Project requirements

O projeto da formação é elaborar um gestor de inventário modular eficiente e versátil.

O objetivo é terem um programa capaz de gerir a informação sobre diversos itens num armazém de distribuição.

O armazém é composto por fileiras, que contém prateleiras, cada prateleira tem várias zonas. Cada zona tem apenas 1 item.

Podemos ter N fileiras, prateleiras e zonas. A numeração começa em 0 e podemos assumir que quanto maior for, mais longe está o local da base. Ou seja, a fileira 0 está mais próxima da base e dentro da fileira, a prateleira 0 segue a mesma regra e assim adiante.

Em cada zona iremos guardar um item, que irá ter pelo menos as seguintes características:

- Identificador Numérico
- Nome
- Quantidade
- Qualidade

Cada item tem a sua quantidade independente que não pode ser alterada (paletes que só são tratadas como um todo) mas podemos ter vários do mesmo item no mesmo armazém (identificador numérico e nome igual) e temos de ser capazes de para cada item encontrar todas as suas ocorrências.

A qualidade de um item está representado em 3 categorias e cada categoria tem características particulares:

- Frágil: As características de itens frágeis são: Data de validade e distância da base. Devido à sua fragilidade, devem ser guardados no máximo na fileira X, informação que deve ser incluída na categoria.
- Oversized: Itens oversized ocupam várias zonas. A sua característica é quantas zonas contíguas precisa para ser armazenado.
- Normal: Itens normais não possuem características extra.

Deve também armazenar a timestamp em que o item foi colocado no armazém.

O nosso programa deve ser também capaz de encontrar um local para um novo item que chegue.

Para a alocação de lugares no armazém, devemos aceitar várias estratégias, que sigam todas o mesmo contrato de alocação que para um dado item e estado do armazém retorne uma zona livre e correta para o item. Deve ter duas estratégias implementadas e devemos poder, no código, com apenas uma alteração trocar entre elas:

- A colocação dos itens no armazém deve sempre tentar ocupar a zona mais próxima da entrada antes de ir mais longe.
- A colocação deve seguir um estilo round robin, ignorando lugares que possam ter sido abertos com a saída de itens entretanto.

Por fim, o nosso programa deve ter a capacidade de manter uma lista com filtros para decidir se certos itens podem entrar no nosso armazém. Para designar os filtros, devemos criar uma trait com um predicado que aceite o estado do armazém e o item.

A nossa lista deve ser capaz de guardar diversas implementações de filtros que possamos criar. Antes de adicionar os itens ao nosso armazém, teremos que verificar que eles são de facto aceites pela nossa lista de filtros.

O nosso programa deve também ser capaz de, eficientemente, fazer:

- Uma busca para ver se temos itens com um certo ID e retornar se sim/não e quantos.
- Uma busca para ver se temos itens com um certo nome, retornando um resultado igual à busca anterior.
- Uma busca que retorne a zona (incluindo fileira e prateleira também) onde está guardado um determinado item. Se houverem vários itens desse tipo guardados deve retornar todos
- Alocação de local para um item que chegou.
- Colocação de um item no seu local atribuído.
- Remoção de um item da sua zona.
- Uma busca para ver todos os itens que temos armazenados naquele momento, ordenados por nome ascendente.
- Dado um dia, verificar quantos itens estariam a 0 ou menos dias de atingirem o prazo de validade (ou se já o passaram).

Nota: Devem utilizar estruturas de dados para atingir a eficiência pretendida. Buscas com complexidade lineares não irão ser consideradas como 100% certas pois a sua eficiência não é escalável. Se tiverem dúvidas sobre a complexidade de alguma implementação por favor contactem.