U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

MESTRADO INTEGRADO EM ENGENHARIA INFORMÁTICA E COMPUTAÇÃO | 1º ANO
EIC0012 | PROGRAMAÇÃO | 2018/2019

TRABALHO PRÁTICO Nº 2

Agência de Viagens "NiceHolidays"

INTRODUÇÃO

A agência de viagens **NiceHolidays** comercializa diversos tipos de pacotes de viagens para diferentes tipos de público-alvo. A empresa pretende desenvolver um programa informático para gerir a sua atividade no setor do turismo. A sua atividade envolve a gestão de **clientes** e de **pacotes turísticos**.

Uma **agência** é caraterizada pelos seguintes atributos: nome, número de identificação fiscal (NIF), morada, URL do seu *web site*, lista de pacotes turísticos que comercializa, e lista de clientes.

Cada **cliente** é caraterizado por: nome, NIF, número de pessoas no agregado familiar, morada, lista de pacotes turísticos comprados, e total de compras efetuadas.

Um **pacote turístico** inclui: um identificador único, o(s) local(ais) turístico(s) a visitar, a data de início, a data de fim, o preço por pessoa e o número máximo de pessoas que podem inscrever-se no pacote.

Uma **morada** inclui: o nome da rua, o número da porta, o número do andar ("-" se não aplicável), o código postal e a localidade.

Uma **data** é representada por: ano, mês e dia.

O objetivo último do trabalho prático é desenvolver um programa que permita a uma agência de viagens: i) gerir pacotes de viagens (adicionar, alterar ou eliminar) ; ii) gerir clientes (adicionar novos clientes, vender-lhes um pacote turístico; ou eliminar clientes); iii) cálculo de estatísticas úteis para gestão da agência.

OBJETIVOS

Este trabalho prático tem como objetivo desenvolver um programa para a gestão da agência de viagens **NiceHolidays*.***

Este segundo trabalho é uma continuação do Trabalho 1 pelo que deve ser reutilizada uma grande parte do código desenvolvido no Trabalho 1. No entanto, neste trabalho será dada mais ênfase a novos aspetos da linguagem C++ que não foram utilizados no trabalho anterior. Em particular será obrigatória a utilização de um conjunto de classes adequadas à aplicação.

Para além dos conhecimentos treinados no primeiro trabalho, ao realizar este segundo, os estudantes terão oportunidade de aplicar os conhecimentos mais avançados de programação em C++ adquiridos até ao momento, e que incluem:

- definição e uso de classes;
- *overload* de funções;
- definição e *overload* de operadores;
- utilização de iteradores;
- algoritmos de ordenação;
- uso de diversos tipos de estruturas de dados da STL de C++ (*sets*, *maps, multimaps*, etc).

DESENVOLVIMENTO DO PROGRAMA

Definição de novos tipos de dados

Neste segundo trabalho os estudantes deverão substituir a utilização de estruturas (**struct**) associadas à definição de novos tipos de dados, utilizadas no trabalho anterior, por um conjunto de classes. Os métodos que manipulavam os dados armazenados nas *struct* serão agora parte dos membros função das classes que contêm a informação das "*struct* correspondentes". Em breve, será fornecido aos estudantes um ficheiro com um esboço inicial de classes adequadas a esta aplicação. Os estudantee têm toda a liberdade de rejeitar ou adaptar a estrutura de classes sugerida.

Funcionamento geral

O programa deve começar por ler o ficheiro **agency.txt** que contém: a informação estática da agência (nome, NIF, URL e morada), o nome do ficheiro onde está guardada a lista de clientes (ex: **clients.txt**) e o nome do ficheiro onde está guardada a informação dos pacotes turísticos (ex: **packets.txt**). Uma vez obtido o nome dos ficheiros onde está a informação de clientes e pacotes turísticos, estes devem ser lidos de seguida, de modo a completar a informação da agência relativa a clientes e pacotes turísticos. Cada item de informação no ficheiro **agency.txt** ocupa uma linha respeitando a seguinte ordem: nome, NIF, URL, morada, nome do ficheiro de clientes e nome do ficheiro de pacotes turísticos. Uma morada é guardada numa linha com a seguinte sintaxe: "rua / número da porta / número do andar / código postal / localidade" (ver exemplo na próxima secção).

No ficheiro de **clientes**, a informação de cada cliente é escrita a seguir à do anterior apenas separada por uma linha contendo a *string* **"::::::::::"** (10 carateres ':' consecutivos). Cada item de informação de cada cliente é escrito numa linha separada. A ordem dos itens/linhas de um cliente é: nome, NIF, número de pessoas no agregado familiar, morada, lista de *ids* dos pacotes turísticos comprados e valor total de compras efetuadas.

No ficheiro de **pacotes turístico**s, a informação de cada pacote é escrita a seguir à do pacote anterior apenas separada por uma linha contendo a *string* **"::::::::::"** (10 carateres ":" seguidos). Cada item de informação de cada pacote é escrito numa linha separada. A ordem dos itens/linhas de um pacote são: identificador único, data de início, data de fim, número máximo de pessoas, preço por pessoa, local. Uma data tem o seguinte formato: ano/mês/dia.

O formato dos ficheiros será ilustrado adiante.

Os dados lidos dos ficheiros de clientes e de pacotes, no início do programa, devem ser guardados nas estruturas de dados, internas ao programa, escolhidas para o efeito. Antes do fim do programa, o conteúdo dessas estruturas de dados deve ser escrito nos ficheiros respetivos, de forma a manter os dados atualizados.

Ficheiro de agências, de clientes e de pacotes turísticos

Os ficheiros utilizados no programa são ficheiros de texto que podem ser criados com um editor de texto simples como, por exemplo, o Notepad do Windows. O programa deve começar por perguntar ao utilizador qual o nome do ficheiro da agência a usar, terminando com um erro caso algum dos ficheiros indicados (agência, clientes e pacotes turísticos – os nomes dos dois últimos não são pedidos, estão nas 2 últimas linhas do ficheiro da agência) não exista.
NOTA**:** com os exemplos de ficheiros que se apresentam a seguir pretende-se simplesmente ilustrar a sintaxe dos campos guardados. Não há necessariamente consistência entre os seus conteúdos: os *ids* dos pacotes turísticos usados neste exemplo de ficheiro de clientes podem não existir no exemplo de ficheiro de pacotes turísticos mostrado.

Apresenta-se a seguir um possível conteúdo de um ficheiro da agência.

```
NiceHolidays
133331145
http://www.niceholiday.pt
Rua Sem Nome / 100 / - / 4400-345 / Porto
clients.txt
packets.txt
```

Exemplo inventado para a agência NiceHolidays

Cada uma das linhas do ficheiro contém a seguinte informação: o nome da agência, o NIF, o URL, a morada (elementos da morada separados pelo carácter '/'), nome do ficheiro contendo informação de clientes, e nome do ficheiro contendo informação sobre os pacotes turísticos.
Um conteúdo possível de um ficheiro de clientes é de seguida mostrado.

```
Rui Manuel
234987156
4
Rua Sem Fim / 200 / 5Esq /1200-001 / Lisboa
10 ; 36 ; 2
504
::::::::::
```

```
Belmiro Miguel
111987666
2
Avenida dos Grilos / 100 / - /2300-101 / Coimbra
100 ; 136 ; 20
1300
```
Exemplo de um ficheiro de clientes com 2 clientes

Cada uma das linhas do ficheiro contém a seguinte informação para cada cliente: nome do cliente, NIF, número de pessoas no agregado familiar, morada (elementos da morada separados pelo carácter '/'), lista dos pacotes turísticos que já comprou (elementos da lista separados por ponto e vírgula). Uma linha com 10 caracteres ':' consecutivos separa um cliente do seguinte.

Um conteúdo possível de um ficheiro de pacotes turísticos é de seguida mostrado.

```
3
1
Madeira – Funchal, Porto Santo
2019/08/01
2019/08/05
300
24
::::::::::
-2
Veneza
2019/03/02
2019/03/05
300
24
::::::::::
3
Douro vinhateiro - Porto, Régua, Pinhão, Vila Real
2019/09/21
2019/09/22
100
62
```
Exemplo de um ficheiro de pacotes turísticos com dois pacotes

A primeira linha indica o identificador numérico único que foi atribuído ao último pacote que foi criado. Cada uma das linhas de um pacote turístico contém a seguinte informação: identificador numérico único do pacote (positivo se o pacote estiver disponível, negativo se a pacote não estiver disponível), principal local turístico de destino (pode ser seguido de um hífen e de uma lista dos principais locais turísticos a visitar, separados por vírgula), data de início da viagem (elementos da data separados pelo carácter '/'), data de fim da viagem (elementos da data separados pelo carácter '/'), preço por pessoa, número de vagas disponíveis no pacote. Uma linha com 10 caracteres ':' consecutivos separa um pacote turístico do seguinte. Nota: neste caso, foram criados 3 pacotes turísticos, mas o pacote número 2 já não está disponível (tendo por isso um identificador negativo).

Funcionalidades a implementar

A aplicação que deve implementar neste trabalho inclui as seguintes funcionalidades:

1. Ler e guardar a informação da **agência**, dos **clientes** e dos **pacotes turísticos** armazenada em ficheiros.
2. Gerir os **clientes** e **pacotes turísticos**: criar, alterar e remover um **cliente**; criar, alterar ou colocar como indisponível um **pacote turístico** (nota: os pacotes turísticos nunca são efetivamente removidos; ver exemplo acima).
3. Gerar e visualizar de modo formatado a informação de um cliente especificado.
4. Gerar e visualizar de modo formatado a informação de todos os **clientes** da agência.
5. Gerar e visualizar de modo formatado os **pacotes turísticos** disponíveis (todos, todos relativos a um destino específico, todos entre duas datas, todos os relativos a um destino específico e entre duas datas).

6. Gerar e visualizar de modo formatado os **pacotes turísticos** vendidos (a um cliente específico, a todos os clientes).
7. Efetuar a compra de uma pacote turístico por um cliente.
8. Calcular e visualizar o número e o valor total de pacotes vendidos.
9. Obter o nome dos N locais mais visitados (um pacote pode incluir visitas a vários locais), ordenados por ordem decrescente do número de visitas (= número de pacotes vendidos que passam por esse local).
10. Gerar uma listagem de todos os clientes na qual se indica, para cada cliente, um dos pacotes em que seja visitado um dos N locais mais visitados (ver ponto 9) que ele ainda não visitou.

NOTAS SOBRE O DESENVOLVIMENTO

Funcionamento geral do programa

O funcionamento geral do programa é o seguinte:

- Ler o conteúdo dos ficheiros de agência, clientes e pacotes e guardar em estruturas de dados adequadas.
- Começar por exibir um menu onde aparecem as opções que implementam as funcionalidades principais - gestão de clientes, gestão de pacotes, estatísticas (funcionalidades 9 e 10, descritas anteriormente), terminar – e, realizar a operação selecionada, apresentando submenus, sempre que necessário. Em cada submenu deve existir sempre a opção de voltar ao menu anterior; deve haver sempre a possibilidade de cancelar a execução de uma opção que tenha sido selecionada (exemplo: se o utilizador escolheu inserir cliente, deve poder "voltar atrás" durante a leitura dos dados do cliente, sem inseri-lo).
- No final, gravar (se necessário) a informação da agência, clientes e pacotes.

Devem ser tomadas as precauções necessárias para evitar que o programa deixe de funcionar corretamente devido a entradas inválidas do utilizador, nomeadamente, valores fora da gama admissível.

Escrita do código

Na escrita do código devem ser respeitadas as indicações dadas nas aulas, nomeadamente, em relação aos seguintes aspetos:

- Escolha das estruturas de dados mais adequadas para representar os dados do programa.
- Escolha adequada dos identificadores de tipos, variáveis e funções.
- Estrutura modular do código.
- Escolha adequada das classes.
- Separação em dois ficheiros (.h e .cpp) da definição de cada classe e da implementação dos seus membros função.
- Comentários ao código.

ENTREGA DO TRABALHO

- Criar uma pasta com o nome **TxGyy**, em que **x** representa o número da turma e **yy** representa o número do grupo de trabalho – por exemplo, **T2G05**, para o grupo **5** da turma **2** - e copiar para lá o código-fonte do programa e ficheiros de dados (apenas os ficheiros com a extensão **.cpp** e **.h + agency.txt, clients.txt e packets.txt**). Incluir também um ficheiro **ReadMe.txt** (em formato de texto simples) indicando o estado de desenvolvimento do trabalho, isto é, se foram cumpridos todos os objetivos ou, caso contrário, quais os que não foram cumpridos, ou ainda que melhorias foram implementadas, se for esse o caso. Tudo o que não estiver especificado pode ser decidido pelos elementos do grupo de trabalho, sendo as decisões adoptadas descritas no mesmo ficheiro **ReadMe.txt**.
- Compactar o conteúdo desta pasta num ficheiro **`Gxx.zip`** ou **`Gxx.rar`** e submeter este ficheiro na página da disciplina de Programação, no Moodle da FEUP. Não serão aceites entregas por outras vias.
- Os estudantes deverão informar o docente da turma TP respetiva qual a constituição dos grupos de trabalho, até **29/abr/2019**; a lista dos grupos constituídos será oportunamente publicada no Moodle.
- Data limite para a entrega: **17/maio/2019 (às 23:55h)**.