Lista de Exercícios de Estruturas de Dados e Algoritmos

Dada a seguinte representação de uma árvore binária:

```
typedef struct ab{
      int info;
      struct ab *esq, *dir;
}TAB;
```

Escreva as seguintes funções:

(Q1) cópia de uma árvore: **`TAB* copia (TAB *a);`**

(Q2) espelho de uma árvore (o que está a esquerda na árvore original, estará a direita no espelho, e vice-versa): **`TAB* espelho (TAB *a);`**

(Q3) maior elemento da árvore: **`TAB* maior(TAB *a);`**

(Q4) menor elemento da árvore: **`TAB* menor(TAB *a);`**

(Q5) uma função que, dadas duas árvores deste tipo, testa se estas árvores são iguais. A função retorna um se elas são iguais e zero, caso contrário. A função deve obedecer ao seguinte protótipo: **`int igual (TAB* a1, TAB* a2);`**

(Q6) uma função em C que, dada uma árvore binária qualquer, retire todos os elementos pares da árvore original. A função deve ter o seguinte protótipo: **`TAB* retira_pares (TAB* arv);`**

(Q7) se esta estrutura **`TAB`** tivesse um campo cor (**`int cor`**), defina uma função em C que, ao receber uma árvore binária "sem cor" e totalmente balanceada (isto é, a distância da raiz a qualquer folha da árvore é sempre a mesma), retorne esta árvore com os nós coloridos somente de vermelho e preto, sendo que o nó pai NUNCA pode ter a mesma cor de seus filhos. A função deve possuir o seguinte protótipo: **`void colore (TAB* arv);`**

(Q8) descubra a quantidade de nós internos: **`int ni(TAB *a);`**

(Q9) ache a quantidade de nós folha: **`int nf(TAB *a);`**

(Q10) refaça as operações anteriores usando uma árvore binária de busca (ABB) no lugar de uma árvore binária.