Programação Imperativa
LMCC (1º ano)

Exame de Recurso

Data: 26 de Julho de 2005
Hora: 9:30

Dispõe de **2:00 horas** para realizar este exame.
Responda na folha fornecida em anexo e não se esqueça de preencher o cabeçalho

1 Questão (cálculo numérico)

Especifique o algoritmo para redução de uma fracção à sua forma mais simples, i. e., o numerador é o mais pequeno possível. A função especificada deverá receber uma fracção como argumento e dar como resultado uma nova fracção. Posteriormente codifique em C a sua solução.

%---

2 Questão (ficheiros)

O programa `mente` é um tradicional Filtro de Texto que separa as palavras terminadas em "`mente`".
A função principal interactua com o utilizador para o que for necessário e chama uma função auxiliar `mente0()` para proceder à leitura e filtragem do texto.

Escreva, então, em linguagem C, a função `mente0()`, que copia para uma lista, implementada como *lista ligada dinâmica*, inserindo ordenadamente, todas as palavras do texto de entrada, lido directamente do teclado, terminadas com o sufixo "`mente`". No fim deve escrever num ficheiro de texto de nome `"resultados.txt"` a dita lista, indicando no fim o número de vezes que a palavra surgiu. Note que, assim sendo, não deve repetir palavras.

%---

3 Questão (recursividade)

Relembre as primeiras aulas onde foram discutidos vários algoritmos de conversão de dados. A conversão de inteiro para binário pode ser feita pelo método de divisões sucessivas. Esta função foi amplamente discutida e implementada nas aulas.

Desenvolva duas versões em C da função, uma recursiva e outra iterativa. A assinatura da função deverá ser a mesma e seguir a seguinte forma:

```
void f(int num)
{
 ...
```

```
}
```

O tipo da função é `void` porque se pretende que a função vá imprimindo os bits do número binário (o bit mais significativo deverá ser impresso primeiro de modo a que o número apareça correctamente).

%———————————————————————————

4 Questão (árvore binárias de procura)

Considere as seguintes definições para uma árvore binária de procura de números inteiros e para uma lista ligada de inteiros respectivamente:

```
/* Árvore Binária de Procura */
typedef struct sABPInt
{
  int valor;
  struct sABPInt *esq, *dir;
} NODO, *ABPint;

/* Lista Ligada */
typedef struct sLLInt
{
  int valor;
  struct sLLInt *seg;
} ELEM, Lista;
```

a) Desenvolva a função em C que converte a árvore numa lista atravessando a árvore segundo o algoritmo de travessia `inorder`:

```
Lista convAL( ABPint a)
{
  ...
}
```

b) Especifique a função, em C, que retorna o elemento mais pequeno armazenado numa árvore binária de procura.

c) Especifique a função, em C, que recebendo duas lista ligadas de inteiros ordenadas dá como resulta uma lista ligada de inteiros ordenada resultante da fusão das duas listas passadas como argumento.

%———————————————————————————

5 Questão (estrutura de dados)

Considere uma árvore genealógica ascendente (que relaciona uma pessoa com os seus pais) e que em termos mais formais pode ser descrita da seguinte maneira:

```
AG  = nome * Pai * mãe | desconhecido
Pai = Mãe = AG
```

1. Especifique em C o tipo de dados AG;
2. Especifique a função `bisavós`, que dada uma árvore genealógica de uma pessoa P (raíz da AG é P), imprime no stdout os nomes de todos os bisavós de P conhecidos.
3. Especifique uma função que conte o número de pessoas incluídas numa determinada AG.

%———————————————————————————