

01. (5 pontos) Processo de software é como uma metodologia para as atividades, ações e tarefas necessárias para desenvolver software de alta qualidade. Dessa forma, um processo de software é como uma série de passos previsíveis ou um roteiro, que ajudará na criação de um produto ou sistema de alta qualidade e dentro do prazo estabelecido entre as partes.

O modelo em Cascata é o paradigma mais visto e mais amplamente empregado na engenharia de software. A figura ilustra o modelo em Cascata e seus principais estágios.

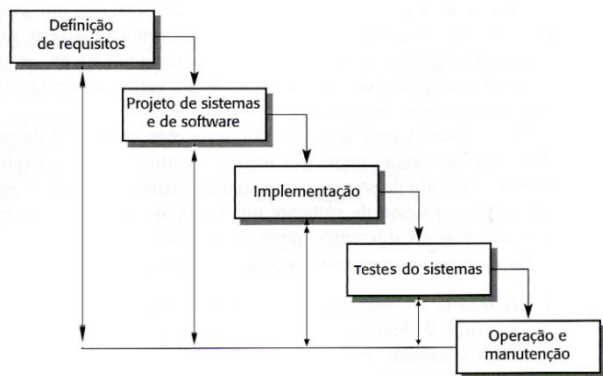


Figura – Modelo em Cascata.

De forma sucinta, disserte sobre o modelo em Cascata, comentando suas fases, aplicabilidade, os problemas que podem surgir na sua aplicação, as vantagens e as desvantagens desse modelo.

02. (5 pontos) Observe o modelo EER (Extended-ER ou Enhanced-ER) desenvolvido com a ferramenta visual para design MySQL Workbench na Figura.



Figura – Diagrama EER.

Com base no diagrama EER da figura, escreva o código SQL de criação das tabelas, levando em conta os tipos de dados e as seguintes “constraints”:

- Todas as chaves primárias (PK, *primary key*) devem ser autoincrementáveis;
- A *constraints* PK da tbFuncionarios deve ser identificada como “pk**tbFuncionarios**”;
- A *constraints* PK da tbSetor deve ser identificada como “pk**tbSetor**”;
- A *constraints* FK da tbFuncionarios deve ser identificada como “fk**tbFuncioSetor**”;
- Os campos nomeFuncion e nomeSetor devem ser *not null*.

Apresente os comandos SQL, para inserir os seguintes registros nas respectivas tabelas.

tbFuncionarios			
codFuncion	nomeFuncion	salarioFuncion	tbSetor_codSetor
1	Pedro	R\$ 2500,00	1

2	Maria	R\$ 3000,00	2
3	Lucas	R\$ 2800,00	2
4	Francisca	R\$ 1500,00	3
5	Jorge	R\$ 1500,00	1
6	João	R\$ 500,00	1

**Tabela 1 – Dados da tabela tbFuncionarios.**

tbSetor	
codSetor	nomeSetor
1	Desenvolvimento
2	Manutenção
3	RH
4	Gerência

**Tabela 2 – Dados da tabela tbSetor.**

**03. (5 pontos)** Existem vários cenários que podem levar um usuário a querer mais de um sistema operacional (S.O.), instalado em seu computador, por isso existe o recurso *multi boot*. *Multi boot* é um sistema que permite a escolha de um entre vários sistemas operacionais instalados num mesmo microcomputador, quando ligado. Normalmente, é chamado de *dual boot* pelo fato de que, na maioria dos casos, possui 'dois' sistemas operacionais.

Suponha que um usuário iniciante acabou de comprar um microcomputador com as seguintes especificações: PC com processador: Intel Core i7 de 3,8 GHz e Cache L3 de 8 MB, memória RAM DDR3 de 8 GB, HD SATA com capacidade de 1 TB e 7200 rpm, monitor LCD de 18,5", placa de vídeo Geforce GTX 750 com 2GB DDR5, mouse, teclado e as caixas de som.

Por curiosidade, o usuário deseja instalar os sistemas operacionais Windows 7 Profissional 64 bits (Português – PT-BR) e o Linux Ubuntu 16.04 LST, no mesmo HD, em *dual boot*, com metade do espaço do HD para cada SO, mas ele tem receio de danificar seu computador.

Para facilitar a vida do usuário leigo, desenvolva um *checklist* básico, enumerando um conjunto de condutas, itens ou tarefas que devem ser lembradas ou seguidas, levando em consideração que esse processo exige alguns conhecimentos técnicos, como alteração de sequência de boot, particionamento do HD (quantidade de partições, tamanho das partições e tipo de partição para cada SO) e ordem de instalação.

**04. (5 pontos)** A partir das informação abaixo:

- IP: 192.168.100.2
- CIDR (de *Classless Inter-Domain Routing*): 24

Calcule os dados para uma rede TCP/IPv4:

- IP em bits:
- Rede:
- Rede em bits:
- Máscara de sub-rede:
- Máscara de sub-rede em bits:
- Broadcast:
- Broadcast em bits:
- IPs em toda a rede:
- IPs disponíveis para uso:
- Primeiro IP utilizável:
- Último IP utilizável:

OBS.: resolva os item de (a) a (k), explicando como você chegou a esse resultado. Nos itens que exigem operações matemáticas, apresente os cálculos de maneira organizada e legível.

05. (5 pontos) Desenvolva um ALGORITMO em pseudocódigo, C/C++, Pascal, Java, Python ou Ruby, que leia três valores inteiros  $le$ ,  $ld$ ,  $li$  e verifique se eles podem ser os comprimentos dos lados de um triângulo. Caso seja verdade, exibir a mensagem: "Podem formar um triângulo!". O algoritmo também deve verificar se é um triângulo equilátero, isósceles ou escaleno e exibir a mensagem "Tipo Equilátero", "Tipo Isósceles" ou "Tipo Escaleno". Se eles não formarem um triângulo, escrever a mensagem "Não é um triângulo!".

Antes da elaboração do algoritmo, torna-se necessária a revisão de algumas propriedades e definições.

- **Propriedade** – O comprimento de cada lado de um triângulo é menor do que a soma dos comprimentos dos outros dois lados; fora desta condição, as medidas não são consideradas de um triângulo;
- **Definição 1** – Chama-se triângulo equilátero o que tem os comprimentos dos três lados iguais;
- **Definição 2** – Chama-se triângulo isósceles ao triângulo que tem os comprimentos de dois lados iguais;
- **Definição 3** – Chama-se triângulo escaleno o que tem os comprimentos dos três lados diferentes (vide figura).

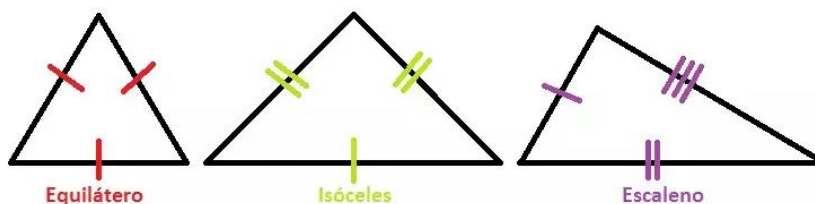


Figura – Tipos de triângulos.