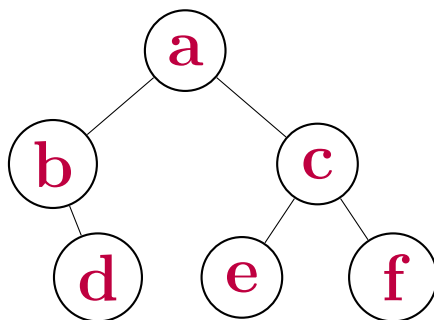


Lista de exercícios

Exercício 1) Considerando a árvore abaixo, escreva funções para imprimir os nós considerando os seguintes tipos de percursos:

- pré-ordem
- in-ordem
- pós-ordem



```
Arvore *a = constroi_arv ('a',  
    constroi_arv('b',  
        cria_arv_vazia(),  
        constroi_arv('d', cria_arv_vazia(), cria_arv_vazia())  
    ),  
    constroi_arv('c',  
        constroi_arv('e', cria_arv_vazia(), cria_arv_vazia()),  
        constroi_arv('f', cria_arv_vazia(), cria_arv_vazia())  
    )  
);
```

Exercício 2) Árvores binárias podem ser descritas de acordo com a seguinte notação textual: a árvore vazia é representada por <>, e árvores não-vazias, por < raiz esq dir >. Com essa notação, a árvore do exercício 1) é representada por:

<a<b<><d<><>>><c<e<><>>><f<><>>>>>

Escreva uma função que recebe uma árvore como entrada e a imprime utilizando esta notação. Use o seguinte protótipo para a sua função:

```
void imprime_arv_marcadores (Arvore *a);
```

Exercício 3) Escreva uma função que retorna um valor booleano (um ou zero) que indica a ocorrência ou não de um dado caractere na árvore. Considere o seguinte protótipo para a sua função:

```
int pertence_arv (Arvore *a, char c);
```

onde char **c** é o caractere que deve ser procurado na árvore **a**.

Exercício 4) Escreva uma função que conte o número de nós de uma árvore binária. Utilize o seguinte protótipo para a sua função:

```
int conta_nos (Arvore *a);
```

Exercício 5) Escreva uma função que calcula a altura de uma árvore binária. Utilize o seguinte protótipo para a sua função:

```
int calcula_altura_arvore (Arvore *a);
```

Exercício 6) Escreva uma função que conta o número de nós folhas em uma árvore binária. Utilize o seguinte protótipo para a sua função:

```
int conta_nos_folha (Arvore *a);
```