Universidade de Trás-os-Montes e Alto Douro

Escola de Ciências e Tecnologia. Departamento de Engenharias

LEINF

ano lectivo: **2021 / 2022**

unidade curricular: **Algoritmia** (e Estruturas de Dados)

Caderno de Exercícios

(Anexo 02): proposta de código base para Árvores Binárias

02

(Code: Generic Binary Trees)

```
//utilizar os #include necessários

//-----------------------------------------

#define DATA(node)      ((node)->pData)
#define LEFT(node)      ((node)->pLeft)
#define RIGHT(node)     ((node)->pRight)

#define EMPTY           NULL
#define NO_LINK         NULL

//-----------------------------------------

typedef enum _STATUS {ERROR=0,OK=1} STATUS;
typedef enum _BOOLEAN {FALSE=0,TRUE=1} BOOLEAN;

typedef struct _BTREE_NODE {
    void *pData;
    struct _BTREE_NODE *pLeft;
    struct _BTREE_NODE *pRight;
} BTREE_NODE;

typedef BTREE_NODE *BTREE;

//-----------------------------------------
//---declaracao de funcoes----------

STATUS initBTree(BTREE *);
BOOLEAN emptyBTree(BTREE);
BOOLEAN isLeaf(BTREE_NODE *);
STATUS createNewBTNode(BTREE_NODE **, void *);
void printIntBTree(BTREE);

STATUS insertIntBST(BTREE *, void *);
STATUS insertBT(BTREE *, void *, void *, void *);

//-----------------------------------------
```

```
/************************************************************
* Função initBTree(): Inicializa a árvore binária
*
* Parâmetros: pBT – árvore binária (passado por ref)
* Saída: STATUS
************************************************************/

STATUS initBTree(BTREE *pBT)
{
    *pBT=NULL;
    return OK;
}

/************************************************************
* Função emptyBTree(): verifica se a árvore binária está vazia
*
* Parâmetros: BT – árvore binária
* Saída: TRUE se a árvore binária estiver vazia, FALSE caso contrário
************************************************************/

BOOLEAN emptyBTree(BTREE BT)
{
    return (BT==NULL)? TRUE : FALSE;
}

/************************************************************
* Função isLeaf(): verifica se adeterminado nó da árvore binária é folha
*
* Parâmetros: pNode – apontador para nó (da árvore binária)
* Saída: TRUE se for uma folha, FALSE caso contrário
************************************************************/

BOOLEAN isLeaf(BTREE_NODE *pNode)
{
    return ((LEFT(pNode)==NULL) && (RIGHT(pNode)==NULL))? TRUE : FALSE;
}

/************************************************************
* Função printBTree(): apresenta no ecrã os elementos da árvore binária
*
* Parâmetros: BT – árvore binária
* Saída: void
*
* Esta função não é genérica: só funciona para elementos de tipo INT
************************************************************/

void printIntBTree(BTREE BT)
{
    if(emptyBTree(BT)==TRUE) return;

    printIntBTree(LEFT(BT));
    printf("%d, ",*(int *)DATA(BT));
    printIntBTree(RIGHT(BT));

    return;
}
```

```
/************************************************************
* Função insertIntBST(): insere um elemento (nó) numa BST
*
* Parâmetros: pBT – BST (passado por ref)
*                 pData - apontador genérico para os dados a inserir no nó criado
* Saída: OK se o nó foi inserido na BST e ERROR caso contrário
* Esta função não é genérica: só funciona para elementos de tipo INT
************************************************************/

STATUS insertIntBST(BTREE *pBT, void *pData)
{
    if(emptyBTree(*pBT))
    {
        BTREE_NODE *pNew;

        if(createNewBTNode(&pNew,pData)==ERROR)
            return ERROR;

        *pBT=pNew;
        return OK;
    }
    else if(*(int *)(pData) < *(int *)(DATA(*pBT)))
            insertIntBST(&(LEFT(*pBT)),pData);
    else insertIntBST(&(RIGHT(*pBT)),pData);

    return OK;
}

/************************************************************
* Função createNewBTNode(): cria um nó da árvore binária
*
* Parâmetros: pData - apontador genérico para os dados a inserir no nó criado
* Saída: apontador para o nó criado ou NULL em caso de erro
************************************************************/

STATUS createNewBTNode(BTREE_NODE **pNew, void *pData)
{
    BTREE_NODE *pTemp;

    if((pTemp=(BTREE_NODE *)malloc(sizeof(BTREE_NODE)))==NULL)
        return ERROR;

    *pNew=pTemp;
    DATA(pTemp)=pData;
    LEFT(pTemp)=NULL;
    RIGHT(pTemp)=NULL;

    return OK;
}
```