Paradigmas da Programação II
LESI + LMCC (1^o ano)

Trabalho Prático

Ano Lectivo de 00/01

> Este trabalho único deve ser realizado em grupo ao longo do semestre, e deve ser entregue em 4 fases, nas datas marcadas abaixo.

1 Objectivos e Organização

Este trabalho prático tem como principal **objectivo** a consolidação dos conhecimentos adquiridos:

- na análise e especificação de problemas;
- na escolha das estruturas de dados e respectivos algoritmos para resolver os problemas especificados;
- na codificação, na linguagem imperativa C, das estruturas de dados e dos algoritmos concebidos;
- na execução e teste e dos programas C escritos;
- na documentação dos projectos de programação realizados;
- na utilização do ambiente linux para programação, nomeadamente no que respeita ao uso do make e do RCS.

O **relatório** a elaborar, deve ser claro e sucinto e, além do respectivo enunciado e da descrição de cada uma das fases executadas, deverá conter exemplos de utilização e o código do programa. Como foi dito, o relatório terá de ser escrito em LaTeX sugerindo-se que adoptem o esqueleto proposto que pode ser descarregado a partir da página WWW principal da disciplina, no URL `http://www.di.uminho.pt/~prh/curpp200.html`.

Para o efeito, esta folha contém vários enunciados, dos quais deverá resolver **pelo menos um** (para efeitos de avaliação prática da disciplina só um trabalho conta). Cada enunciado cria um enredo completo e contem 4 partes, pelo que o trabalho deve ser desenvolvido e apresentado em 4 fases segundo o calendário:

1ª fase: semana de 26 de março

2ª fase: semana de 23 de abril

3ª fase: semana de 14 de maio

4ª fase: semana de 28 de maio (primeira semana após fim das aulas)

2 Enunciados

2.1 Selecção de Projectos de I&D

Considere a seguinte situação:
Pretende-se criar, com a ajuda do computador, um sistema de informação (SI) para apoio à selecção de projectos de investigação candidatos ao financiamento disponibilizado por uma Instituição Nacional (suponha a FCT).
Cada projecto candidato é caracterizado pelos seguintes atributos:

- titulo
- instituição proponente
- nome do investigador responsável
- duração do projecto (número de anos)
- lista de palavras-chave (até 5), designando as grandes áreas/tópicos de investigação
- orçamento: gastos previstos (em contos) distribuídos pelas seguintes rúbricas:
  - compra equipamento
  - pagamento tarefeiros
  - despesas com consumíveis
  - despesas com consultores
  - deslocações nacionais ou estrangeiras
  - gastos gerais
- classificação atribuída pelos 4 avaliadores; correspondente a cada um, será fornecida uma sequência de 5 letras[1] representando a nota dada a cada um dos 5 critérios de apreciação das propostas: importância do projecto, originalidade, qualidade da proposta, idoneidade da equipe, exequibilidade.

2.1.1 divulgação

Usando um browser da WWW, construa uma apresentação (conjunto de diapositivos) em HTML que possa servir para a FCT **divulgar** a *abertura desse concurso para financiamento de projectos*, salientando a que áreas científicas se destina, quem pode concorrer, como concorrer, prazos, quais os critérios de selecção, etc.
Sugestão: pode criar os seus diapositivos directamente em HTML, a partir do zero, ou então usar um gerador qualquer de apresentações que conheça, como por exemplo o programa `mktalk` que pode descarregar a partir do URL `http://natura.di.uminho.pt` (escolha na secção `tool` a opção `gerador de slides`).

2.1.2 aquisição dos dados

Na primeira fase do trabalho, além dos diapositivos para apoio à divulgação (como pedido acima), pretende-se que desenvolva um programa que lhe permita ler os dados referentes a vários projectos, atribuir a cada um número interno de código (pode ser sequencial) e calcular o custo total do projecto e a classificação final (média dos 4 avaliadores sendo, para cada um, calculada a média das notas atribuidas aos 5 parâmetros, considerando N=1, M=2, S=3, B=4, E=5).
O programa deve escrever num ficheiro de texto (de resultados) o código interno, título, orçamento

[1] cada letra pode tomar um dos valores N, M, S, B, E, significando: Não satisfaz; Minimamente satisfatório; Satisfaz; Bom; Excelente.

total solicitado e classificação final, por ordem decrescente de classificações.
Finalmente o programa deve determinar o subsídio a atribuir pela FCT a cada projecto, de acordo com o seguinte critério:

1. se a classificação for inferior a 3 pontos, o projecto será *rejeitado*, isto é, não lhe e atribuido qualquer financiamento.
2. é 30% do solicitado, se a classificação for 3 pontos;
3. é 75% do solicitado, se a classificação for 4 pontos;
4. é 100% do solicitado, se a classificação for 5 pontos.

2.1.3 armazenamento e pesquisa de informação

Na segunda fase do trabalho, pretende-se que guarde em memória os dados lidos referentes a cada um dos projectos, associados ao código que foi automaticamente atribuido, à classificação calculada e ao financiamento atribuido.
Para o armazenamento deve usar uma lista ligada, recorrendo a memória alocada dinamicamente.
O programa deve, depois, facultar as vulgares operações de acesso à informação, a saber:

- **listagem global** de todos os projectos, ordenada por Instituição Proponente e, dentro desta, por nome do Investigador Responsável;
- **procura de um projecto** dado o seu Código, ou o Título;
- **procura de todos os projectos** pertencentes a um dada Área/Tópico de investigação (de acordo com a classificação em palavras-chave fornecida).

2.1.4 armazenamento em ficheiro sequencial

Na terceira fase do trabalho, pretende-se que acrescente ao menu principal do seu programa duas novas opções: uma que lhe permita guardar em memória secundária (ficheiro sequencial) todos os dados lidos até ao momento e que estão armazenados na lista ligada criada na fase anterior; outra, que permita recuperar de um ficheiro sequencial todos os dados armazenados de modo a recarregá-los para uma lista ligada em memória.
Em ambos os casos, o utilizador deve indicar o nome do ficheiro em que quer escrever ou ler.
Deve ser também incluída a possibilidade de especificar na linha de comando de invocação, a seguir ao nome do seu programa, o *switch* `-f` seguido do nome dum ficheiro. Nesse caso, o programa deve carregar, no arranque, automaticamente os dados a partir do ficheiro mencionado (equivalente à segunda opção pedida acima).

2.1.5 geração de relatórios HTML

Na quarta fase do trabalho, pretende-se que gere um conjunto de 4 páginas em HTML correspondentes às 4 listas de projectos rejeitados, financiados a 30%, a 75% e a 100%, indicando para cada projecto o *título*, *instituição* e *investigador responsável*. Deverá, ainda, ser produzida uma página extra que servirá de índice de acesso às 4 páginas pedidas.
Essas páginas devem ser criadas a partir dos dados armazenados em ficheiro sequencial (construído conforme solicitado na fase anterior).
Esta tarefa deverá ser executada em modo *batch* (não interactivo), pelo que será activada automaticamente quando na linha de comando de invocação, a seguir ao nome do seu programa, aparecer o *switch* `-h` seguido do nome do ficheiro.
Para tornar o sistema mais poderoso, pretende-se, também, que o programa aceite (ao ser invocado) o *switch* `-m` seguido do nome de dois ficheiros. Nessa situação, deve **fundir** os dois ficheiros dados para um terceiro ficheiro (de nome fixo, a ser escolhido pelo grupo), procedendo, como acima, em modo *batch* (isto é, autonomamente, sem intervenção do utilizador).

2.2 Gestão de Obras numa Editora

Considere a seguinte situação:
Pretende-se criar, com a ajuda do computador, um sistema de informação (SI) para gerir uma editora livreira, essencialmente na perspectiva da sua relação de trabalho com os autores.
O ciclo de vida de uma obra é constituido pelas seguintes fases: entrega do original (pelo autor); apreciação da obra (pelo editor); rejeição (nesse caso o processo termina aqui), ou celebração do contrato; edição (pelo revisor); revisão do texto (pelo autor); impressão.
Cada original é descrito pelos seguintes atributos:

- titulo
- nome do autor, ou autores
- número de páginas
- número de fotografias
- tipo (um entre uma lista limitada de hipóteses: romance, biografia, ficção, policial, aventura, etc.)
- ano em que foi escrito
- data de entrega
- nome do receptor (pessoa da editora que aceitou a obra para análise)
- avaliação preliminar atribuida pelo receptor; uma sequência de 3 letras (cada letra pode ser A, B, ou C) representando a primeira opinião relativa aos seguintes critérios de decisão: reconhecimento do autor; oportunidade comercial do tipo de obra; facilidade de edição/publicação.

2.2.1 divulgação

Para começar o seu trabalho, constrúa uma apresentação (conjunto de diapositivos) em HTML que, usando um browser da WWW, possa servir para a editora em causa anunciar, quer os serviços que pode prestar aos autores, quer as principais colecções que publica, quer ainda as obras editadas mais recentemente.
Sugestão: pode criar os seus diapositivos directamente em HTML, a partir do zero, ou então usar um gerador qualquer de apresentações que conheça, como por exemplo o programa `mktalk` que pode descarregar a partir do URL `http://natura.di.uminho.pt` (escolha na secção `tool` a opção `gerador de slides`).

2.2.2 aquisição dos dados

Na primeira fase do trabalho, além do sistema de divulgação pedido acima, pretende-se que desenvolva um programa que lhe permita ler os dados referentes aos originais recepcionados ao longo de um dia.
Depois de atribuir um número interno de código (pode ser sequencial) a cada original, deve ser calculada a sua classificação preliminar fazendo a média das notas atribuidas aos 3 parâmetros acima indicados (considere A=20, B=10, C=5).
O titulo da cada original recepcionado, juntamente com o código interno, a classificação calculada e a data provável para comunicação da decisão dos editores[2] deve ser escrito num ficheiro de texto (de resultados), agrupando as obras por tipos.

[2] considere que, em média, se demora 1 semana por cada 100 páginas (faça sempre o arredondamento à semana, por excesso).

2.2.3 armazenamento e pesquisa de informação

Na segunda fase do trabalho, pretende-se que guarde em memória os dados lidos referentes a cada um dos originais, associados ao código que foi automaticamente atribuido, à classificação calculada e à data provável de decisão.
Para o armazenamento deve usar uma lista ligada, recorrendo a memória alocada dinamicamente.
O programa deve, depois, facultar as vulgares operações de acesso à informação, a saber:

- **listagem global** de todos os originais registados, ordenada por Data de Registo e, dentro desta, por Nome do primeiro Autor;
- **procura de um original** dado o seu Código, ou o Título;
- **procura de todos os projectos** pertencentes a um dado Tipo.

2.2.4 armazenamento em ficheiro sequencial

Na terceira fase do trabalho, pretende-se que acrescente ao menu principal do seu programa duas novas opções: uma que lhe permita guardar em memória secundária (ficheiro sequencial) todos os dados lidos até ao momento e que estão armazenados na lista ligada criada na fase anterior; outra, que permita recuperar de um ficheiro sequencial todos os dados armazenados de modo a recarregá-los para uma lista ligada em memória.
Em ambos os casos, o utilizador deve indicar o nome do ficheiro em que quer escrever ou ler.
Deve ser também incluída a possibilidade de especificar na linha de comando de invocação, a seguir ao nome do seu programa, o *switch* `-f` seguido do nome dum ficheiro. Nesse caso, o programa deve carregar, no arranque, automaticamente os dados a partir do ficheiro mencionado (equivalente à segunda opção pedida acima).

2.2.5 geração de relatórios HTML

Na quarta fase do trabalho, pretende-se que gere um conjunto de 2 páginas em HTML correspondentes às 2 listas de livros liminarmente rejeitados (classificação média menor do que 10 pontos), e aceites para apreciação (classificados com média maior ou igual a 10 pontos), indicando para cada livro o *título*, *nome dos autores*. As páginas apenas devem incluir livros co data de recepção menor ou igual a uma data indicada. Deverá, ainda, ser produzida uma página extra que servirá de índice de acesso às 4 páginas pedidas.
Essas páginas devem ser criadas a partir dos dados armazenados em ficheiro sequencial (construído conforme solicitado na fase anterior).
Esta tarefa deverá ser executada em modo *batch* (não interactivo), pelo que será activada automaticamente quando na linha de comando de invocação, a seguir ao nome do seu programa, aparecer o *switch* `-h` seguido do nome do ficheiro e da data de referência limite.
Para tornar o sistema mais poderoso, pretende-se, também, que o programa aceite (ao ser invocado) o *switch* `-m` seguido do nome de dois ficheiros. Nessa situação, deve **fundir** os dois ficheiros dados para um terceiro ficheiro (de nome fixo, a ser escolhido pelo grupo), procedendo, como acima, em modo *batch* (isto é, autonomamente, sem intervenção do utilizador).

2.3 Arquivo de letras de música com acordes de viola

Pretende-se manter um arquivo de letras de música em formato chord com colaboração distribuída.

Para tal é necessário guardar uma lista dos acordes conhecidos, e permitir ir recebendo novas músicas.

Cada música, ao ser recebida, deve levar um número identificador

Um música vai ser definida pelos seguintes atributos:

- título

- nome dos autores
- nome do cantor ou agrupamento
- data
- nome e email de quem enviou a música
- nome do ficheiro que contem a letra da música em causa com os respectivos acordes

A Notação para inserir acordes pelo meio da letra das músicas consiste em escrever a sigla que designa o acorde pretendido entre parentesis rectos, como se exemplifica a seguir:

```
...
[Am]They made a [Am/G]record and
it [Am/F#]went in the [Am/G]charts,
The [G]sky [F]was the [C]limit.
[G] [C] [G] [G] [C] [G]
...
```

A Notação para definição de acordes (não necessária na fase 1) é a seguinte:

```
{define: mi 0 0 1 2 2 0}
{define: la 0 2 2 2 0 x}
{define: lam 0 1 2 2 0 x}
```

2.3.1 divulgação

Usando um browser da WWW, construa uma apresentação (conjunto de diapositivos) em HTML que possa servir para apresentar e divulgar o projecto (indicando objectivos, como participar, etc, não exitando em inventar um pouco: enunciado está ainda muito incompleto...).
Sugestão: pode criar os seus diapositivos directamente em HTML, a partir do zero, ou então usar um gerador qualquer de apresentações que conheça, como por exemplo o programa `mktalk` que pode descarregar a partir do URL `http://natura.di.uminho.pt` (escolha na secção `tool` a opção `gerador de slides`).

2.3.2 aquisição de dados

Na primeira fase, pretende-se que

- desenvolva um programa para fazer a aquisição dos dados referentes a uma ou mais músicas, devendo indicar-se no fim quantas letras de cada autor foram lidas e qual foi o cantor/agrupamento que apareceu em mais músicas;
- acrescente a esse programa uma função que, dado o ficheiro que contém uma música (poema com acordes):
  1. Construa a lista dos acordes usados (sem repetições);
  2. Crie um ficheiro de texto só com o poema.

2.3.3 armazenamento e pesquisa de informação

Na segunda fase do trabalho, pretende-se que guarde em memória os dados lidos referentes a cada uma das músicas, associados ao código que foi automaticamente atribuido.
Para o armazenamento deve usar uma lista ligada, recorrendo a memória alocada dinamicamente.
O programa deve, depois, facultar as vulgares operações de acesso à informação, a saber:

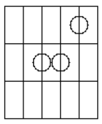
- **listagem global** de todos as músicas, ordenada por Cantor/Agrupamento e, dentro desta, por Título;
- **procura de uma música** dado o seu Código, ou um dos Autores;
- **procura de todas as músicas** criadas numa determinada Data.

2.3.4 armazenamento em ficheiro sequencial

Na terceira fase do trabalho, pretende-se que acrescente ao menu principal do seu programa duas novas opções: uma que lhe permita guardar em memória secundária (ficheiro sequencial) todos os dados lidos até ao momento e que estão armazenados na lista ligada criada na fase anterior; outra, que permita recuperar de um ficheiro sequencial todos os dados armazenados de modo a recarregá-los para uma lista ligada em memória.
Em ambos os casos, o utilizador deve indicar o nome do ficheiro em que quer escrever ou ler.
Deve ser também incluída a possibilidade de especificar na linha de comando de invocação, a seguir ao nome do seu programa, o *switch* `-f` seguido do nome dum ficheiro. Nesse caso, o programa deve carregar, no arranque, automaticamente os dados a partir do ficheiro mencionado (equivalente à segunda opção pedida acima).

2.3.5 Geração de páginas HTML

Na quarta fase do trabalho pretende-se que gere páginas HTML para um conjunto de músicas seleccionadas de acordo com as pesquisas anteriores.
As páginas geradas deverão incluir a figura dos acordes contidos na música respectiva.
Por exemplo o acorde *la menor* definico como `{define: lam 0 1 2 2 0 x}` poderia corresponder à seguinte imagem:



Sugere-se para tal que use a biblioteca gd (experimente `locate libgd` para ver se tem essa biblioteca instalada na sua máquina). Veja o seguinte exemplo de uso da biblioteca gd:

```
#include "gd.h"
#include <stdio.h>

main(){

  gdImagePtr im;
  int white,black;
  FILE * out;

  im = gdImageCreate(140,140);
  white=gdImageColorAllocate(im,255,255,255);
  black=gdImageColorAllocate(im,0,0,0);
  gdImageLine(im,20,0,20,140,black);
  gdImageLine(im,40,0,40,140,black);
  gdImageArc(im,20,20,20,20,0,360,black);

  out=fopen("_.png","w");
  gdImagePng(im,out);
}
```

coloca em _.png a seguinte imagem:

2.4 Arquivo musical em formato ABC

Pretende-se manter um arquivo de partituras na internet com colaboração distruibuída.
Esse arquivo guarda partituras em formato ABC (um formato completamente textual).
Cada música, ao ser recebida, deve levar um número identificador
Um música vai ser definida pelos seguintes atributos:

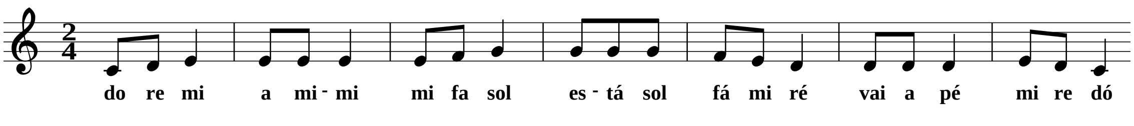
- título
- nome dos autores
- nome do cantor ou agrupamento
- data
- nome e email de quem enviou a música
- nome do ficheiro que contem a partitura em formato ABC

Exemplo da notação ABC aplicada à conhecida música "Do re mi a Mimi..."

```
X: 1
K: C
M: 2/4|
L: 1/8
CDE2 | EEE2 | EFG2 | GGG | FED2 | DDD2 | EDC2
w: do re mi a mi-mi mi fa sol es-tá sol fá mi ré vai a pé mi re dó

Explicação:
  M: 2/4|               - compasso (neste caso binário)
  L: 1/8                - tempo de base de cada nota

  C=dó,D=ré,E=Mi, etc - notas com a duração do tempo de base
  G2                    - Sol com o dobro da duração do tempo de base
  |                     - barra de separação do compasso
  w: ...                - letra correspondente (separações por ' ' ou '-')
```



2.4.1 divulgação

Usando um browser da WWW, construa uma apresentação (conjunto de diapositivos) em HTML que possa servir para apresentar e divulgar o projecto (indicando objectivos, como participar, etc, não exitando em inventar um pouco: enunciado está ainda muito incompleto...).
Sugestão: pode criar os seus diapositivos directamente em HTML, a partir do zero, ou então usar um gerador qualquer de apresentações que conheça, como por exemplo o programa `mktalk` que pode descarregar a partir do URL `http://natura.di.uminho.pt` (escolha na secção `tool` a opção `gerador de slides`).

2.4.2 aquisição de dados

Na primeira fase, pretende-se que

- desenvolva um programa para fazer a aquisição dos dados referentes a uma ou mais músicas, devendo indicar-se no fim quantas letras de cada cantor/agrupamento foram lidas e quem foi o contribuinte que enviou mais músicas;
- acrescente a esse programa uma função que, dado o ficheiro que contém uma partitura em formato ABC:
  1. verifique se há tantas sílabas quantas as notas.
  2. supondo que a partitura só trás a primeira barra de compasso, acrescente as outras barras (todos os compasso devem ter a mesma duração).

2.4.3 armazenamento e pesquisa de informação

Na segunda fase do trabalho, pretende-se que guarde em memória os dados lidos referentes a cada uma das músicas, associados ao código que foi automaticamente atribuido.
Para o armazenamento deve usar uma lista ligada, recorrendo a memória alocada dinamicamente.
O programa deve, depois, facultar as vulgares operações de acesso à informação, a saber:

- **listagem global** de todos as músicas, ordenada por Cantor/Agrupamento e, dentro desta, por Data;
- **procura de uma música** dado o seu Código, ou um dos Autores;
- **procura de todas as músicas** enviadas pela mesma Pessoa.

2.4.4 armazenamento em ficheiro sequencial

Na terceira fase do trabalho, pretende-se que acrescente ao menu principal do seu programa duas novas opções: uma que lhe permita guardar em memória secundária (ficheiro sequencial) todos os dados lidos até ao momento e que estão armazenados na lista ligada criada na fase anterior; outra, que permita recuperar de um ficheiro sequencial todos os dados armazenados de modo a recarregá-los para uma lista ligada em memória.
Em ambos os casos, o utilizador deve indicar o nome do ficheiro em que quer escrever ou ler.
Deve ser também incluída a possibilidade de especificar na linha de comando de invocação, a seguir ao nome do seu programa, o *switch* `-f` seguido do nome dum ficheiro. Nesse caso, o programa deve carregar, no arranque, automaticamente os dados a partir do ficheiro mencionado (equivalente à segunda opção pedida acima).

2.4.5 Geração de cancioneiros

Na quarta fase pretende-se gerar cancioneiros com as partituras em LaTeX e HTML.

Para gerar a imagem com a partitura comece por converter o ficheiro abc em Postscrit(ps) (usando o programa de domínio público `abc2ps` que deverá procurar na internet e instalar na sua máquina).

Seguidamente converta o ficheiro ps para Encapsulated Postscript (eps) usando o programa `ps2epsi` normalmente disponível em unix/linux. Os ficheiros eps são directamente incluíveis em LaTeX (ver exemplo de uso no texto latex original deste enunciado, disponível na página da cadeira, nomeadamente o uso de `usepackage{epsf}` e a inclusão das imagens).

Para definir quais as músicas a incluir e porque ordem, a função pedida nesta fase, aceita um ficheiro definidor que inclui linhas contendo:

- código de música ou

- nome de cantor/agrupamento (pretendendo-se incluir todas as músicas desse cantor/agrupamento)

Deverá escolher e documentar a sintaxe que deve seguir esse ficheiro.
Use o programa latex2html para geração de HTML.