Projeto Prático – Versão A

Gestão de Estudantes

Introdução

O objetivo deste projeto é desenvolver uma aplicação que faça o registo de estudantes com nome, data de nascimento, nacionalidade, nº matrículas e ECTS concluídos. Além disso, deve permitir fazer listagens dos estudantes (segundos diversos critérios, à escolha do utilizador da aplicação) e o cálculo de variadas medidas estatísticas relativas aos estudantes. Além da correta implementação das funcionalidades exigidas (enumeradas abaixo), valoriza-se a eficiência das aplicações que forem apresentadas, bem como a pertinência de funcionalidades adicionais que sejam propostas.

Dados para o trabalho:

Existem já muitos dados gravados em ficheiros, que devem ser carregados para o programa. Esta informação é fornecida em 2 ficheiros de texto, "*dados.txt*" e "*cod_nMat_ECTS.txt*", que têm o seguinte formato:

estudantes.txt

<cod>\t<nome>\t <dataN>\t <nacionalidade>

```
125    Clotilde Barros Bacelar          08-04-2001    Portuguesa
...
...
8050 Celso Rosado Carvalheira Martins   30-12-2001    Brasileira
8051 Camilo Mendes Pinto Real           16-12-1999    Portuguesa
...
1000 Eduarda Domingos Ferraz            14-07-2005    Guineense
```

situacao_Escolar_Estudantes.txt

<codigo>\t<nMat>\t<ECTS>\tAno_Curso>\t <MediaActual>

```
125     4    140     3     12.5
...
1336    3    170     3     16.2
1337    1     60     2     17.3
...
...
1000    2    110     2     11.3
```

Funcionalidades:

O projeto a implementar terá de implementar as seguintes funcionalidades básicas:

1: Na fase de "instalação" da aplicação, ativável através de um qualquer mecanismo que considerarem mais adequado, onde:

   a) Deverão ser especificados, na fase de instalação, o número máximo de registos que será possível guardar em cada um dos dados estruturados que vão guardar em memoria a informação que a aplicação vai tratar, exceto se os dados estruturados forem tratados dinamicamente. Neste caso, a existência de um máximo de registos a tratar e se há um número inicial de registos e depois, um valor para cada incremento, quando o número de registos disponíveis no momento for atingido, é uma opção do grupo. Poderá haver ainda o outra informação de inicialização a especificar, se a arquitectura do sistema concebida o exigir. Toda esta informação recolhida deverá, obviamente, ser guardada em memória e assegurada a sua persistência (em ficheiro(s)), da forma que entenderem;

b) Serão lidos os ficheiros de dados fornecidos, carregando-os para estruturas de dados convenientes, não esquecendo de assegurar a respectiva persistência dos dados existentes em memória, (p. ex., executando a opção 11 da aplicação);

c) Aquando da utilização "normal", ao arrancar a aplicação, ler os ficheiros de dados gravados na utilização anterior da aplicação (ou na instalação), carregando-os para dados simples ou estruturados, utilizando estruturas de dados convenientes já usadas na alínea anterior, não esquecendo também de ler eventuais outros ficheiros que tenham sido acrescentados para guardar outros dados importantes para o funcionamento da aplicação (parâmetros do sistema, tais como: número máximo de registos, numeração de documentos e outros);

2: Gerir estudantes: inserir, eliminar, atualizar (obs.: a) ao inserir ou eliminar um estudante, deve-se, igualmente, tratar os dados correspondentes à sua situação escolar; b) quando for inserido um novo estudante, o código respetivo deve ser sequencial e gerado automaticamente);

3: Apresentar a informação relacionada com o(s) estudante(s), cuja pesquisa é feita, especificando parte do nome;

4: Listar os estudantes, cujas datas de nascimento estejam dentro de um intervalo de datas especificadas (data inf. até data sup.) e pertencentes a um conjunto máximo de 5 nacionalidades;

5: Calcular e mostrar quantos estudantes existem em cada um de 6 escalões etários (idade entre x e y anos), especificados pelo utilizador;

6: Determinar o número médio de matrículas dos estudantes (geral e por nacionalidade);

7: Listar estudantes por ordem alfabética do apelido (considera-se apelido a última palavra do nome completo);

8: Determinar o número de estudantes finalistas (pelo menos 154 ECTS realizados);

9: Determinar a média de idades dos estudantes de uma nacionalidade, atendendo ao ano que estão a frequentar;

10: Determinar o número de estudantes em risco de prescrever (3 matrículas e menos de 60 ECTS, 4 matrículas e menos de 120 ECTS ou mais de 5 matrículas e não finalista) e proceder à sua listagem;

11: Assegurar a persistência dos dados existentes em memória, utilizando para o efeito ficheiros binários, sempre que o utilizador assim o entender e, obrigatoriamente, ao sair da aplicação, não devendo esquecer-se a atualização de outros ficheiros que tenham sido igualmente criados e utilizados para guardar outra informação importante e que terá sido carregada em memória, aquando do arranque da aplicação (neste caso, poderão ser guardados em fich. tipo txt), pois os dados que contêm poderão ser conhecidos com facilidade e, eventualmente, alterados, se for necessário.

Nas diversas listagens a implementar, dada a potencial elevada quantidade de informação, deve implementar um mecanismo de avanço página a página e deve ser oferecida a possibilidade de gravar a listagem em ficheiro de texto .txt e .csv, para posterior consulta num simples editor de texto e no Excel o aplicativo similar.

Extras:

E1: Listar os estudantes que nasceram em cada dia da semana ou cujo aniversário, num determinado ano, é ao Domingo;

E2: Listar estudantes cujo aniversário, num determinado ano, é na Quaresma (período temporal entre o Carnaval e a Páscoa).

Alternativa - Gerar uma consulta do tipo apresentado infra:

Número de estudantes por Intervalos de Classificação Média e Ano Frequentado

Filtros:	Intervalos de Média Actual	Todos
	Anos de Inscrição	1.º a 3.º

.

	Ano de Inscrição			
Intervalo de Média Actual	**1.º**	**2.º**	**3.º**	**Todos os anos**
Sem Classificação	245	0	0	245
[10 .. 12[	144	234	123	501
[12 .. 14[	133	245	321	699
[14 .. 16[	94	121	211	426
[16 ..18[	23	24	32	79
[18 .. 20]	14	22	24	60
Todos os Intervalos	653	646	711	

Desenvolvimento Faseado

O desenvolvimento deste projeto deverá ser feito de uma forma faseada, devendo os alunos garantir que todas as funcionalidades codificadas até ao momento estão a funcionar corretamente.

"É preferível apresentar um programa que implementa poucas funcionalidades, mas que funcionam corretamente, do que um programa totalmente desenvolvido, mas que não faz nada."

Os estudantes deverão garantir a robustez da aplicação, verificando todos os casos de erro (por exemplo, ao nível dos ficheiros).

Documentação

O código produzido pelos estudantes deverá ser comentado. Os comentários presentes no código devem explicitar e explicar o funcionamento da aplicação, assim como as decisões tomadas. Devem seguir as seguintes regras:

- A função deve ser precedida por um comentário que explique a funcionalidade implementada e o significado, se for caso disso, de cada um dos parâmetros e do valor devolvido;
- O código deve ser comentado sempre que realize alguma operação não óbvia;
- Os comentários devem ser claros, gramaticalmente corretos e usando frases simples;
- A declaração de todas as variáveis e constantes devem ser acompanhadas de um comentário com uma breve descrição;
- Cada bloco de código (seleção ou repetição) deve ser precedido de um breve comentário explicativo.

Como regra geral deve considerar que cada função deve cumprir as seguintes regras:

- Inicializar sempre as variáveis na sua declaração;
- Testar a validade dos parâmetros recebidos;
- Declarar constantes e nunca usar números no código;
- Evitar repetições de código - usar funções, ciclos, etc.;
- Evitar o uso de variáveis globais;
- Escrever código simples e claro.

Observações

- Os grupos poderão integrar um máximo de 4 alunos;
- Os estudantes/grupos podem conversar entre si para discutir possíveis soluções para algum problema que tenham, mas não devem partilhar código fonte, sob pena de ambos os trabalhos serem penalizados na classificação atribuída, se o "plágio" for apenas pontual, mas, relativamente a este assunto, atentem ao descrito no penúltimo item desta lista;
- O código pode ser dividido em módulos, para que possa ser desenvolvido em paralelo (havendo uma partilha, da responsabilidade do grupo ou, preferencialmente, de um "gestor de projecto" que terá necessariamente, vários outros papeis, da maior importância). De qualquer modo, todos os elementos do grupo devem conhecer todo o trabalho desenvolvido, como se cada um tivesse realizado todas as fases que permitiram criar a aplicação e todos os demais componentes complementares);
- A aplicação pode ser estruturada sob a forma de um Projecto, devendo ser convenientemente distribuída por diversos ficheiros, cada um com um conteúdo com uma dada caracterização. P. ex. um fich. "main.c"; outro "func.c" com as funções, ou vários ficheiros de nomes expressivos com as funções divididas por grupos de funcionalidades (com algo de comum); ficheiros tipo

"header" (xxxx.h) com structs ou outros elementos integrantes do programa que possam ser incluídos em vários ficheiros do projecto, etc. Esta estruturação, se bem realizada, será, naturalmente, valorizada;

- Deverão ser criadas as estruturas de dados necessárias;
- Deverão usar os ficheiros que forem fornecidos para carregar em memória os dados estruturados criados, cujos tipos devem ser os apropriados, possivelmente do tipo de cada uma das estruturas de dados criadas. Não esquecer que, quando especificado, a importação dos dados dos ficheiros, pode implicar algum processamento adicional (p. ex., o cálculo de um ou mais valores ou a atualização do estado do sistema (dada a necessária atualização de outros dados estruturados em memória);
- Todo o processamento de dados, durante a utilização da aplicação (inserção de novos registos, alterações, consultas, e outras operações) deve ser feito utilizando os dados estruturados existentes em memória;
- A validação de dados será valorizada, quer na operação de leitura dos ficheiros para memória, quer relativamente aos dados especificados pelo utilizador;
- O trabalho deve ser implementado em linguagem C standard e deve apenas usar os conhecimentos lecionados nas aulas de Algoritmos e Programação;
- A funcionalidade do programa é, nesta unidade curricular, muito mais importante que os aspetos estéticos. Será valorizada a clareza e simplicidade do código. Deverá assim usar funções e estruturar o programa de modo a torná-lo simples, bem estruturado, e sem repetições desnecessárias de código. Será ainda valorizada a conveniente indentação do código e a inclusão de comentários, sempre que tal for interessante do ponto de vista da legibilidade (aliás, este tema foi já realçado na secção "Documentação";
- Possíveis melhorias incluídas no programa, devidamente fundamentadas e enquadradas com o programa desta unidade curricular, serão consideradas e valorizadas;
- Para os dados estruturados e sua manipulação, a) podem utilizar estruturas de tamanho fixo (ex. vectores declarados e criados, usando o operador [] e, sendo, depois, os elementos acedidos igualmente, através do mesmo operador); b) podem utilizar memória dinâmica e, depois, ponteiros e aritmética de ponteiros, para aceder ao cada elemento. Como a utilização da opção b) é, do ponto de vista da utilização de recursos mais eficiente, naturalmente, a aplicação que a utilizar será também alvo de uma valorização superior.

  Obs. Quanto à opção a seleccionar para esta característica da aplicação, é importante não esquecer a "*quote*" incluída na seção "Desenvolvimento Faseado" e que se passa a transcrever: "*É preferível apresentar um programa que implementa poucas funcionalidades, mas que funcionam corretamente, do que um programa totalmente desenvolvido, mas que não faz nada.*".

  Neste contexto, esta frase poderia ser adaptada para algo como "*É preferível apresentar um programa onde a declaração e manipulação de dados estruturados utilize o operador [], mas que funcionam corretamente, do que um programa totalmente desenvolvido utilizando ponteiros e aritmética de ponteiros para cesso aos registos, mas que não funciona.*".

Ainda relativamente a esta opção de desenvolvimento, se não se sentirem "à vontade" com o uso de ponteiros, podem, numa 1.ª fase do desenvolvimento e teste da aplicação, usar o operador []. Depois de terem todas (ou aquelas que conseguirem desenvolver) e convenientemente testadas, e todas as restantes tarefas correspondentes a todos os elementos a entregar que devem integrar o MiniProjecto já realizadas e com a devida qualidade e, gerado o respectivo pdf pronto a submeter, então, se ainda tiverem tempo disponível (significando isto não irão prejudicar o estudo para outras UC a que ainda não obtiveram aprovação ou decidiram, para melhor se poderem preparar, submeter-se a avaliação em outra prova ou época), podem então converter a aplicação que têm pronta, adotando a solução a) para uma outra, onde tentarão usar a solução b). Não esquecer, que, a cada transformação realizada, devem testar o que vai sendo desenvolvido e, corrigindo-o, se for o caso. No final, se a nova versão estiver também com as funcionalidades implementadas iguais à anterior, podem, então preparar o novo fich. .zip com a nova versão, não esquecendo de providenciar à alteração do relatório, para incluir a

informação de a aplicação ter sido implementada usando memória dinâmica e efectuar outras alterações, se algum código apresentado ou outro texto tenha de ser agora adaptado, dada a adopção da opção

- Deve ser elaborado um relatório que descreva o trabalho produzido e as funcionalidades implementadas, adotando o modelo e diretrizes publicadas no Moodle;
- Não serão admitidos plágios, mesmo que parciais - trabalhos copiados terão nota ZERO (tanto quem copiou como quem deixou copiar);
- Caso se revele necessário, poderão ser feitas atualizações ou alterações a este enunciado, pelo que os estudantes deverão estar atentos a esta eventualidade. Qualquer atualização ou alteração será devidamente anunciada no Moodle desta unidade curricular.

Entrega

O trabalho a entregar deve consistir num único ficheiro comprimido com o formato ZIP, devendo ser entregue até data a anunciar no link de submissão, tendo em atenção as seguintes indicações gerais:

1. O relatório solicitado deve estar em formato pdf;
2. A submissão deve ser realizada por um dos elementos do grupo, no link apropriado existente na página da unidade curricular no Moodle. Tal como estará indicado no texto que está junto ao link de submissão, nome do ficheiro deve ter o seguinte formato: EN_NomeApelido1_NomeApelido2_NomeApelido3_NomeApelido4 (com ordenação alfabética crescente) e deve conter todos os ficheiros: o código fonte da aplicação, bem como os ficheiros de dados necessários à execução, o relatório e um fich. txt com a identificação dos estudantes (nome e número). Um exemplo para o nome do ficheiro poderá ser: EN_AntonioCoelho_JoaquimFerreia_ManuelTomas.zip,
3. Em último recurso, o ficheiro com o trabalho pode ser enviado por e-mail, diretamente para jfialho@estgv.ipv.pt, caso por uma infeliz eventualidade, o Moodle não esteja a operacional.

Apresentação e Defesa

A defesa dos trabalhos, com a presença de **TODOS** os elementos do grupo de trabalho, é **obrigatória** em data, hora e local a indicar oportunamente.