AULAS 9 E 10 - ÁRVORES BINÁRIA DE PESQUISA

O tipo de dados abstrato ABP é constituído pelo ficheiro de interface **abp.h** e pelo ficheiro de implementação **abp.c** e implementa a manipulação de árvores binárias de pesquisa que armazenam números inteiros, com capacidade de múltipla instanciação e controlo centralizado de erros.

- Comece por testar convenientemente toda funcionalidade do tipo de dados, usando para esse efeito o programa **testabp.c** e a *makefile* **mkabp** e os ficheiros de árvores disponibilizados: **arv0.txt**, **arv1.txt**, **arv2.txt** e **arv3.txt**. Pode também utilizar o programa de simulação **simabp.c** que permite inserir e remover números da árvore exibindo-a no monitor após cada operação.

Acrescente ao tipo de dados abstrato as funcionalidades seguintes.

Nota: As funções devem ser implementadas de modo eficiente, sem visitar nós desnecessários.

- Uma **função recursiva** para apagar o elemento mínimo armazenado na árvore. A função deve ter o seguinte protótipo:

```
void ABPDeleteMin (PtABPNode *proot);
```

- Uma **função repetitiva** para apagar o elemento máximo armazenado na árvore. A função deve ter o seguinte protótipo:

```
void ABPDeleteMax (PtABPNode *proot);
```

- Uma **função repetitiva** para obter um ponteiro para o nó do elemento `pvalue` ou, caso ele não exista na árvore, para o nó do maior dos elementos menores do que ele. Se esse nó não existir, deverá ser devolvido o **ponteiro nulo**. A função deve ter o seguinte protótipo:

```
PtABPNode ABPFloorValue (PtABPNode proot, int pvalue);
```

- Uma **função recursiva** para obter um ponteiro para o nó do elemento `pvalue` ou, caso ele não exista na árvore, para o nó do menor dos elementos maiores do que ele. Se esse nó não existir, deverá ser devolvido o **ponteiro nulo**. A função deve ter o seguinte protótipo:

```
PtABPNode ABPCeilValue (PtABPNode proot, int pvalue);
```

- Uma **função recursiva** para determinar o número de elementos inferiores a `pvalue armazenados na árvore` – **sugestão:** será útil usar a função **ABPSize** já existente no tipo de dados ABP. A função deve ter o seguinte protótipo:

```
int ABPRank (PtABPNode proot, int pvalue);
```

- Uma **função recursiva** para registar ordenadamente, numa **fila de inteiros** já existente e referenciada pelo ponteiro pqueue, os elementos da árvore pertencentes ao intervalo fechado [`plow`, `phigh`]. A função deve ter o seguinte protótipo:

```
void ABPElements (PtABPNode proot, PtQueue pqueue, int plow, int phigh);
```

- Uma **função repetitiva** para determinar se a árvore é alternadamente composta por números pares e ímpares ou ímpares e pares, fazendo uma travessia em ordem, recorrendo a uma pilha. **Tenha em atenção que a função – após as validações necessárias – deve começar por determinar a paridade do elemento mínimo da árvore**. A função deve ter o seguinte protótipo:

```
int ABPIsEvenOdd (PtABPNode proot);
```

Para testar estas funções pode utilizar o programa de simulação **trababp.c**.