

Ficha 9

Programação Imperativa

Árvores binárias

1. Relembre o problema proposto na Ficha anterior de calcular a palavra mais frequente de um texto.

Considere ainda que se implementa o tipo `Dicionario` como uma árvore binária de procura (ordenada pela palavra).

Apresente definições das funções

```
typedef struct abin {  
    char *pal;  
    int ocorr;  
    struct abin *esq, *dir;  
} *Dicionario;
```

- `void initDic (Dicionario *d)` que inicializa o dicionário a vazio.
- `int acrescenta (Dicionario *d, char *pal)` que acrescenta uma ocorrência da palavra `pal` ao dicionário `*d`. A função deverá retornar o número de vezes que a palavra `pal` passou a ter após a inserção.
- `char *maisFreq (Dicionario d, int *c)` que calcula a palavra mais frequente de um dicionário (retornando ainda em `c` o número de ocorrências dessa palavra).

2. Relembre as definições de listas ligadas e árvores binárias de inteiros.

```
typedef struct abin {  
    int valor;  
    struct abin *esq,  
                *dir;  
} *ABin;
```

```
typedef struct lista {  
    int valor;  
    struct lista *prox;  
} *LInt;
```

Defina as seguintes funções de conversão entre estes tipos de dados.

- `ABin fromList (LInt l)` que produz uma árvore binária de procura balanceada a partir de uma lista ordenada.

De forma a tornar essa função mais eficiente pode definir uma função auxiliar `ABin fromListN (LInt *el, int N)` que produz uma árvore binária de procura balanceada a partir dos `n` primeiros elementos da lista `*el`, deixando em `el` a lista dos restantes elementos.

- `LInt inorderL (ABin a)` que produz uma lista ligada ordenada a partir de uma árvore binária de procura.

De forma a tornar essa função mais eficiente pode definir uma função auxiliar `LInt inorderLAux (ABin a, LInt *end)` que coloca em `*end` o endereço do último elemento da lista produzida.

3. Considere o tipo `DLInt` para representar listas duplamente ligadas (em cada célula contem o endereço da próxima e da anterior).

```
typedef struct dlista {  
    int valor;  
    struct dlista *prox,  
                 *ant;  
} *DLInt;
```

Defina a função `DLInt inorderDL (ABin a)` que produz uma lista duplamente ligada ordenada a partir de uma árvore binária de procura (retornando o endereço do menor elemento da lista).