 0, 0, 0, 0],
[0, 1, 1, 0, 0, 0, 0],
[1, 0, 1, 0, 0, 1, 0],
[1, 0, 0, 0, 1, 1, 0]
]
```

No exemplo acima há seis ilhas.

Que algoritmo de travessia em grafos poderia ser utilizado para resolver o problema? Explique detalhadamente a sua proposta de solução.

**8)** (1,0) Sobre grafos, marque a opção correta:

a) Um grafo ponderado é um grafo não direcionado em que todos os pares de vértices são adjacentes, ou seja, há uma aresta entre cada par de vértices.

b) Todo grafo completo possui pesos associados a todas as suas arestas.

c) Um caminho em um grafo é considerado complexo se todos os vértices ao longo do caminho forem distintos.

d) O grau de um vértice em um grafo não direcionado é o número de arestas que incidem sobre ele.

e) Se existir um caminho *p* que conecte *i* a *j*, então *i* é alcançável a partir de *p* através de *j*.

9) (0,5) Analise o pseudocódigo da figura a seguir:

**Algoritmo:** SSotilli $(G, w, s)$

**para cada** $v \in V$ **faça**
  $d_v \leftarrow \infty;$
  $\rho_v \leftarrow nil;$

$d_s \leftarrow 0;$
$S \leftarrow \emptyset;$
$Q \leftarrow V;$

**enquanto** $Q \neq \emptyset$ **faça**
  $u \leftarrow REMOVE - MINIMO(Q);$
  $S \leftarrow S \cup \{u\};$
  **para cada** *vértice* $v \in adj(u)$ **faça**
    **se** $v \in Q$ *e* $[d_v > (d_u + w_{u,v})]$ **então**
      $d_v \leftarrow (d_u + w_{u,v});$
      $\rho_v \leftarrow u;$

Resposta

A PRIMEIRA LINHA SELECIONADA APENAS TESTA. O CASO EM NEGRITO OCORRE APENAS SE A CONDIÇÃO É VERDADEIRA

O algoritmo `SSotilli` possui uma operação conhecida como *relaxamento de aresta*. Quando um vértice *u* é considerado "encerrado", todos os seus vértices adjacentes *v* são verificados para fazer o relaxamento de aresta. Isto significa que, ao se calcular o caminho mínimo até *u*, verifica-se a possibilidade de haver, para o vértice *v*, um caminho passando por *u* que seja mais barato do que o caminho atual para *v*. Neste caso **o custo do caminho para *v* é atualizado, bem como seu predecessor na árvore de caminhos mínimos, que passa a ser o vértice *u*.**

a) (0,25) Que algoritmo é esse? Dijkstra C

b) (0,25) Indique em que parte do pseudocódigo acontece o trecho formatado em negrito. 0,1 X

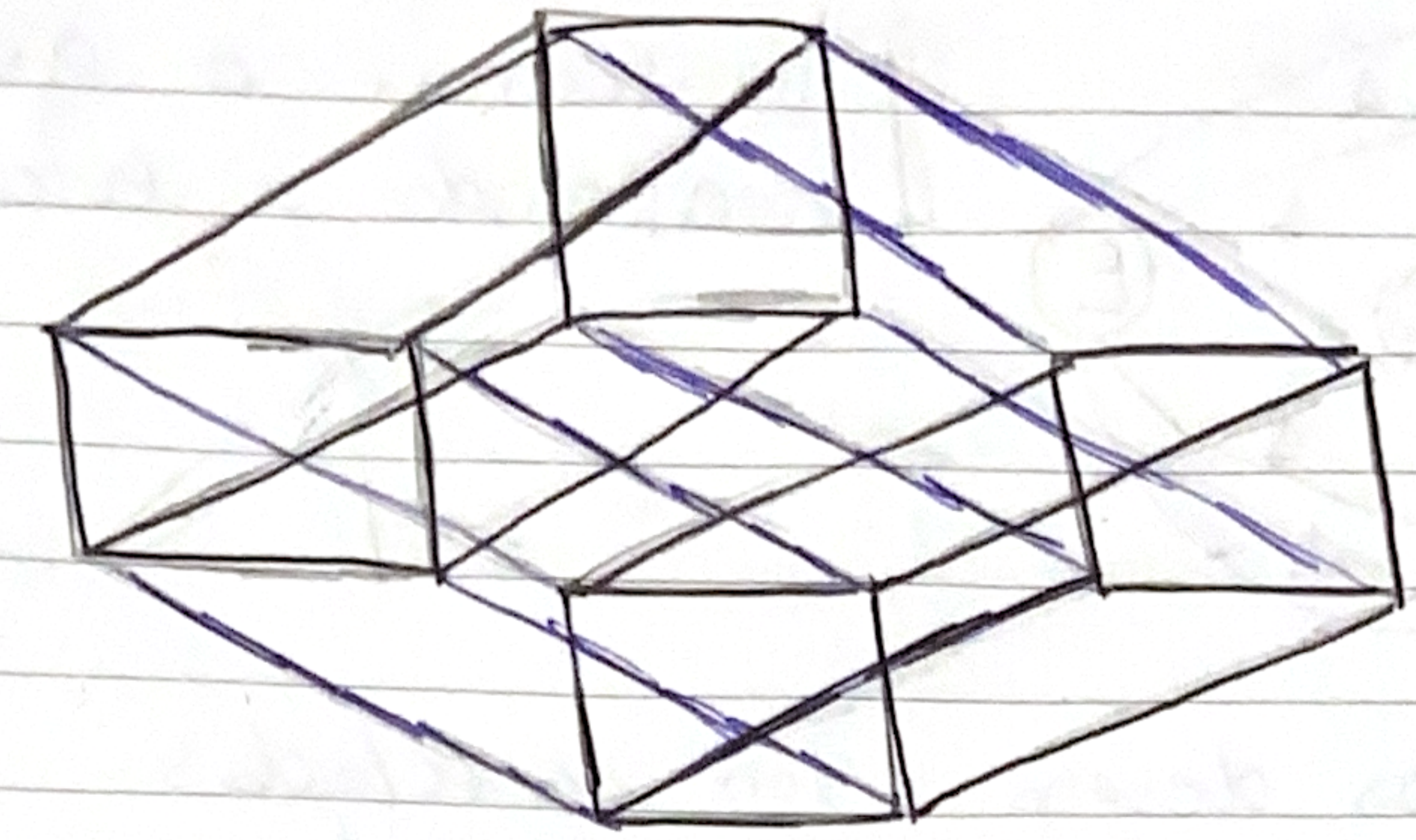
**10)** (2,0) Resolva algum problema do Beecrowd de qualquer categoria, desde que sua solução use grafos. Envie o código e uma captura de tela comprovando o aceite.

**Obs.1**: O problema resolvido deve ter nível mínimo 3 e pontuação mínima 2,5.

**Obs.2**: O código deve estar com o máximo de comentários possível, explicando em detalhes a solução, a estrutura utilizada para a representação do grafo e toda lógica envolvida no código.

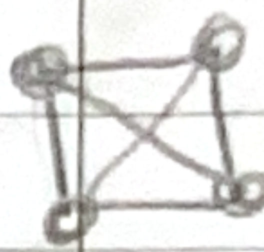
Giovani Zanella

1)



a partir dos 2 Grafos cubos $Q_3$ em preto, foi construido o $Q_4$

2) Um grafo ~~completo~~ é um grafo simples onde cada par de vertices é adjacente

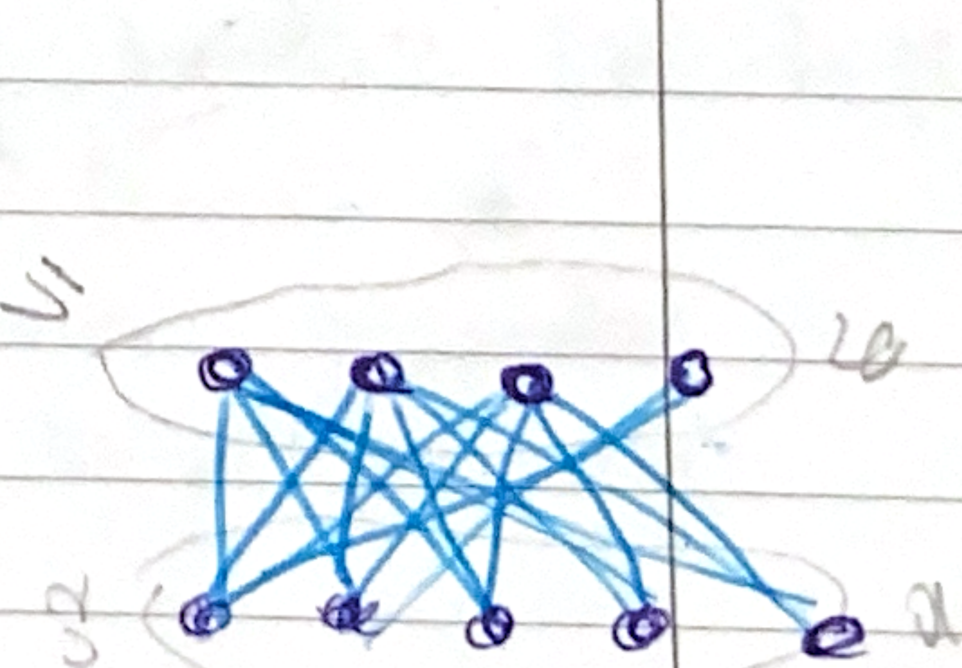


3 saida ao HO

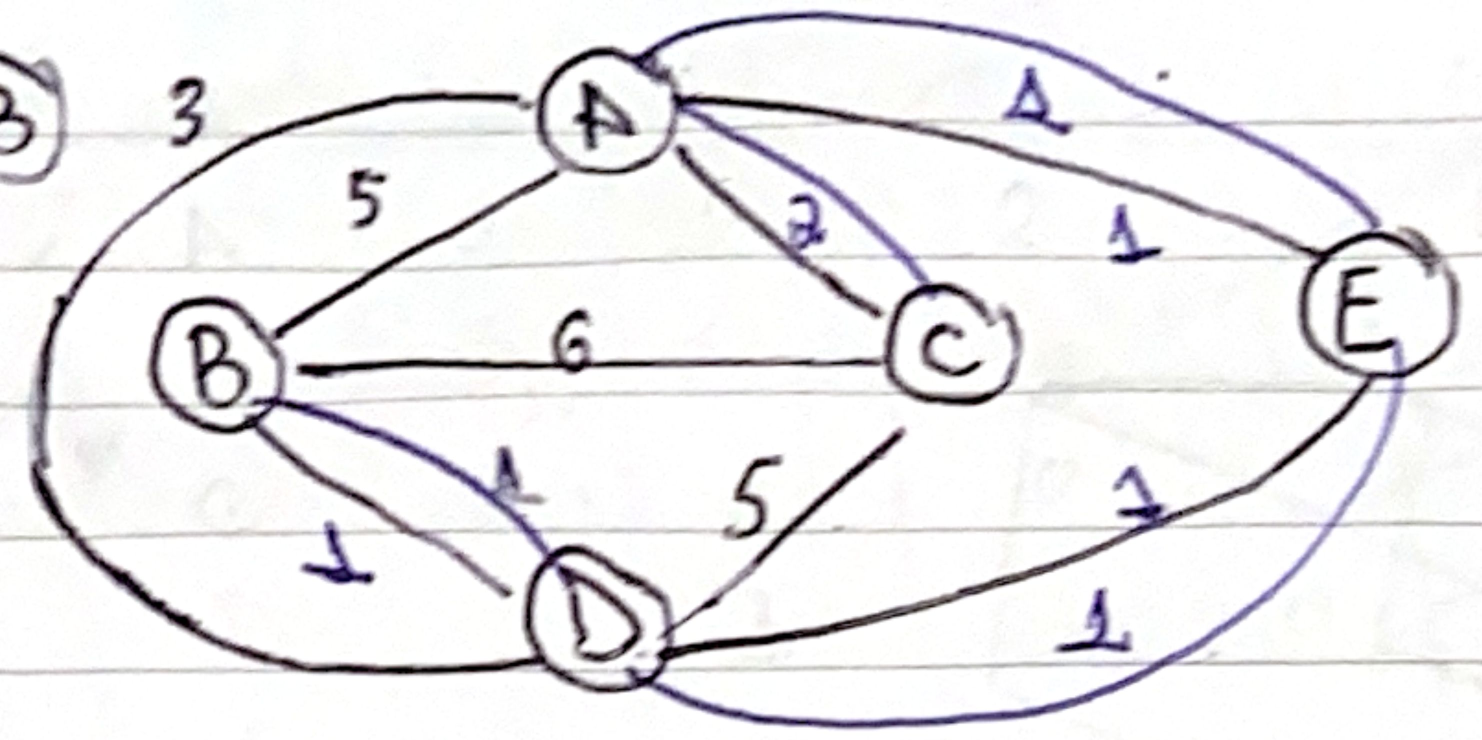
canalagria

olhando para o grafo que é um grafo completo e semelhante ao $K_{50}$, para ele ser euleriano precisa percorrer por todas as arestas somente 1 vez e Hamiltoniano precisa percorrer por todos os vertices. O grafo $K_{50}$ não é euleriano porque cada vertice fará 49 ligações (ele não é ligado com ele mesmo), tendo um numero impar de ligações, provando não ser Euleriano. É Hamiltoniano pois constitui em passar em todos os vertices 1 vez e retorna a origem.

b) Grafo bipartido completo, temos 2 conjuntos de vertices que são ligados por aresta do conjunto de vertice oposto, ou seja não ligamos vertices do mesmo grupo. Não será euleriano pois o conjunto V1 = 20 vertices, cada V desse conjunto vai ter uma ligação com os vertices do conjunto V2 = 21 vertices ou seja terá um numero impar de arestas. também Não é Hamiltoniano pois ele acaba passando mais de uma vez em um mesmo vertice.



③




tranformar o grafo realizando a eulerização

1°) todas as arestas devem estar conectadas
2°) verificar os vertices impares e adicionar arestas [B, A, D, E]

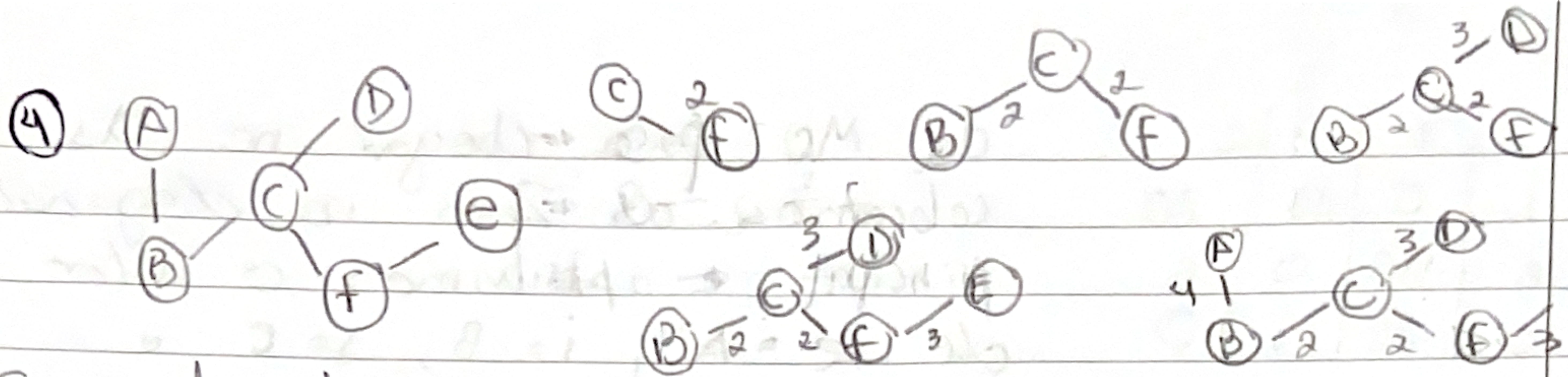
Custo total = somar arestas = 29

8) D) o grau de um vertice não direcionado é o número de aresta que incidem sobre ele.

9) a) Algoritimo de dijkstra

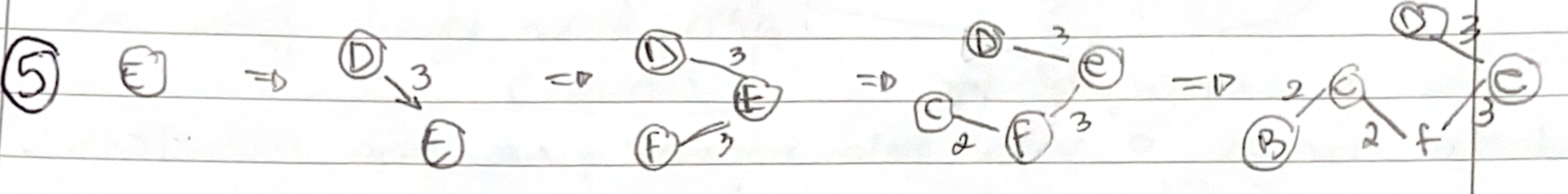
b) $d_v \leftarrow (d_u + w_{u,v})$;
$p_v \leftarrow u$;

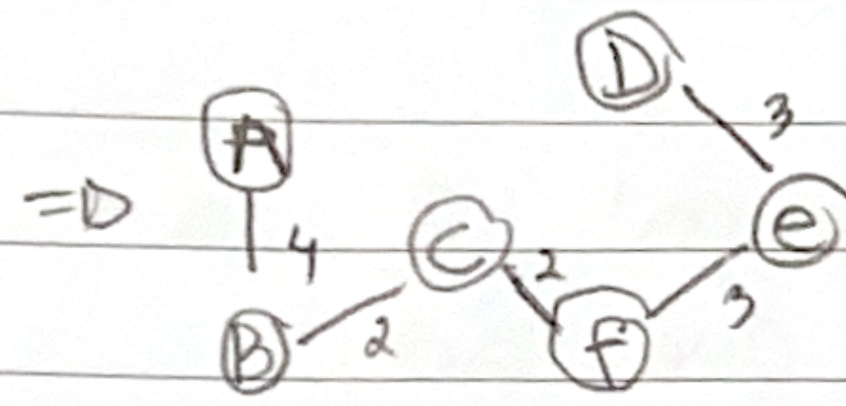
4)



Somando tudo custo fica 14

5)

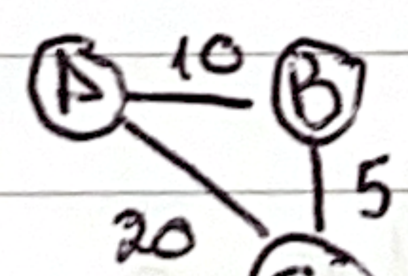




Partindo do vertice E
Escolher o caminho menos custoso em relação ao vertices não visitados
Custo total = 14

6) objetivo é encontrar o menor caminho entre todos os vertices do grafo, verifica se a distancia de origem ate o destino é menor do que ir da origem até um ponto intermediario K, e deste ponto ir até o destino.
É representada por uma matriz de distância que é atualizada em etapas para refletir os menores custos entre os pares de vertices, sua complexidade é $O(V^3)$ onde V é o numero de vertices. Ele é adequado para grafos com pesos negativos, mas sem circulos negativos.

Ex)



Destino

| origem | A | B | C |
|---|---|---|---|
| A | 0 | 10 | 20 |
| B | 10 | 0 | 5 |
| C | 20 | 5 | 0 |

para construir a matriz inicial $M_0$, Podemos definir que a coluna é a origem e a linha é o destino, os valores de matriz são extraidos do Grafo.
vemos que de A para B tem posso 10, entao partindo da origem A colocamos o valor na matriz. De A p/ C o peso é 20, e de B p/ C é 5

obs) Diagonal principal sempre receberá o valor zero, a menos que o vertice ligue com ele mesmo

| M1 | A | B | C |
|---|---|---|---|
| A | 0 | 10 | 20 |
| B | 10 | 0 | 5 |
| C | 20 | 5 | 0 |

da M0 para chegar na M1, colocamos os Zeros na diagonal principal e atribuimos o valor de $k = A$, $i = B$, $j = C$ e utilizamos na inequacão

$d[B][C] > d[B][A] + d[A][C]$ , aplicar o valor que estiver na linha B e coluna C, da M0, fazer igual para as demais

$5 > 10 + 20$

~~20 > 15~~

se a inequação for verdadeira troca o valor pelo menor, se não mantem. no caso é falso então mantemos o 5.

Para Gerar M2, mantemos diagonal principal e a linha 2 e coluna 2, os valores vazios que são [A][C] e [C][A] usamos a inequacão alterando o ponto intermediario p/ $k = B$ → partimos de M1

| | A | B | C |
|---|---|---|---|
| A | 0 | 10 | 15 |
| B | 10 | 0 | 5 |
| C | 15 | 5 | 0 |

$d[A][C] > d[A][B] + d[B][C]$

$20 > 10 + 5$ , é vdd trocamos por 15

Para Gerar M3 a partir de M2, mantemos diagonal pricipal e linha e coluna 3. O ($k = C$) e verificamos a distancia [A][B] e [B][A]

| | A | B | C |
|---|---|---|---|
| A | 0 | 10 | 15 |
| B | 10 | 0 | 5 |
| C | 15 | 5 | 0 |

$d[A][B] > d[A][C] + d[C][B]$

$10 > 15 + 5$ é falso então mantem.

Obs: Esqueci de mencionar quando ~~todos~~ os vertices não possuem ligações entre se colocamos "∞" na matriz