UNIVERSIDADE ABERTA
www.uab.pt

Ministério da Educação e Ciência

e-Fólio

U.C. 21077

Linguagens de Programação e-Fólio A – Linguagem OCaml

-- INSTRUÇÕES --

1) O e-fólio tem uma cotação de 4 valores.

2) Qualquer tentativa de plágio resultará numa nota final de zero valores.

3) Este e-fólio deve ser resolvido usando a *linguagem Java e OCaml*.

4) Deve ser submetido um ficheiro comprimido (ZIP ou RAR) com o nome e número de estudante contendo:

a) Código do programa;

b) Ficheiro readme.txt com a informação necessária para compilar e executar o programa;

c) Relatório em formato *.pdf* até 4 páginas descrevendo a solução apresentada e os testes efetuados

E-fólio A

Você recebeu uma proposta mágica que não pôde recusar! Um amigo seu, que conhece a sua habilidade em programação, recomendou-lhe ao dono da "Enchanted Emporium", uma loja especializada em produtos mágicos. Impressionado com as suas habilidades, o dono da loja, Manuel Mago contratou-o para desenvolver um sistema de gestão de carrinho de compras único para a sua loja.

Você está encarregue de desenvolver um sistema de gestão de carrinho de compras para esta loja peculiar. O sistema precisa de calcular descontos com base em diferentes aspetos: as categorias de itens no carrinho, a lealdade do cliente, os custos de envio e os custos totais do carrinho. Para isso, você utilizará a linguagem OCaml para implementar as funções de backend e a linguagem Java para o frontend (iteração com o utilizador).

O Sr. Mago entregou-lhe uma base de dados feita por um antigo funcionário "store.pl", com os descontos por categoria a serem aplicados, os descontos de lealdade por ano, e os custos de envio por distrito, ao que parece é uma base de dados em Prolog, mas depois da sua cuidada análise do que foi pedido pelo Sr. Mago, você concluiu que não precisa de saber nada de Prolog para fazer este trabalho. Pois você concluiu que, por exemplo:

- *item(1, "Potion of Healing", "potions", 10.0, 50).* significa item de id 1 chamado"Potion of Healing" da categoria "potions" com o custo de 10 Euros e existem 50 em inventário;
- *discount(*potions*, 0.1).* significa que poções tem desconto de 10%;
- *loyalty_discount(3, 0.15).* significa que 3 anos de lealdade tem desconto de 15%;
- *shipping_cost("Aveiro", 5.0).* significa que o custo de envio para Aveiro é de 5 Euros.

A base de dados completa encontra-se em anexo no ficheiro *store.pl*.

Para ajudá-lo, o Sr. Mago fez uma pequena magia e você recebeu um arquivo de OCaml chamado *main.ml* já com as funções criadas para ler a base de dados, guardando os elementos numa lista.

Sinta-se privilegiado por ter sido escolhido para desenvolver um sistema tão mágico para a "Enchanted Emporium". A magia está em suas mãos agora - boa sorte!

1. A Loja

Como seu primeiro desafio, você deve criar uma classe principal chamada Store em Java.
Dica: *uma loja tem clientes, dê uma olhada na questão 2*

2. Os Clientes

Em seguida, crie uma classe Cliente com atributos como id, cidade, distrito e anos de lealdade. Crie um cliente fictício, para os seus testes.
Dica: *Cada cliente terá seu próprio carrinho de compras.*

3. O Carrinho

Crie uma classe Cart com os atributos apropriados. A classe Cart representará um carrinho de compras e terá atributos apropriados para armazenar os itens adicionados pelo cliente. Crie um carrinho fictício para o cliente, para os seus testes.

4. O cálculo do preço total sem descontos

Implemente uma função em OCaml para calcular o preço total do carrinho sem os descontos aplicados. A função receberá uma lista de itens do carrinho e calculará o preço sem qualquer desconto ou custos de envio.

Exemplo de uma lista de itens: *"1;Potion of Healing;potions;2,3;Enchanted Spellbook; enchanted_books;1" neste caso o id, nome do item, categoria, quantidade.*

Dica: *terá que criar uma função para ler a string recebida e transformar os items em lista ou no formato necessário em OCaml*

4. O cálculo dos descontos por categoria

Agora, você precisará implementar uma função em OCaml para calcular os descontos dos itens no carrinho com base nas suas categorias. A função receberá uma lista de itens do carrinho e calculará o desconto total com base na categoria de cada item.

Dica: *você deve ter alguma função em OCaml para ler os descontos de store.pl*

6. O cálculo dos descontos por lealdade

Implemente uma função em OCaml para calcular o desconto de lealdade. A função receberá o número de anos que um cliente está na loja e o valor total do carrinho de compras, considerando os descontos de categoria.

7. O Custo de Envio

Para garantir que todos os itens chegam aos clientes com segurança, implemente uma função em OCaml para calcular os custos de envio com base no distrito do cliente.

8. O cálculo do preço final do carrinho

Agora com todos os elementos finalmente, implemente uma função em OCaml para calcular o preço final do carrinho com todos os descontos aplicados.

9. A exibição do carrinho de compras

Para uma experiência de compra mágica, crie uma função em OCaml para exibir corretamente o carrinho de compras no sistema. A função receberá uma lista de itens do carrinho e ordena-los-á primeiro por categoria e depois pelo nome do item em cada categoria. E retornará a lista ordenada.

10. Implementação das Operações

Para concluir, crie uma classe em Java para operações de carrinho, com métodos para acessar a cada uma das funções implementadas em OCaml. Utilize subprocessos Java para chamar o encantamento OCaml e realizar os cálculos necessários.

Notas:

1. (C3) Todas as escolhas devem ser fundamentadas no relatório.
2. (C1) A forma de implementar os vários módulos propostos.
3. (C2) A forma e lógica do programa fica ao critério de cada estudante, mantendo as boas práticas de programação.
4. (C2) A facilidade de utilização do programa é valorizada (exemplo: menus de acesso, e outras estruturas e termos complexos)