utad Universidade de Trás-os-Montes e Alto Douro

Escola de Ciências e Tecnologia. Departamento de Engenharias

		%
1	6	
2	7	
3	7	

Eng. Informática

unidade curricular: **Algoritmia** ano lectivo: **2018 / 2019**

Teste Prático 02. **29.Mai.2019.** duração: **60 min** (+5 min tolerância)
(sem consulta)

Número: _____ Nome: _____

Considere o seguinte programa em linguagem C:

```
typedef struct _PERSON{
   char     name[50];
   int      age;
   BOOLEAN alive;
}PERSON;

typedef struct _BTREE_NODE
{
  void * data;
  struct _BTREE_NODE * left;
  struct _BTREE_NODE * right;
} BTREE_NODE;

#define DATA(node)    ((node)->data)
#define LEFT(node)    ((node)->left)
#define RIGHT(node)   ((node)->right)
typedef BTREE_NODE * BTREE;
typedef enum _BOOLEAN {FALSE = 0,TRUE = 1} BOOLEAN;
typedef enum _STATUS {ERROR = 0,OK = 1} STATUS;

int main(int argc, char *argv[])
{
   BTREE btree;
   void *   persons[15];
   char  file_name[20];

   printf("Nome do ficheiro: ");
   scanf("%s", file_name);
   if(ReadFile(persons,file_name))
   {
      btree = CreateBtree(persons,0,15);









      BtreeFree(btree);
   }
   else
      printf("ERRO na leitura do ficheiro\n");
   return 1;
}
```

Invoque as funções necessárias ao funcionamento do programa, a partir do `main`

```
BTREE_NODE * NewBtreeNode(void * data)
{
    BTREE_NODE * tmp_pt;
    if ((tmp_pt = (BTREE_NODE *)malloc(sizeof(BTREE_NODE)))!=NULL)  {
          DATA(tmp_pt) = data;
          LEFT(tmp_pt) = RIGHT(tmp_pt) = NULL;  }
    return tmp_pt;
}
void BtreeFree(BTREE btree)
{
    if (btree != NULL)  {
         BtreeFree(LEFT(btree));
         BtreeFree(RIGHT(btree));
         Free(DATA(btree));
         free(btree);  }
}
BTREE_NODE *InitNode(void * ptr_data,BTREE_NODE * node1,BTREE_NODE * node2)
{
    BTREE_NODE * tmp_pt = NULL;
    tmp_pt = NewBtreeNode(ptr_data);
    LEFT(tmp_pt) = node1;
    RIGHT(tmp_pt) = node2;
    return(tmp_pt);
}
STATUS ReadFile(void ** persons, char * file_name)
{
    FILE * fp;
    Int aux,j, i = 0;
    void * ptr_data;

    if ((fp = fopen(file_name, "r")) != NULL)  {
          while (!feof(fp)) {
              if ((ptr_data = malloc(sizeof(PERSON))) != NULL)  {
                  fscanf(fp, "%[^;];%d;%d\n", ((PERSON *)ptr_data)->name, &((PERSON *)ptr_data)->age,&aux);
                  if(aux) ((PERSON *)ptr_data)->alive = TRUE;
                  else ((PERSON *)ptr_data)->alive = FALSE;
                  persons[i] = ptr_data;
                  i++;  }
              else {
                  for (j = i; j >= 0; j--)
                      free(persons[j]);
                  return(ERROR);  }
          }
          fclose(fp);
          return(OK);  }
    else
          return(ERROR);
}
BTREE_NODE *CreateBtree(void ** v, int i, int size)
{
    if( i >= size)    return(NULL);
    else              return(InitNode(*(v+i),CreateBtree(v,2*i+1,size),CreateBtree(v,2*i+2,size)));
}
```

Suponha um programa que, através de uma árvore binária genérica, representa de forma hierárquica uma família (árvore geneológica). Os dados de cada pessoa são: nome, idade e um boolean indicando se a pessoa está viva (TRUE) ou não (FALSE).

Eng. Informática. **Teste Prático 02. 29.Mai.2019.**
unidade curricular: **Algoritmia** ano lectivo: **2018 / 2019**

01 Desenvolva o código necessário para contar o número de pessoas da árvore que estão vivas (alive = TRUE).

02 Desenvolva o código necessário para alterar o nome de uma pessoa cujo nome original, e o final, devem ser lidos através do teclado.

03 Desenvolva o código necessário para acrescentar 2 nós de árvore (filho da esquerda e da direita) à folha mais à direita da árvore. NOTA: os dados dos novos nós devem ser introduzidos através do teclado.