MC202 (Estruturas de Dados) - 1s2021

Tarefa 07: Árvores binárias

Profº Ricardo Dahab
Assistente: Elisa Dell'Arriva
Assistente: Jônatas Trabuco Belotti

Instituto de Computação - UNICAMP

Sobre a Tarefa

O objetivo desta tarefa é que o(a) aluno(a) se familiarize com o uso de árvores binárias e seus diferentes tipos de percurso. O exercício consiste em fazer a leitura de uma árvore binária, através dos seus elementos em sequência pré-ordem e inordem, reconstruir a árvore e a imprimir em sequência pós-ordem.

Reconstrução de árvore binária

Note que não é possível resolver esse problema olhando somente para uma das sequências de nós. Por isso, é preciso analisá-las em conjunto. Começamos com uma árvore vazia e vamos inserindo os nós de acordo com a sequência em pré-ordem, mas consultando a sequência inordem para inserí-lo no lugar correto.

Suponha que acabamos de inserir o k-ésimo nó, chamado v_k, na árvore. Agora, precisamos inserir o $(k+1)$-ésimo nó, chamado v_{k+1}. O nó v_{k+1} pode estar na subárvore esquerda do nó v_k, na subárvore direita do nó v_k ou em outra subárvore completamente diferente. Para inseri-lo no lugar correto é preciso saber quais nós pertencem a cada subárvore. Nesse momento é preciso encontrar o nó v_k na sequência inordem. Em seguida, dividimos a sequência inordem em duas subsequências: uma com os nós a esquerda de v_k e outra com os nós à direita de v_k. Agora basta tentar encontrar v_{k+1} nessas duas subsequências, se v_{k+1} estiver na primeira subsequência, então ele é filho a esquerda de v_k, se v_{k+1} estiver na segunda subsequência, então ele é filho a direita de v_k. Pode-se então inserir v_{k+1} e recursivamente analisar suas subárvores.

Como exemplo, considere uma árvore de 5 nós com as seguintes sequências pré-ordem e inordem:

```
a b c d e
c b a e d
```

Pela natureza das sequências, sabemos que o primeiro elemento da sequência pré-ordem é a raiz da árvore. Portanto, podemos iniciar a árvore com o elemento `a` como raiz.

Seguindo na sequência pré-ordem, agora precisamos inserir o elemento `b` na árvore. Apenas olhando a sequência pré-ordem não tem como saber se ele é filho a esquerda ou a direita de `a`. Para isso precisamos analisar a sequência inordem. Dividimos a sequência inordem em duas subsequências: uma com os nós a esquerda de `a` e outra com os nós à direita de `a`:

```
[c b] a [e d]
```

Como `b` está na primeira subsequência, devemos o inserir como filho a esquerda de `a`.

Agora vamos inserir o elemento `c`. Novamente dividimos a sequência inordem em duas subsequências a partir do nó `a` e verificamos em qual delas o elemento `c` está. Mais uma vez, o elemento está na primeira,

entretanto, agora o nó `a` já tem um filho a esquerda, então dividimos agora a primeira subsequência em outras duas a partir do nó `b`:

```
[c] b []
```

Como o elemento `c` está na primeira subsequência, o inserimos como filho a esquerda do nó `b`.

Dando continuidade nos elementos da sequência pré-ordem, agora é hora de inserir o nó `d`. Mais uma vez dividimos a sequência inordem em duas subsequências a partir do nó `a` e procuramos em qual delas o nó `d` está. Como o nó `d` está na segunda subsequência e, o nó `a` ainda não tem um filho a direito, inserimos `d` como filho a direta do nó `a`.

Por fim, o último elemento da sequência pré-ordem a ser inserido na árvore é o elemento `e`. Dividimos a sequência inordem em duas subsequências, `e` está na segunda, mas `a` já tem um filho a direita, portanto, subdividimos a segunda subsequência em mais duas a partir do nó `d`:

```
[e] d []
```

Como o nó `e` está na primeira subsequência, o colocamos como filho a esquerda do nó `d`.

Ao final de todos os passos, a árvore resultante é apresentada na Figura 1.

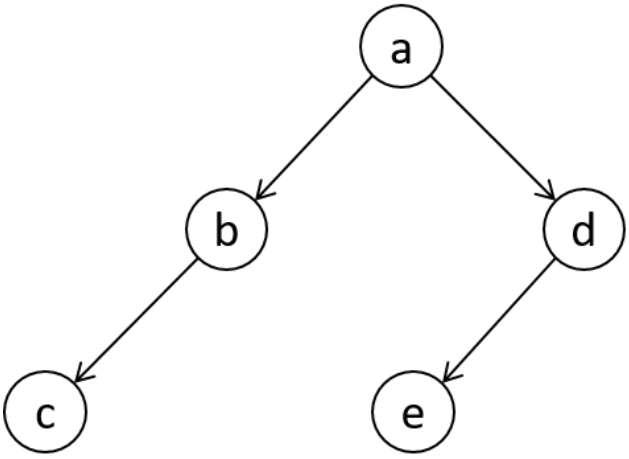

Figura 1: Resultado da reconstrução da árvore a partir das sequências pré-ordem e inordem.

Formato da entrada e saída

A entrada consiste em 3 (três) linhas, a primeira com um número inteiro de 1 até 50, que representa a quantidade de nós da árvore. Na segunda, os valores dos nós ordenados em pré-ordem; na terceira, os valores dos nós ordenados em inordem. Todos os valores de nós são letras do alfabeto.

Como saída você deve imprimir a sequência pós-ordem da árvore reconstruída, com um espaço em branco entre cada nó. Pela natureza recursiva do algoritmo, também terá um espaço em branco após o último nó impresso na linha. O último caractere da saída deve obrigatoriamente ser um `\n`.

Exemplo 1.

Entrada:

```
5
a b c d e
c b a e d
```

Saída:

```
c b e d a
```

Exemplo 2.

Entrada:

```
4
a B c D
D c B a
```

Saída:

```
D c B a
```

Exemplo 3.

Entrada:

```
7
b c g a f d e
g c a b d f e
```

Saída:

```
g a c d e f b
```

Importante

- A árvore deve obrigatoriamente ser implementada utilizando listas ligadas. Vocês não podem, em momento algum, utilizar qualquer outro tipo de estrutura de dados para representar nós e árvores. O não cumprimento dessa exigência resultará em nota 0 (zero) na sua tarefa.
- É obrigatório liberar toda memória alocada dinamicamente. Para fazer essa verificação, recomentamos o uso da ferramenta *valgring*. Caso haja memória não liberada, serão descontados 2 (dois) pontos da nota final desta tarefa.

Alguns avisos e lembretes

- A página da disciplina no SuSy é `https://susy.ic.unicamp.br:9999/mc202abc`.
- Para submeter, utilizem somente os dígitos numéricos do RA e a senha da DAC.
- O número máximo de submissões é 20.

- Esta tarefa tem peso 2 (dois).
- A tarefa estará aberta até 14/06/2021, às 23h59.