Programação Imperativa
Exame de Recurso

1º Ano – LEI/LCC

10 de Julho de 2013
(Duração: 2h)

1. Defina uma função `int elimRep (int v[], int n)` que recebe um array `v` com `n` inteiros e elimina as repetições. A função deverá retornar o número de elementos do array resultante.

   Por exemplo, se o array `v` contiver nas suas primeiras 10 posições os números

   `{1, 2, 3, 2, 1, 4, 2, 4, 5, 4}`

   a invocação `elimRep (v,10)` deverá retornar `5` e colocar nas primeiras 5 posições do array os elementos `{1,2,3,4,5}`

2. Considere a seguinte definição de um tipo para representar listas ligadas de inteiros.

```
typedef struct llint {
    int valor;
    struct llint *prox;
} NodoL, *LLint;
```

   Defina uma função `int maximo (LLint l)` que calcula qual o maior valor armazenado numa lista não vazia.

3. Considere a seguinte definição de árvores binárias de inteiros,

```
typedef struct abint {
    int valor;
    struct abint *esq, *dir;
} NodoA, *ABint;
```

   Defina uma função `int iguais (ABint a, ABint b)` que testa se duas árvores são iguais (têm os mesmos elementos e a mesma forma).

4. O comando `cp` copia o conteúdo de um ficheiro para outro. Defina uma função `int cp (char *origem, char *destino)` que copia o contúdo do ficheiro com nome `origem` para o ficheiro com nome `destino`. A função retorna `0` se a operação for bem sucedida.

5. Defina uma função `void capitaliza (char s[])` que, dada uma *string*, substitui cada letra minúscula a seguir ao caracter '.' pela correspondente letra maiúscula. Tenha em atenção que entre a letra a substituir e o caracter '.' podem aparecer espaços.

   Por exemplo, se a *string* `a` contiver `"ola.  passei a programacao.  imperativa"` a invocação `capitaliza (a)` deverá transformar `a` na *string* `"Ola.  Passei a programação.  Imperativa"`.

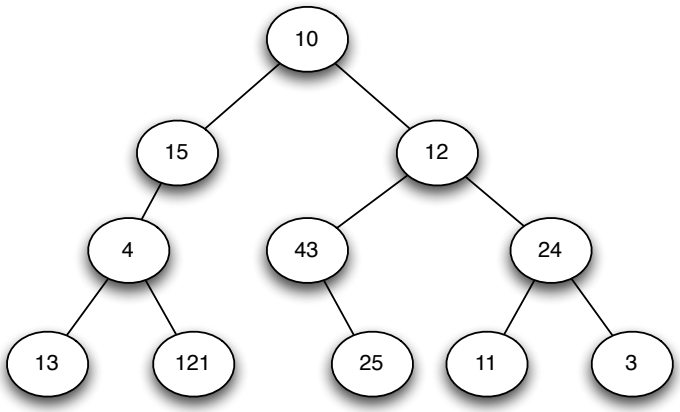
6. Considere a seguinte definição para representar listas de listas de inteiros.

```
typedef struct lllint {
    LLint lista;
    struct lllint *prox;
} NodoLL, *LLLint;
```

   Defina uma função `LLint concat (LLLint l)` que concatena todas as listas.

   A sua definição não deverá alocar qualquer espaço adicional e deverá libertar a memória usada pela lista argumento.

7. Defina uma função `LLint nivel (ABint a, int n)` que, dada uma árvore binária, constrói uma lista com os valores dos elementos que estão armazenados na árvore ao nível `n` (assuma que a raiz da árvore está ao nível `1`). Se `a` for a árvore apresentada abaixo, o resultado de `nivel (a, 3)` deve construir a lista com os elementos `{4,43,24}`, por esta ordem.



8. Defina uma função `int gPrimeDiv (int n)` que, dado um número inteiro maior ou igual a 2, calcula o maior número primo que é divisor de `n`.

   Por exemplo, `gPrimeDiv (1573)` deve dar como resultado `13` uma vez que `1573 = 11*11*13`.