



Estrutura de Dados para Automação Exercícios sobre Árvores AVL

Prof. Rodrigo da Silva Guerra

13 de junho de 2018

BRUNO GABRIEL FLORES SAMPAIO

I Considerando o algoritmo abaixo que implementa a inclusão de elementos numa árvore binária de busca, responda:

```
1  int data[15];
2
3  void init_tree()
4  {
5      int i;
6      for (i = 0 ; i < 15 ; i++)
7      {
8          data[i] = -1;
9      }
10 }
11
12 int left_child(int node)
13 {
14     return 2*node+1;
15 }
16
17 int right_child(int node)
18 {
19     return 2*node+2;
20 }
21
22 void save_number(int root, int number)
23 {
24     if (data[root] == -1)
25     {
26         data[root] = number;
27     } else {
28         if (number > data[root])
29         {
30             save_number(right_child(root), number);
31         } else {
32             save_number(left_child(root), number);
33         }
34     }
35 }
```

I.a. Observe a árvore ilustrada na Figura 1 e preencha a Tabela 1 supondo que a árvore foi construída seguindo o algoritmo de inserção apresentado (2 pontos).

data	41	27	81	7	38	67	84	2	11	33	-1	62	-1	82	92
------	----	----	----	---	----	----	----	---	----	----	----	----	----	----	----

Tabela 1: Array referente ao exercício I.a.

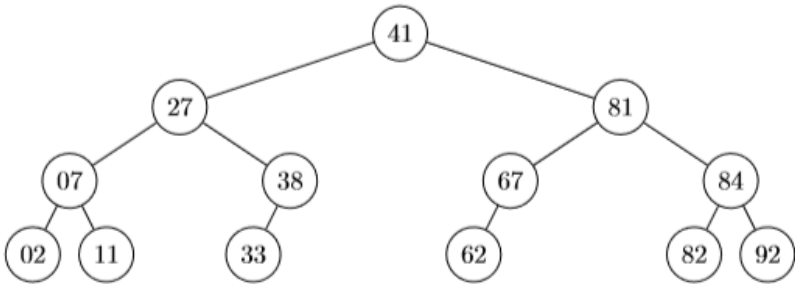


Figura 1: Árvore AVL

II.b O código mostrado suporta árvores de até 4 níveis. Se desejarmos alterar este código para suportar uma árvore de até 17 níveis, qual é o tamanho do array que deveria ser alocado neste caso? (2 pontos)

$$(2^{17})-1 = 131071$$

```
int dados[131071];
```

II.a Suponha agora que a árvore da Figura 1 é uma árvore AVL, onde o fator de equilíbrio é sempre corrigido através de rotações. Desejamos incluir o número 30. Desenhe abaixo como ficará a árvore após essa inclusão (2 pontos).

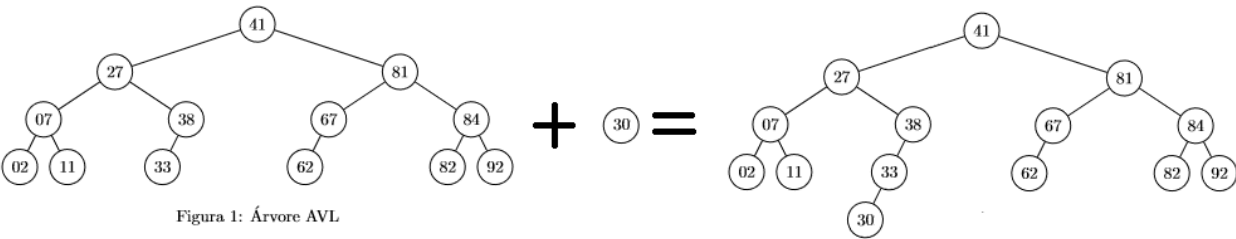
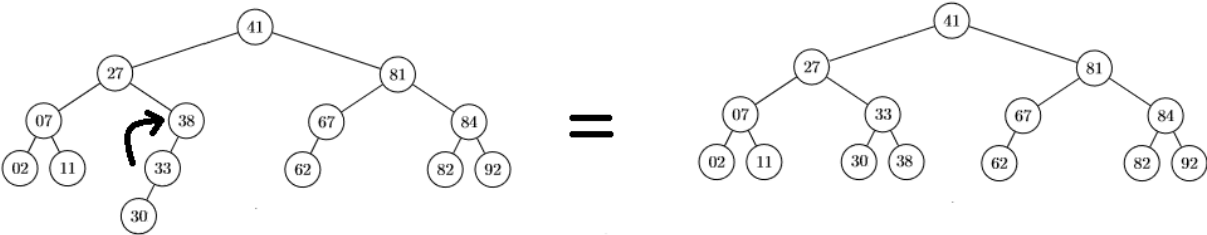


Figura 1: Árvore AVL

ROTACIONANDO



II.b A partir da árvore AVL da Figura 1, desenhe esta ficará organizada após a remoção do número 41 (2 pontos).

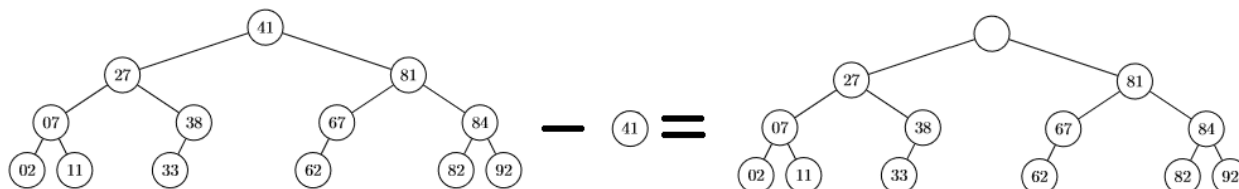
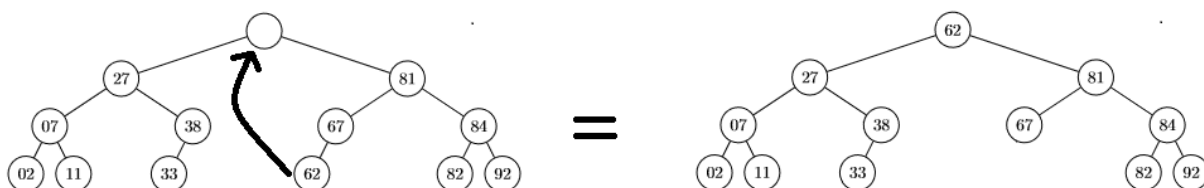


Figura 1: Árvore AVL

ROTACIONANDO



III Supondo a árvore ilustrada na Figura 1, e o trecho de programa abaixo. Supondo que fizemos uma chamada inicial para a função `show(0)` (com argumento zero), escreva a saída esperada na tela (2 pontos).

```

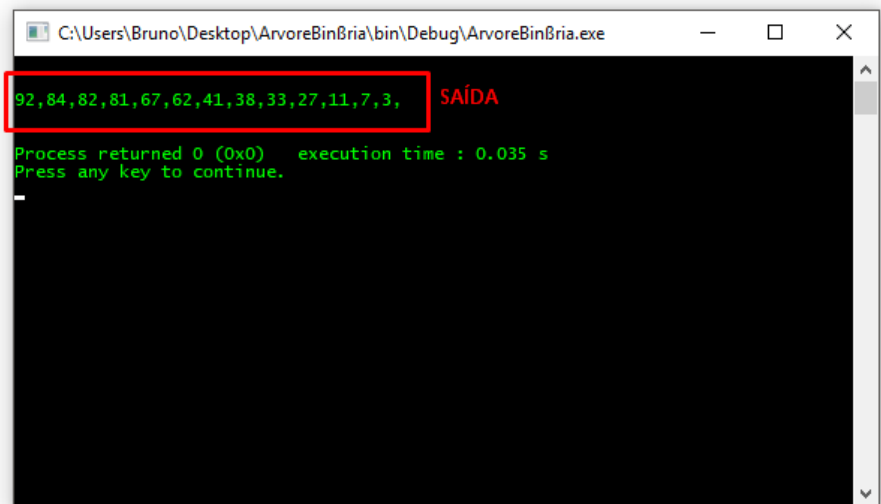
1 void show(int node)
2 {
3     if (data[node] != -1)
4     {
5         show(right_child(node));
6         printf("%d, ", data[node]);
7         show(left_child(node));
8     }
9 }

```

```

38
39 int main() {
40
41     init_arv();
42
43     save_number(0, 41);
44     save_number(0, 27);
45     save_number(0, 81);
46     save_number(0, 7);
47     save_number(0, 38);
48     save_number(0, 67);
49     save_number(0, 84);
50     save_number(0, 3);
51     save_number(0, 11);
52     save_number(0, 33);
53     save_number(0, 62);
54     save_number(0, 82);
55     save_number(0, 92);
56
57     printf("\n\n");
58
59     show(0);
60
61     printf("\n\n");

```



```

C:\Users\Bruno\Desktop\ArvoreBinBria\bin\Debug\ArvoreBinBria.exe
92, 84, 82, 81, 67, 62, 41, 38, 33, 27, 11, 7, 3, SAÍDA
Process returned 0 (0x0)   execution time : 0.035 s
Press any key to continue.

```