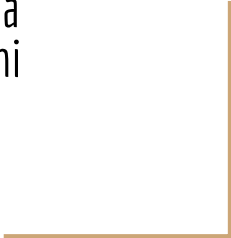




# Manipulação em Árvore B

Prof. Joaquim Uchôa  
Profa. Juliana Gregghi  
DAC/UFLA



# Objetivo

Manipular uma árvore B, inserindo um conjunto de valores e depois removendo parte desses valores inseridos.

A árvore terá no mínimo 2 e no máximo 5 chaves em cada bloco, com ordenação crescente dos valores. Quando houver necessidade, o bloco será dividido antes da inserção.

Quando da remoção, em caso de rotação, a preferência será pelo irmão à esquerda, o mesmo vale para fusão.

Quando a remoção não é em nó folha, o valor é substituído pelo sucessor.

# Valores a serem inseridos

100	61	46	40	66	43	72	24	7
52	23	5	65	27	84	55	38	62
45	33	68	10	29	3	92	11	32
53								

Situação Inicial:  
Raiz vazia

--	--	--	--	--

Situação Inicial:  
Raiz vazia



Em um arquivo, o cabeçalho da árvore B indicaria essa situação informando que a árvore não possui qualquer bloco válido e que a raiz ainda não foi gravada (ou teria um bloco com a raiz vazia em si).

Situação Inicial:  
Raiz vazia



Ainda considerando implementação em arquivos, o cabeçalho poderia também estar num arquivo auxiliar. Mas geralmente, utiliza-se o início do arquivo da árvore B.

Informação mínima que ele armazena: endereço no disco da página (bloco) da raiz e o número de blocos utilizados (até para facilitar a inserção de novos blocos no final).

Inserindo o primeiro elemento (100)

100				
-----	--	--	--	--

Inserindo o primeiro elemento (100)

100				
-----	--	--	--	--

A raiz é o único bloco em que é permitido um número menor de chaves que aquele especificado pela ordem da árvore.

Neste exemplo, a raiz é a única página que pode ter menos que 2 chaves.



Inserindo o primeiro elemento (100)

100				
-----	--	--	--	--

Um bloco contém, pelo menos, as seguintes informações:

- número de chaves válidas
- a informação se é ou não folha
- no caso de não ser folha, o endereço dos filhos (geralmente um vetor com uma posição a mais que o número máximo de chaves)

Inserindo o primeiro elemento (100)

100				
-----	--	--	--	--

Aqui estamos falando em inserção de valores... Mas geralmente são registros grandes, com vários campos/atributos. E a ordenação é feita por um atributo específico, a **chave**.

Inserindo o segundo elemento (61)

100				
-----	--	--	--	--

Quando da inserção, vamos até o último valor do bloco e verificamos se é maior ou não que o elemento a ser inserido. Se é maior, então ele é deslocado e repetimos o processo até achar a posição de inserção do valor.

Esse método é uma simplificação do Insertion Sort.

Inserindo o segundo elemento (61)

	100			
--	-----	--	--	--

Com os valores maiores deslocados, temos o espaço para inserção liberado.

Inserindo o segundo elemento (61)

61	100			
----	-----	--	--	--

Inserindo mais 3 valores: 46, 40 e 66

61	100			
----	-----	--	--	--

A inserção dos 3 próximos valores é relativamente simples, aqui vamos pular o passo-a-passo, para agilizar o processo...

Para cada inserção basta achar a posição correta, para manter o bloco ordenado.

Inserindo mais 3 valores: 46, 40 e 66

40	46	61	66	100
----	----	----	----	-----

Inserindo o 43

40	46	61	66	100
----	----	----	----	-----

Como o bloco em que o 43 será inserido está cheio, é necessário que ele seja dividido...

Isso pode ser feito de duas maneiras:

- O bloco é dividido ao meio antes da inserção.
- O valor é inserido, estourando temporariamente o bloco, que é dividido logo após a inserção.



Inserindo o 43

40	46	61	66	100
----	----	----	----	-----

O “estouro” só pode ocorrer em memória... Um estouro em arquivos poderia fazer com que dados de um bloco sobrescrevessem o próximo.

Mas em geral árvores B são trabalhadas dessa forma: os dados são lidos do arquivo para a memória para que a operação desejada seja realizada (inserção, busca, remoção, etc.)

o está

o...

a inserção.

vidido logo

Inserindo o 43

40	46	61	66	100
----	----	----	----	-----

Quando a página tem um número ímpar de elementos, é mais fácil visualmente (e mais claro na implementação) dividir antes de inserir.

Quando é o oposto (o bloco tem um número par de elementos), pode ser mais fácil inserir antes (estourando) e dividir depois...

Indiferente da escolha, essa precisa ser adotada em todo o processo, sem ser modificada.

Inserindo o 43

40	46	61	66	100
----	----	----	----	-----

Vamos dividir antes de inserir, ...

Inserindo o 43

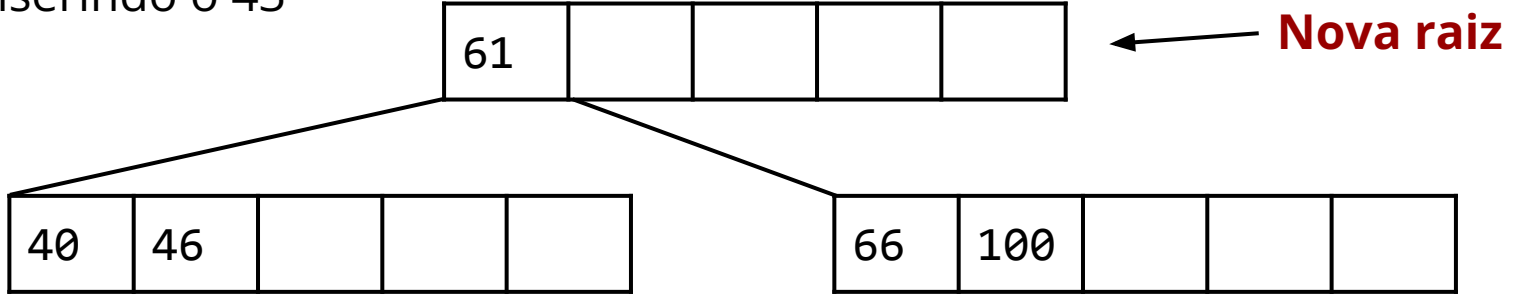
40	46	61	66	100
----	----	----	----	-----

--	--	--	--	--

Metade dos dados (a segunda metade) são copiados para um novo bloco e o elemento “central” é promovido para o bloco superior.

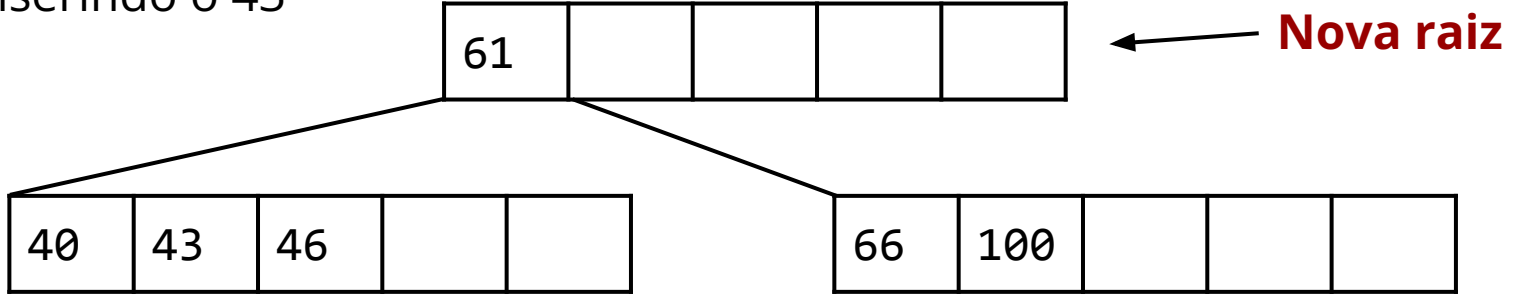
Se não há um bloco superior (a página sendo dividida é a raiz), então o valor promovido gera uma nova raiz

Inserindo o 43

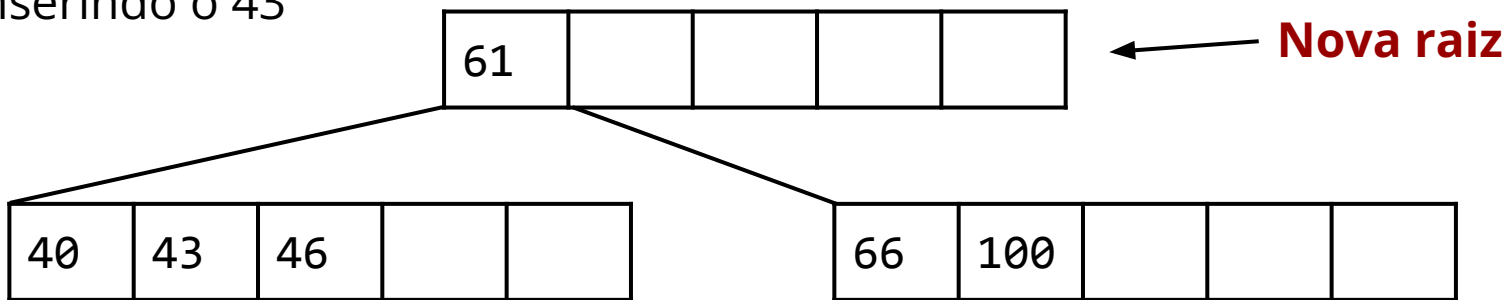


Com o bloco dividido, agora podemos fazer a inserção como de costume...

Inserindo o 43



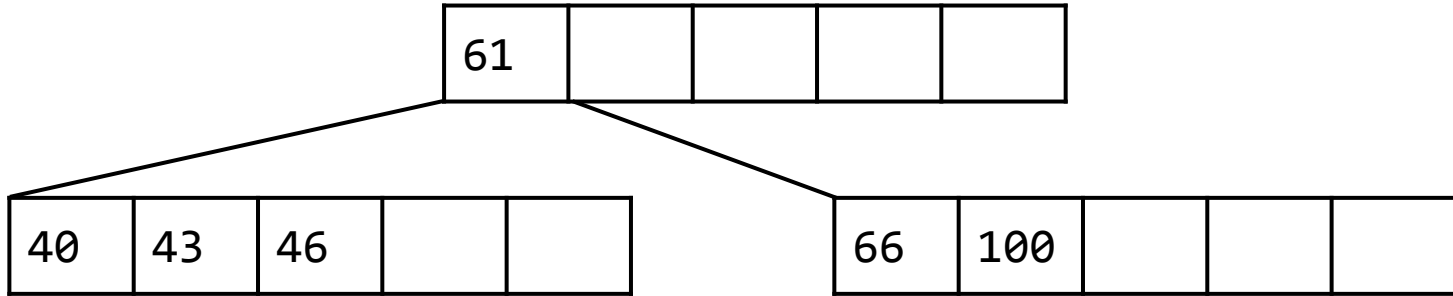
Inserindo o 43



A ligação do bloco aos filhos é feita armazenando (geralmente em um array) a posição de cada filho.

Percebam que há um filho a mais que o número de chaves.

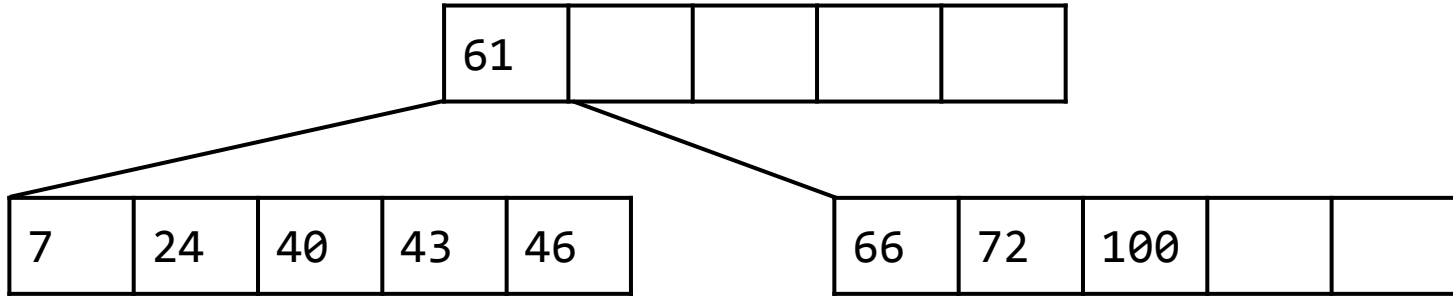
Inserindo 72, 24 e 7



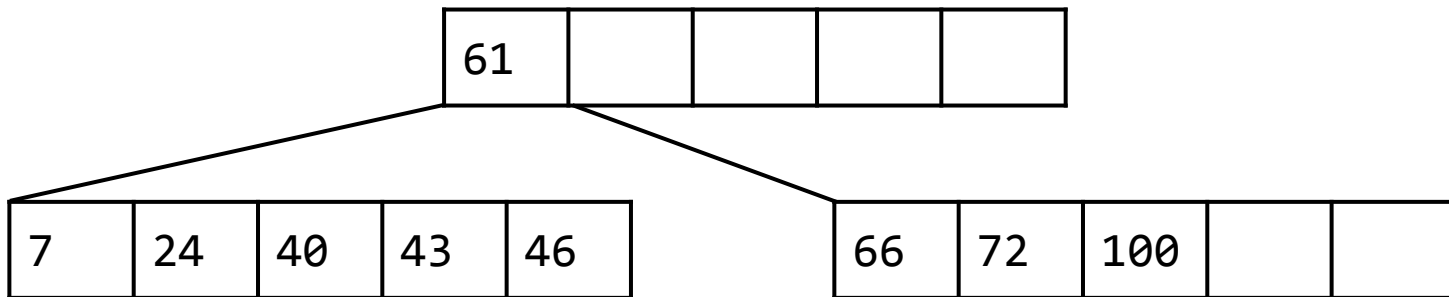
Inserções normais, ...



Inserindo 72, 24 e 7



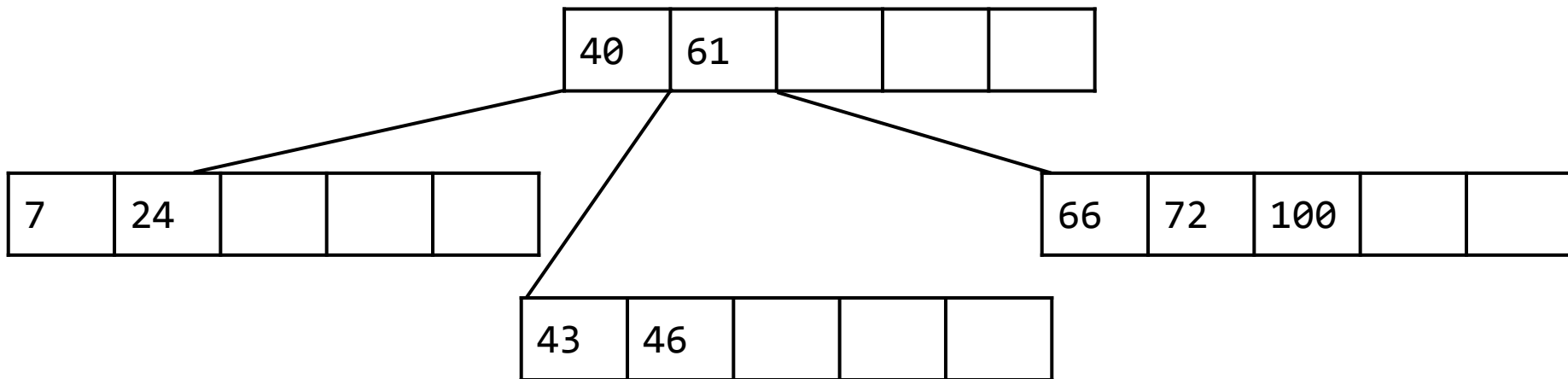
Inserindo 52



**O bloco precisa ser dividido...**

Nesse caso, vamos promover a chave central, com valor 40.

Inserindo 52

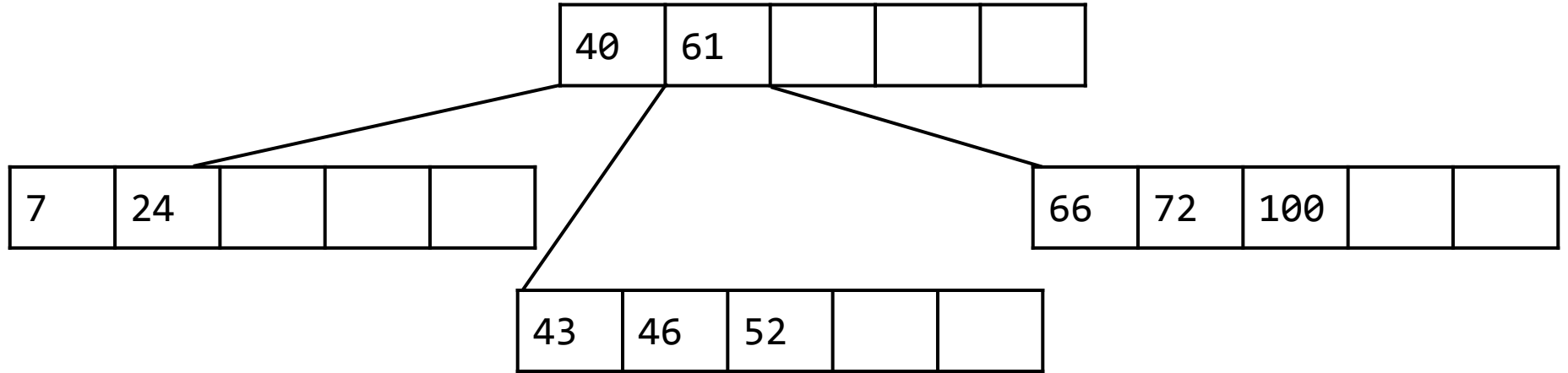


### **O bloco precisa ser dividido...**

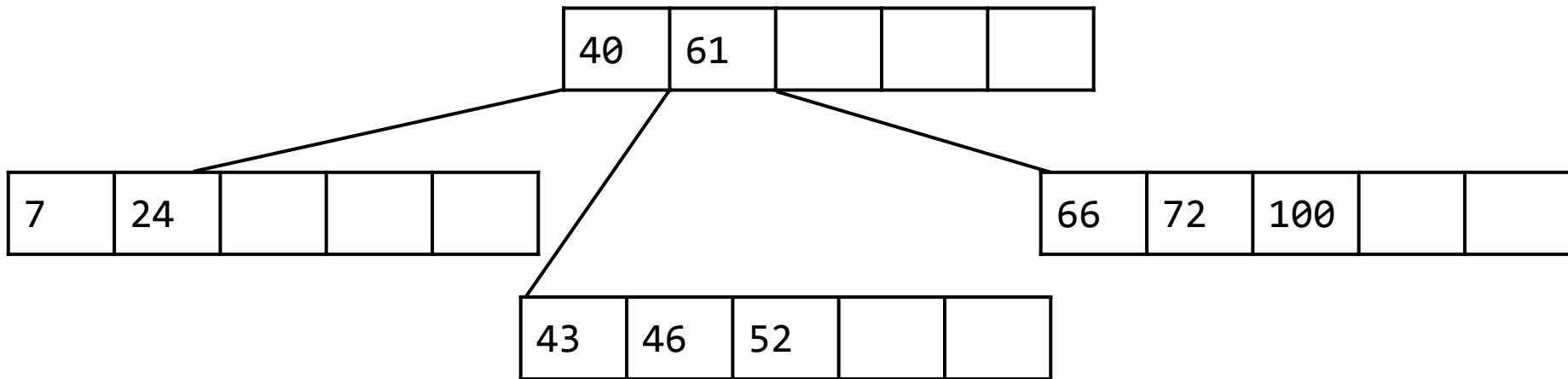
Nesse caso, vamos promover a chave central, com valor 40.

Essa chave é inserida no bloco superior usando a mesma abordagem de procurar a posição... Mas note que os valores e os filhos são movidos...

Inserindo 52

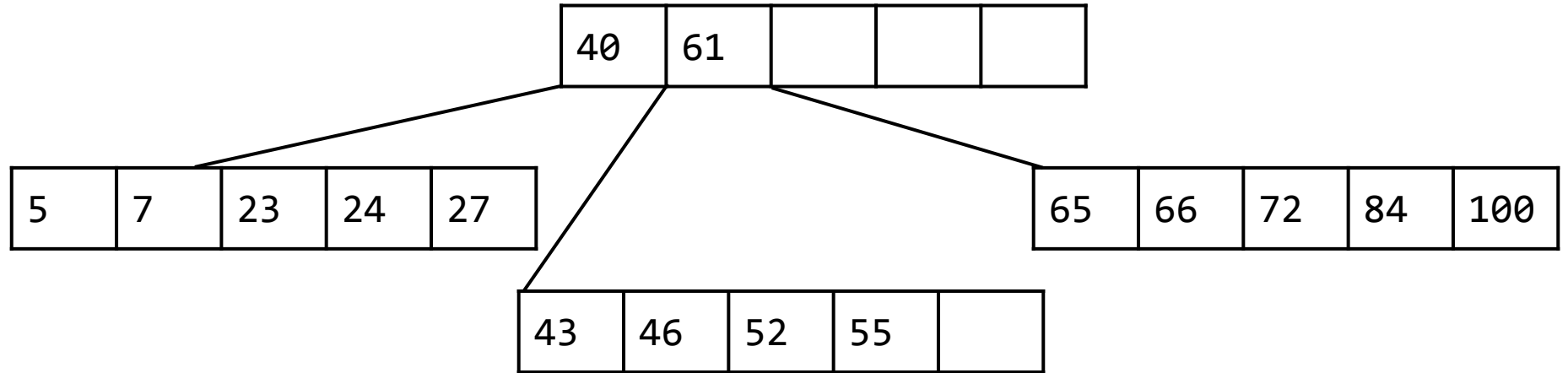


Inserindo 23, 5, 65, 27, 84 e 55

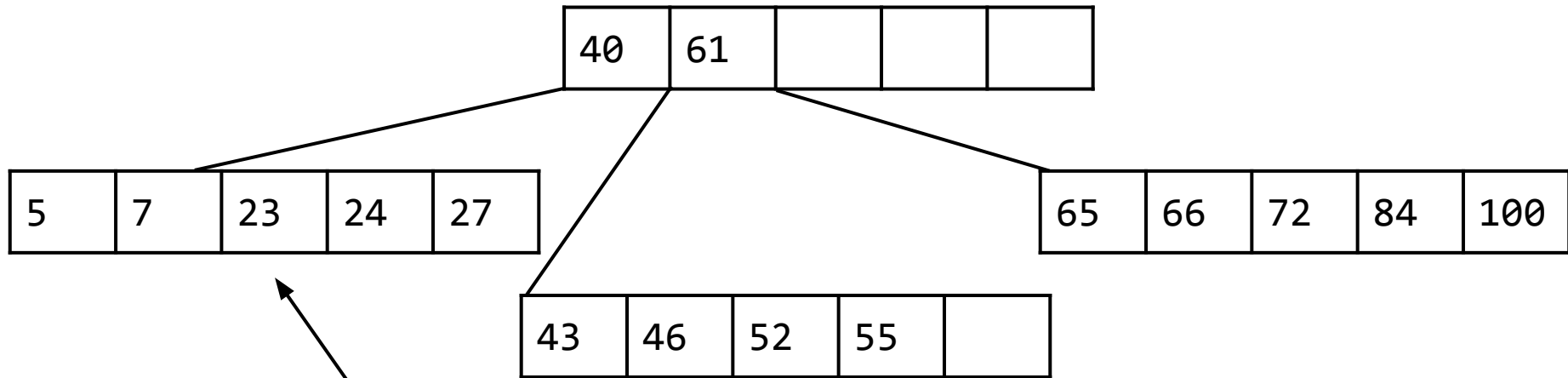


Inserções normais, ...

Inserindo 23, 5, 65, 27, 84 e 55

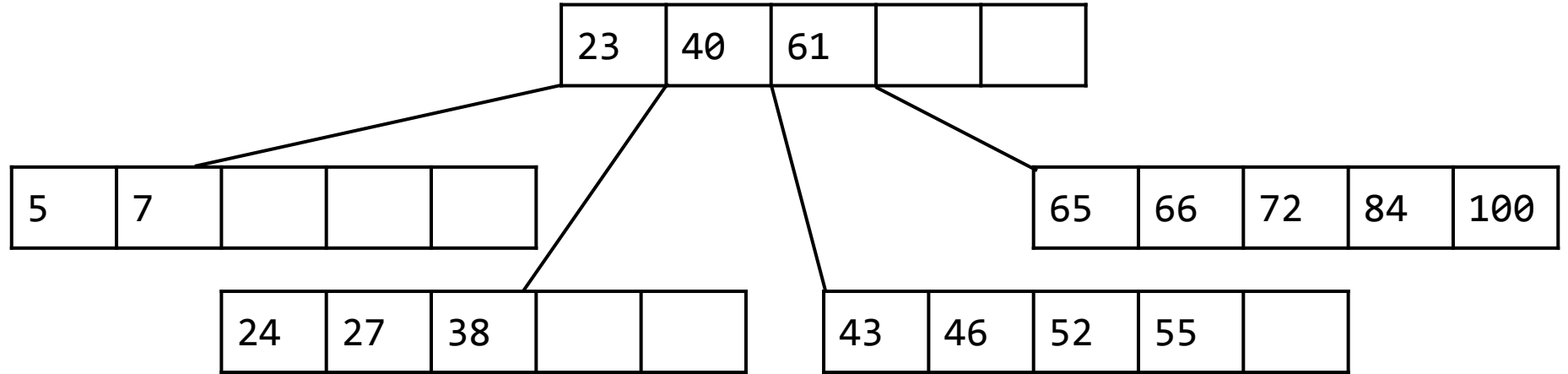


Inserindo 38



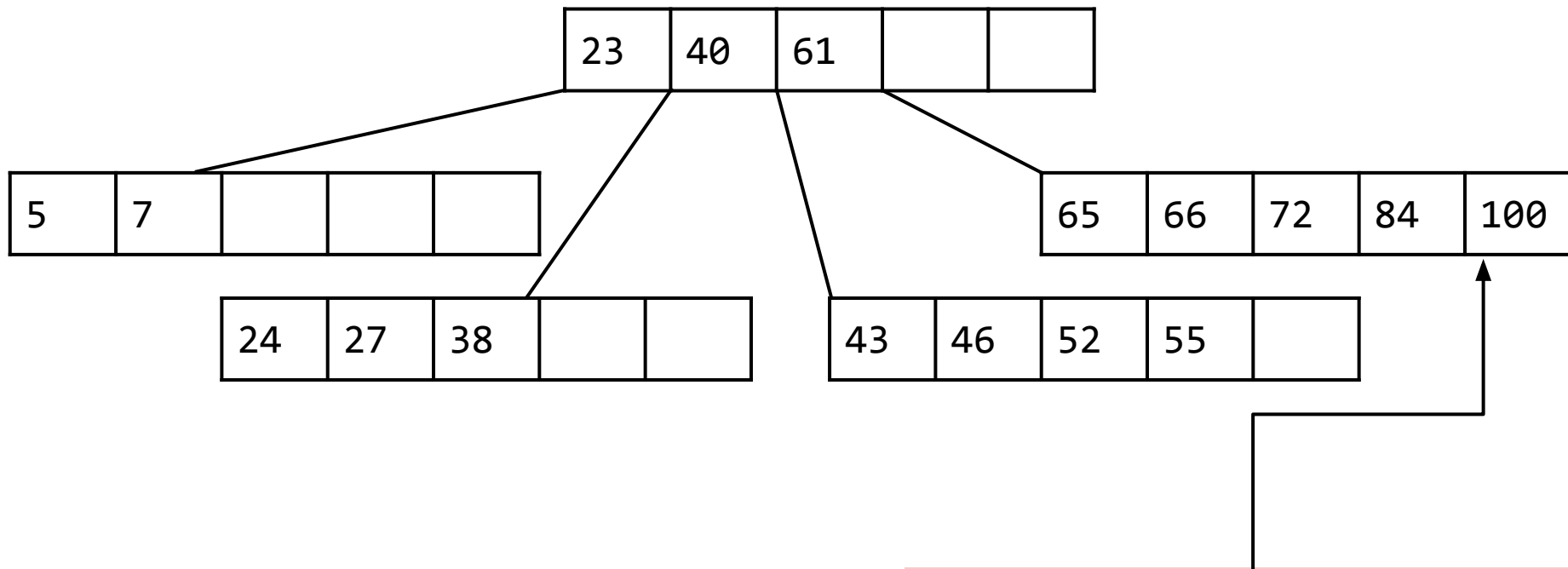
**O bloco precisa ser dividido...**

Inserindo 38



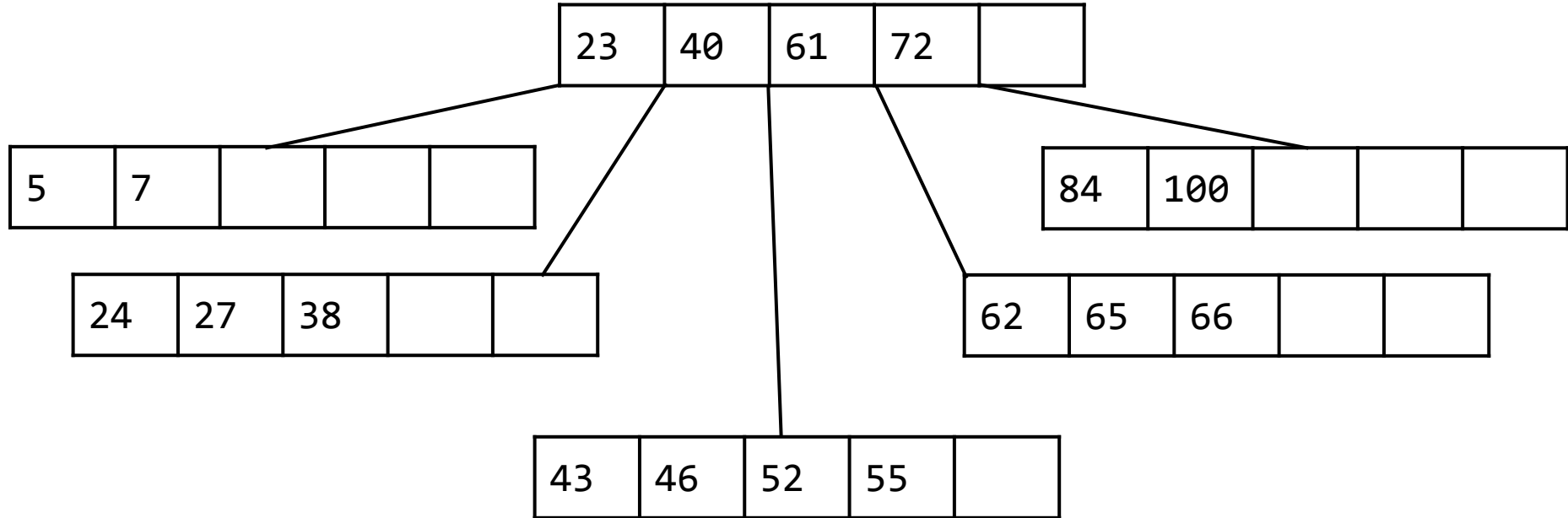


Inserindo 62

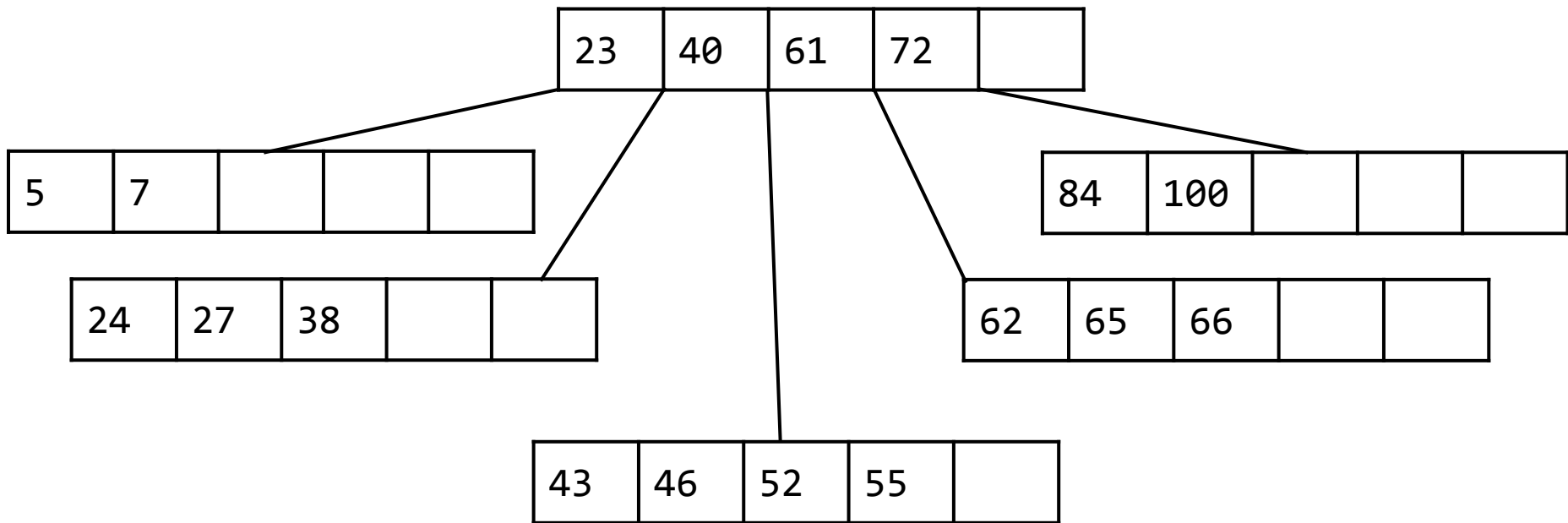


**O bloco precisa ser dividido...**

Inserindo 62

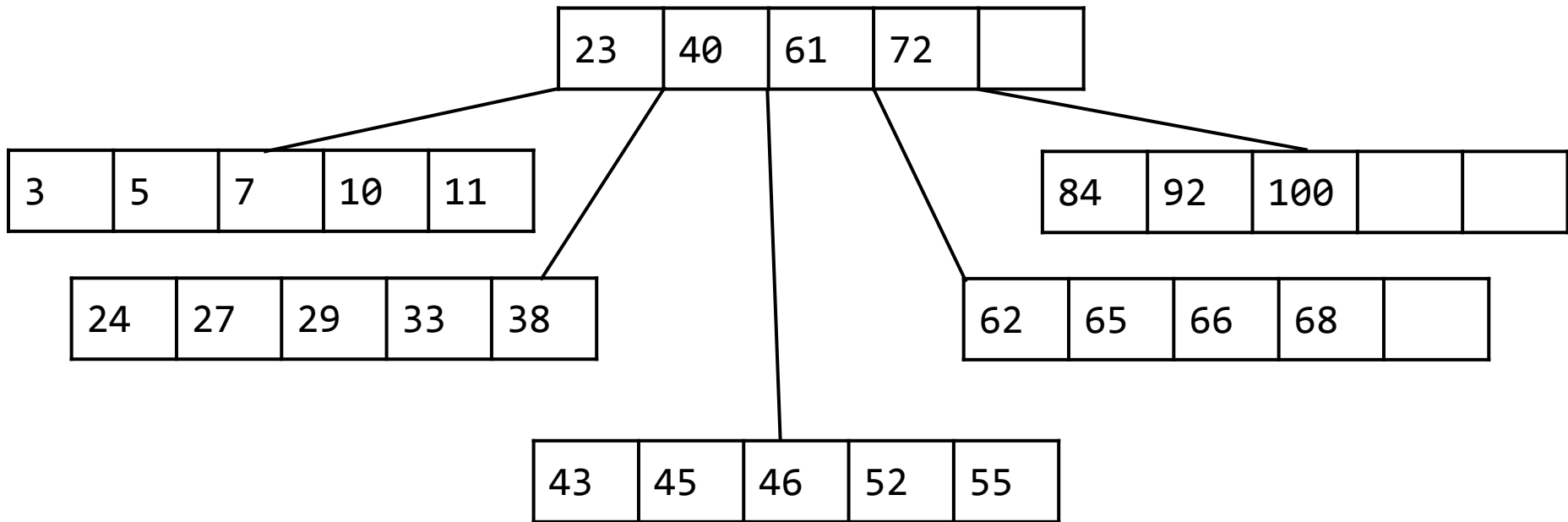


Inserindo 45, 33, 68, 10, 29, 3, 92 e 11

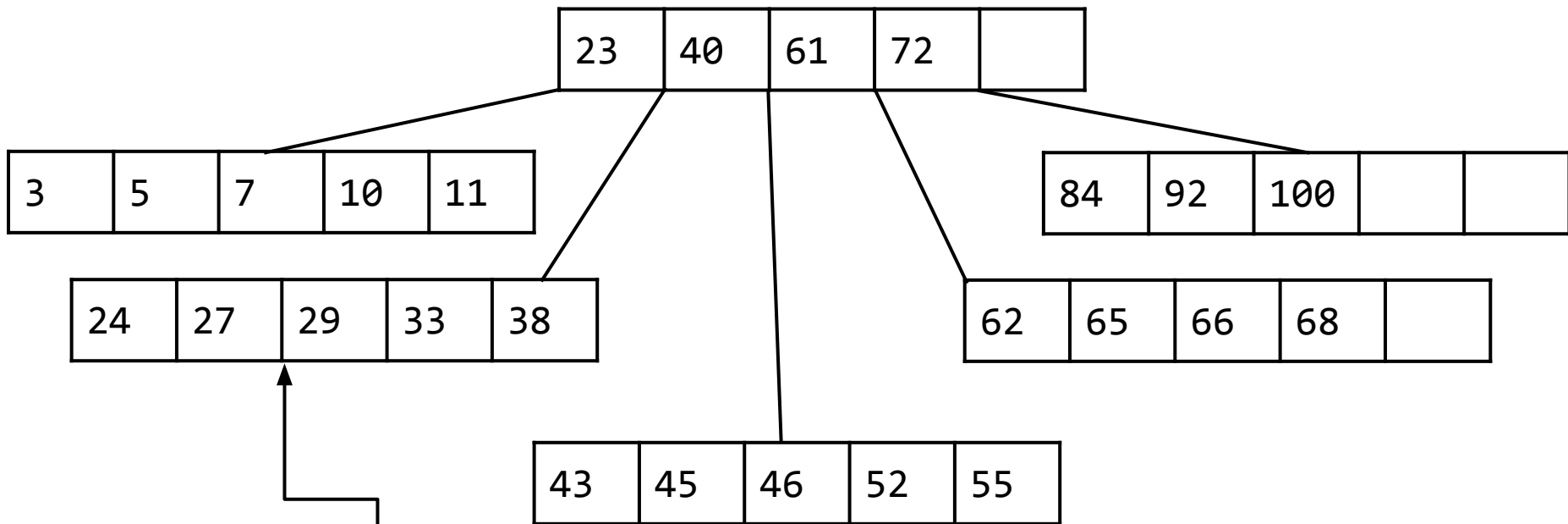


Inserções normais, ...

Inserindo 45, 33, 68, 10, 29, 3, 92 e 11

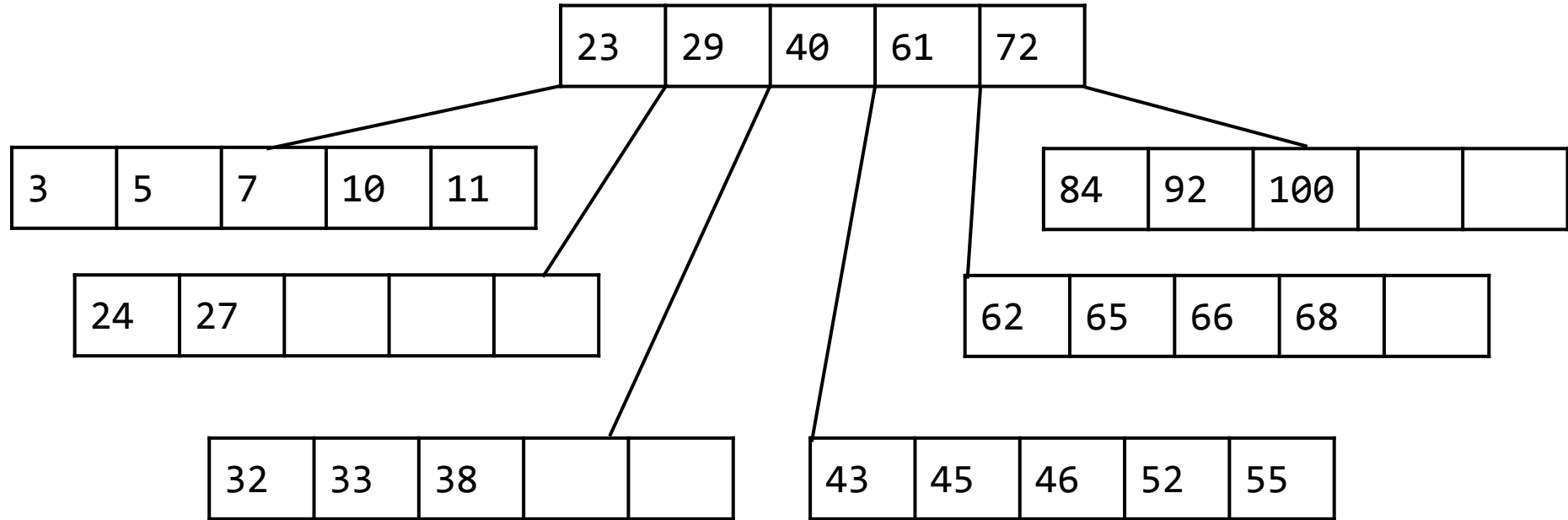


Inserindo 32

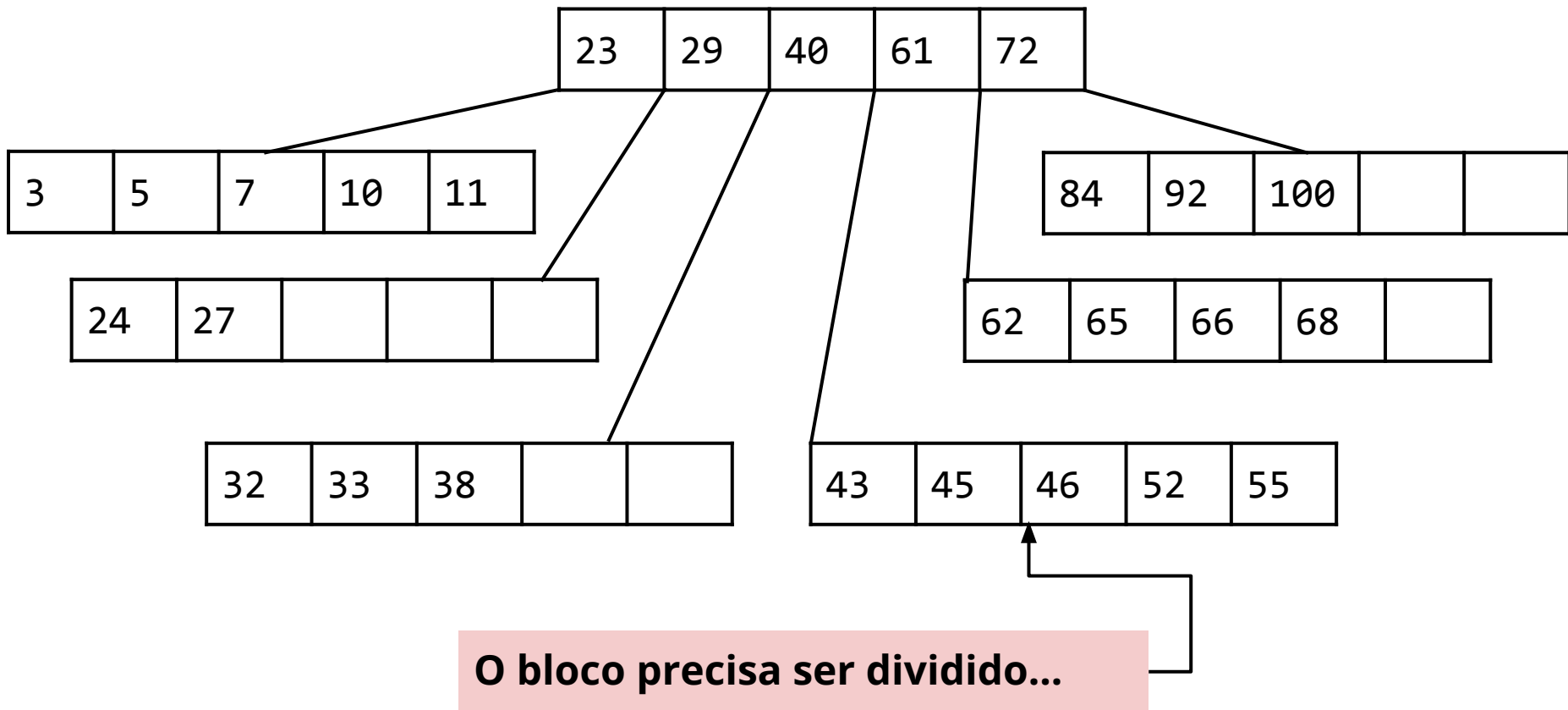


**O bloco precisa ser dividido...**

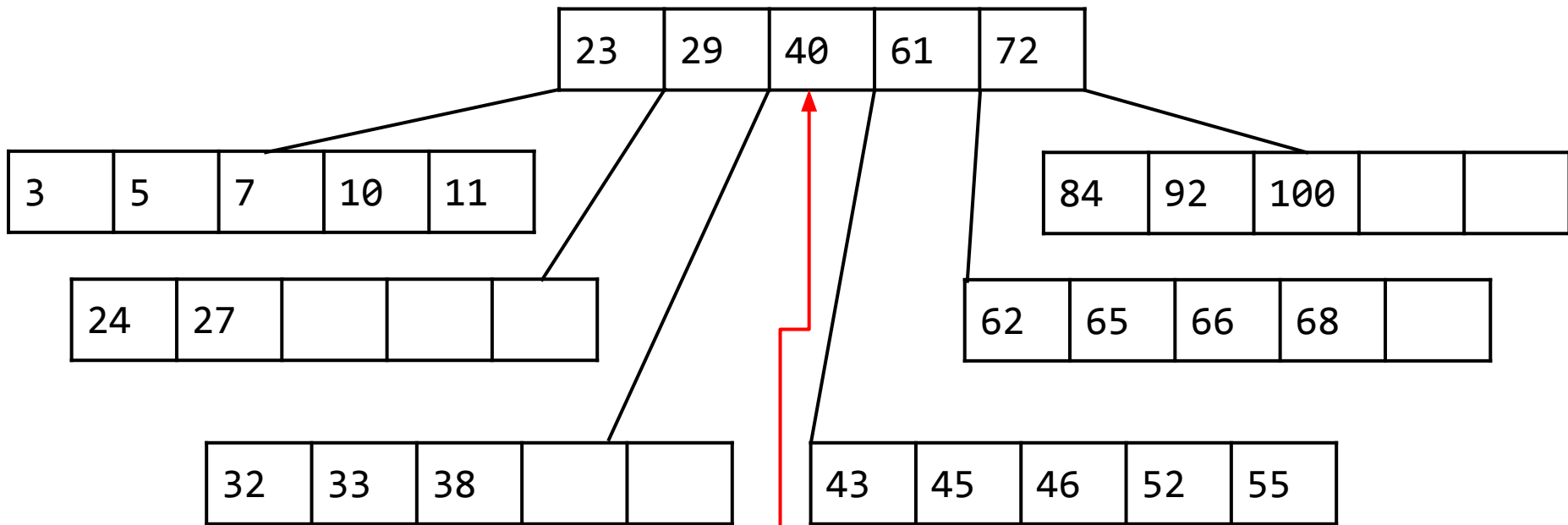
Inserindo 32



Inserindo 53



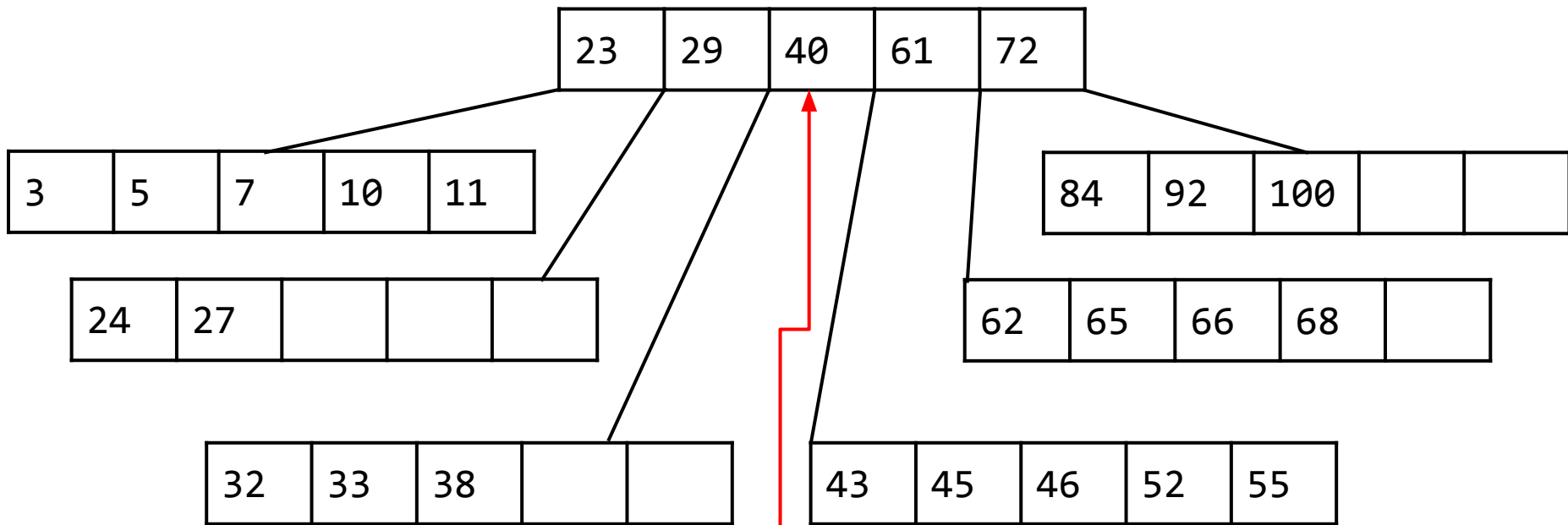
Inserindo 53



**mas o bloco superior (que é a raiz atual) precisa ser dividido antes...**

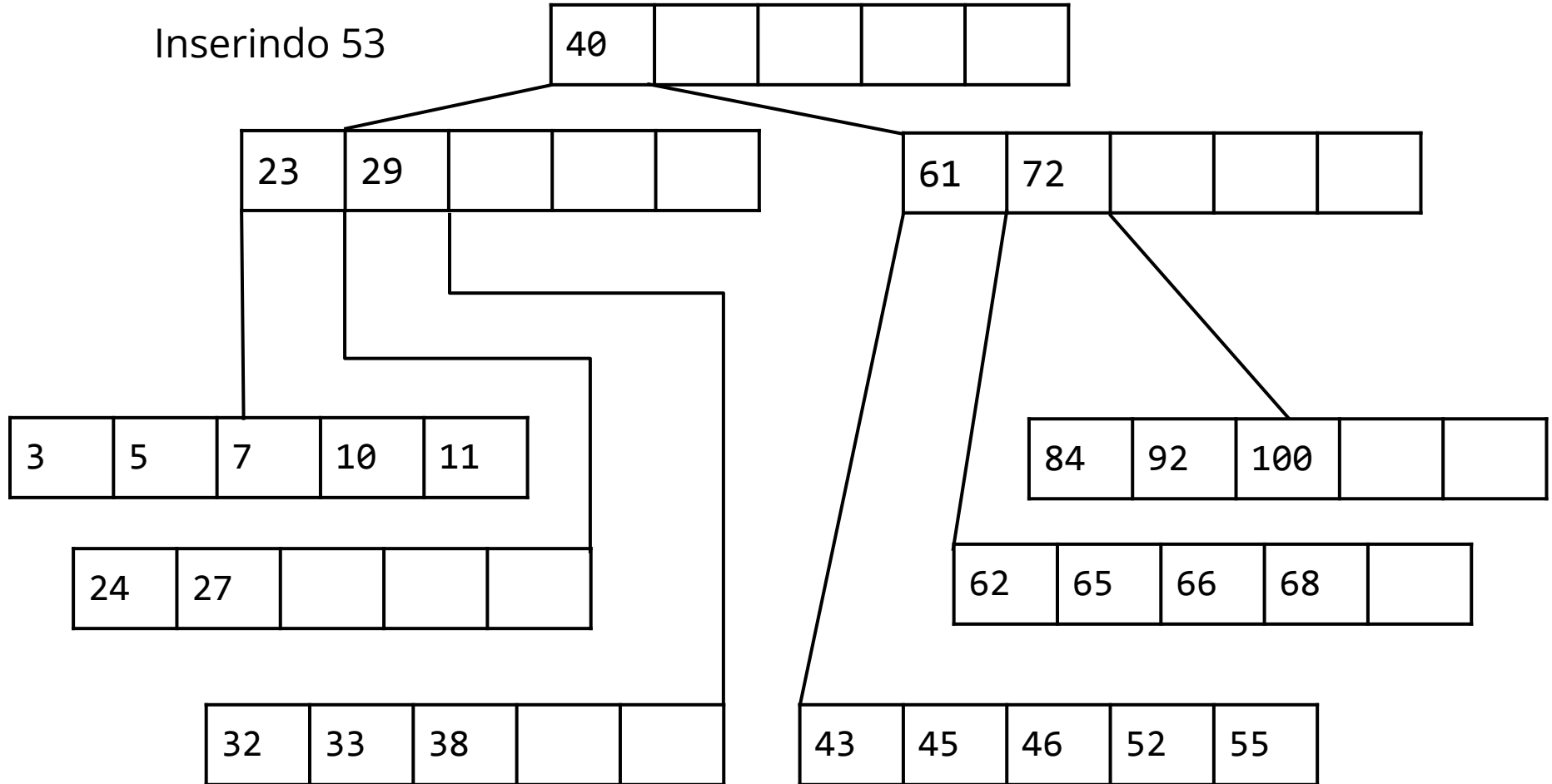


Inserindo 53

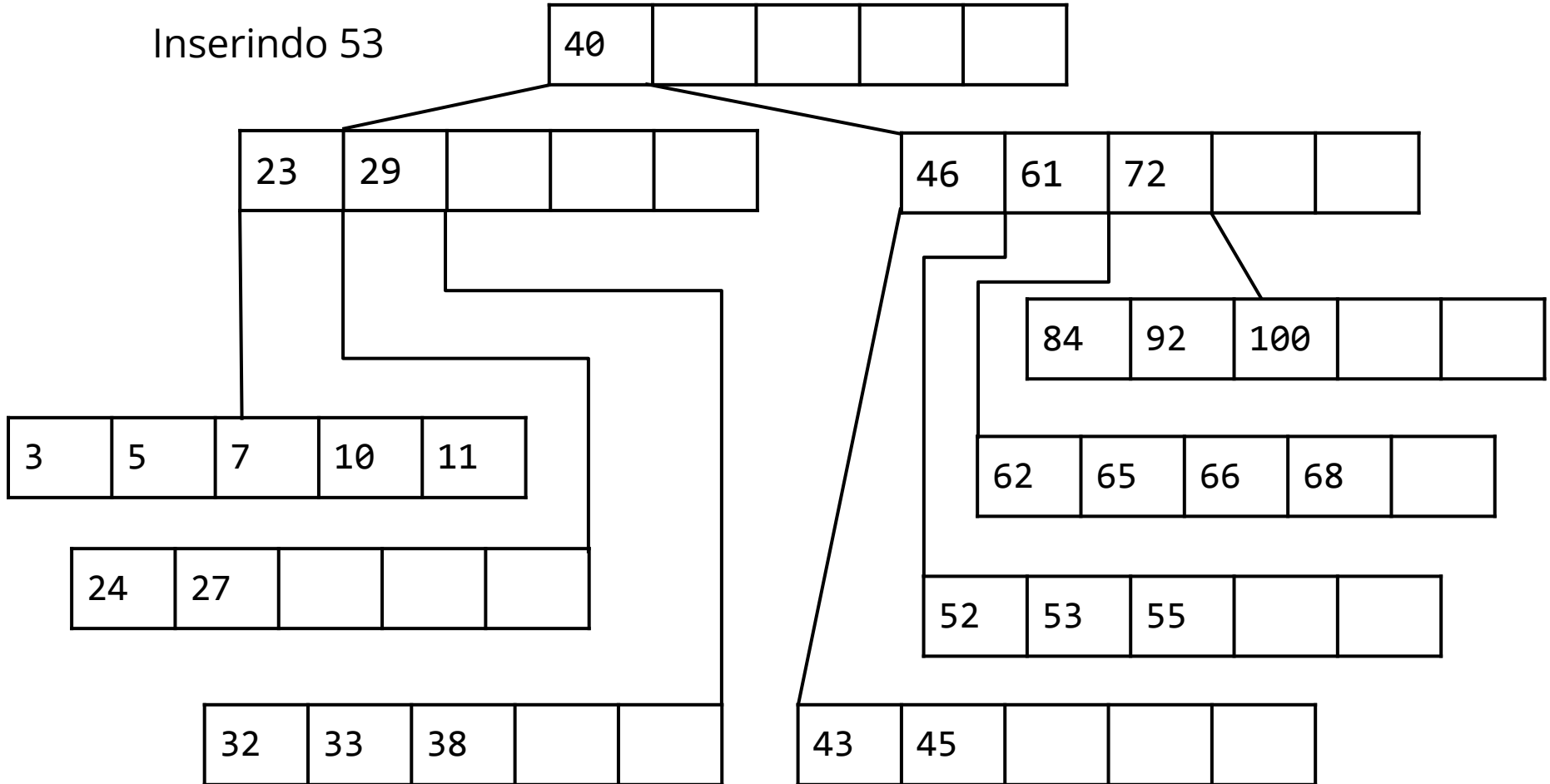


**mas o bloco superior (que é a raiz atual) precisa ser dividido antes...**

Inserindo 53



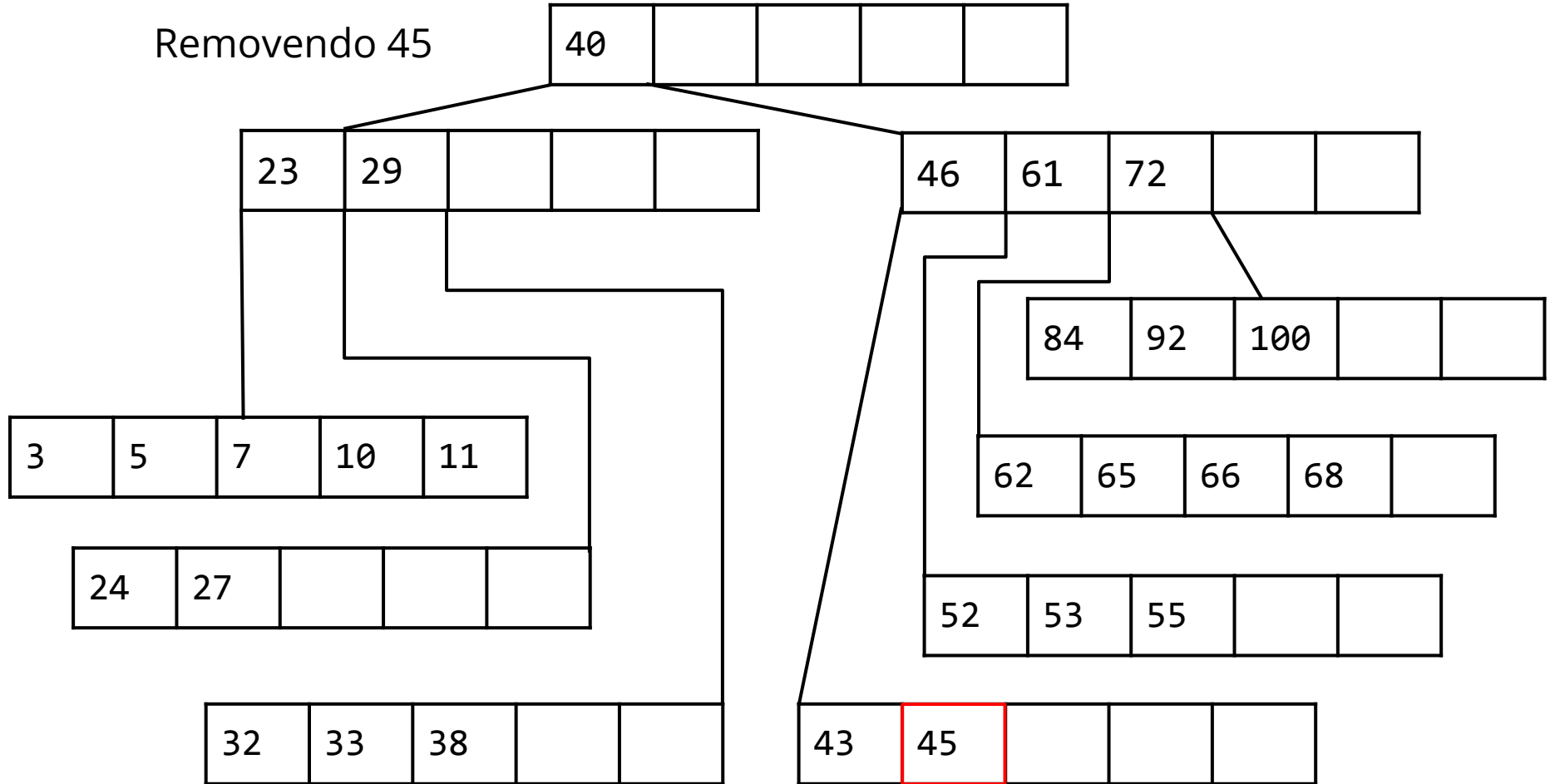
Inserindo 53



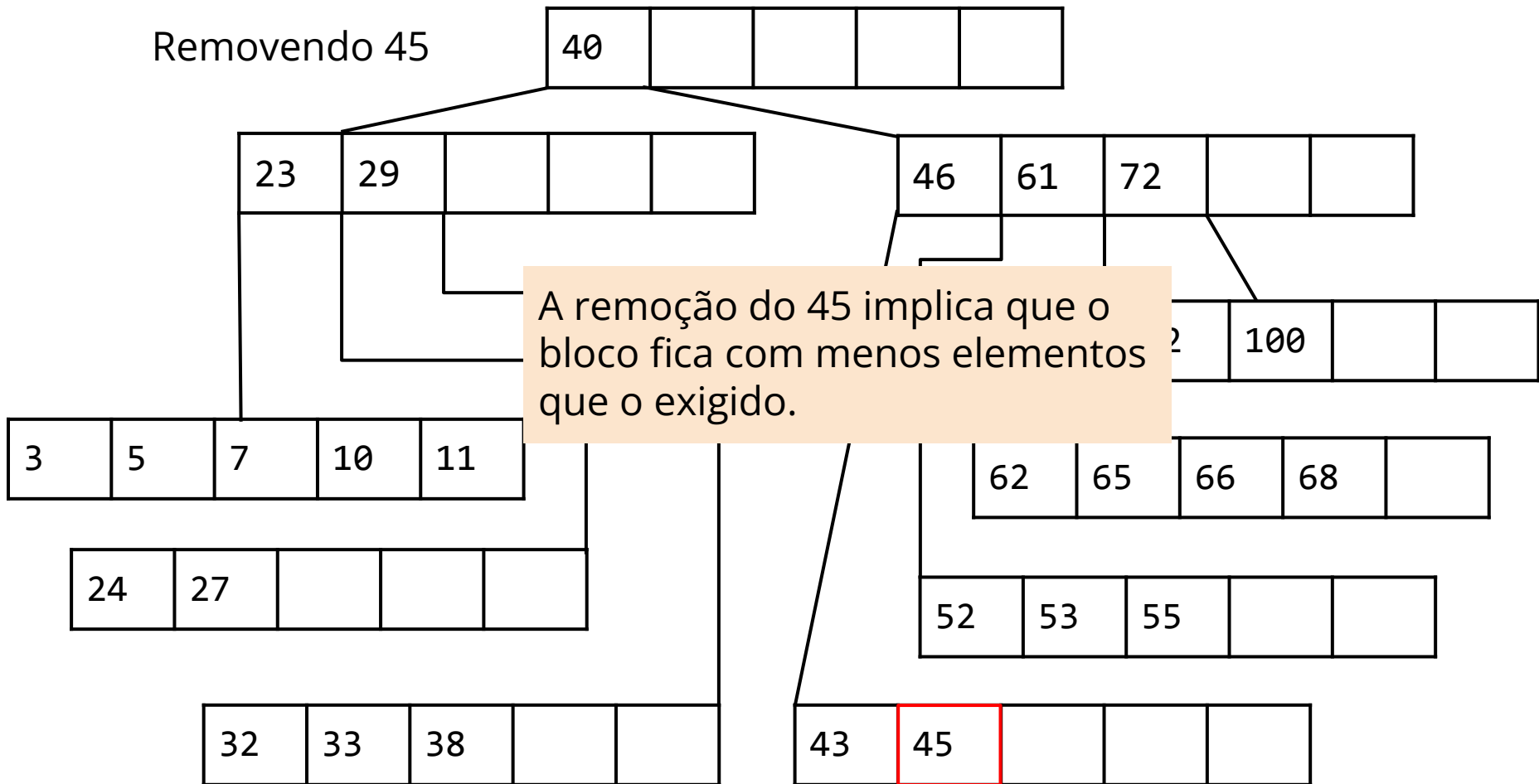
# Valores a serem removidos

45      33      27      55      38      52      53

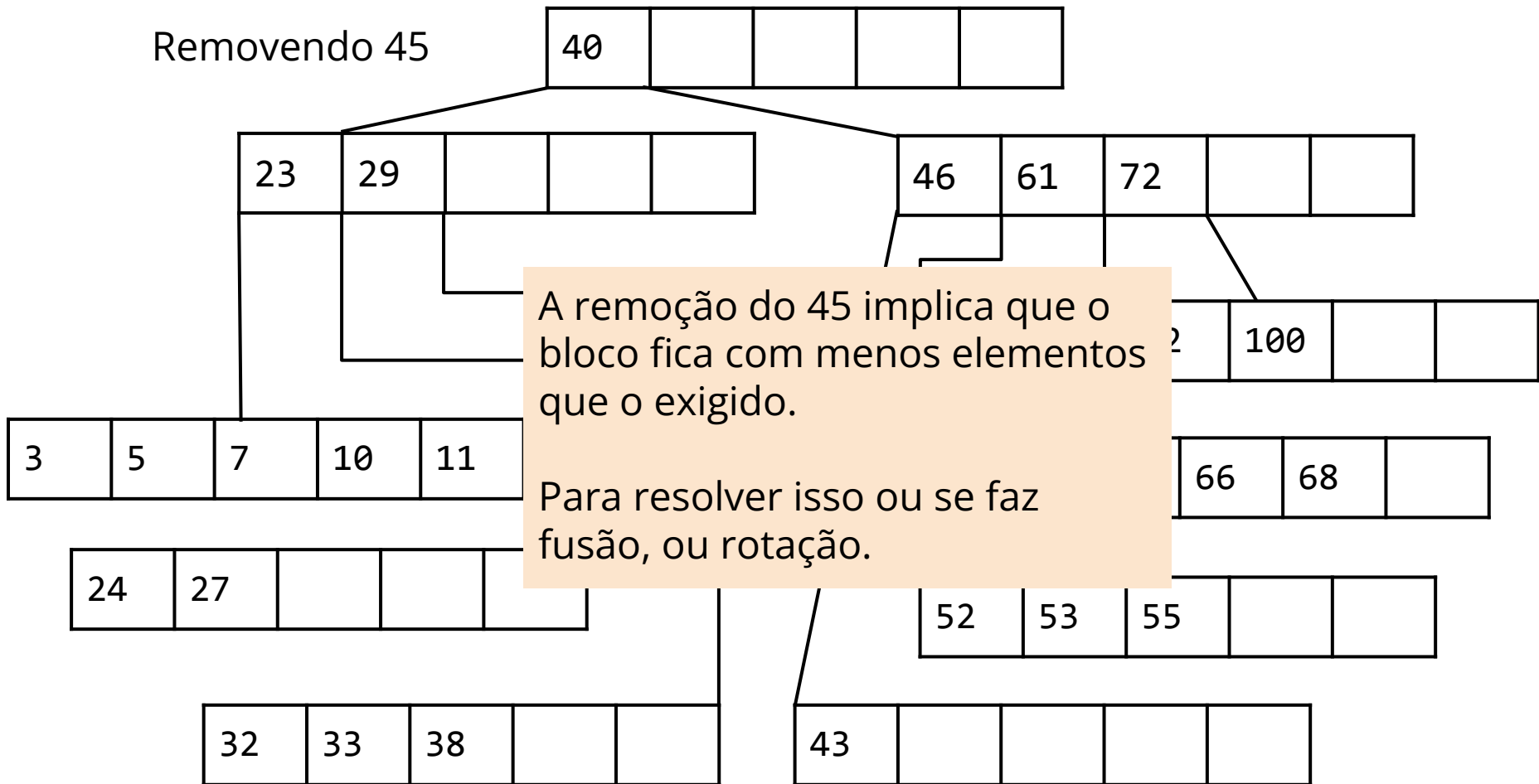
Removendo 45



Removendo 45



Removendo 45



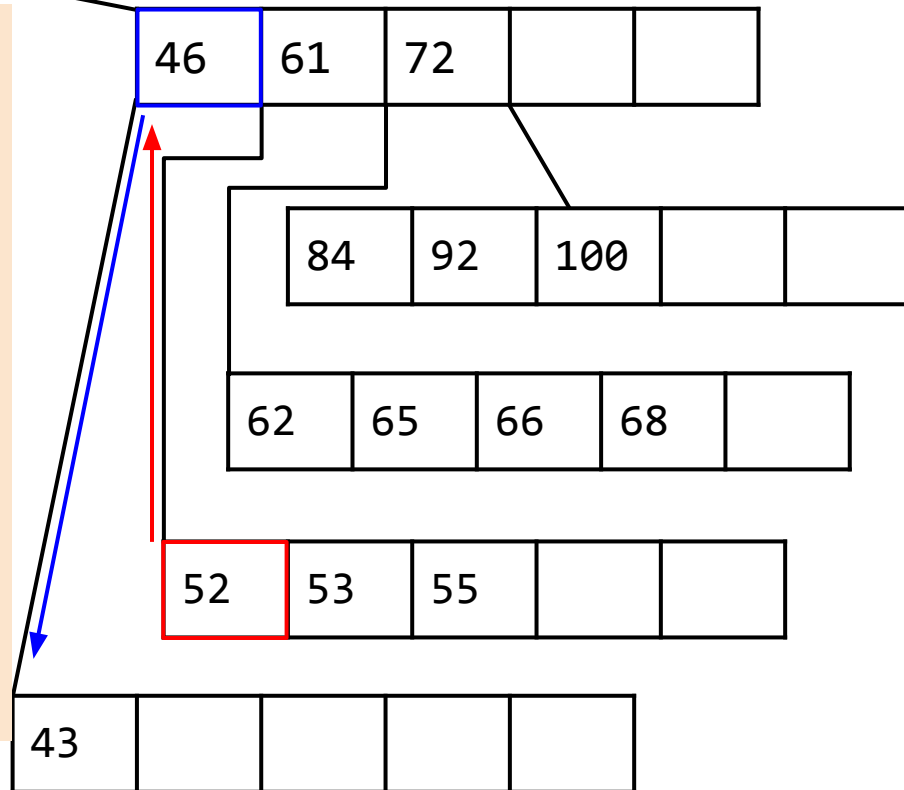
Removendo 45



A remoção do 45 implica que o bloco fica com menos elementos que o exigido.

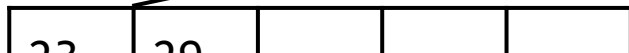
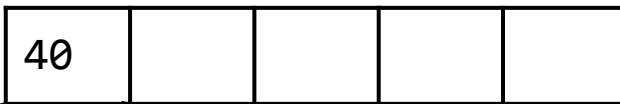
Para resolver isso ou se faz fusão, ou rotação. A fusão é uma tarefa mais complexa, então iremos fazer rotação...

No caso, emprestamos um valor do irmão (a preferência é pelo da esquerda, o que não é o caso).

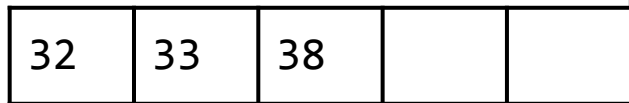
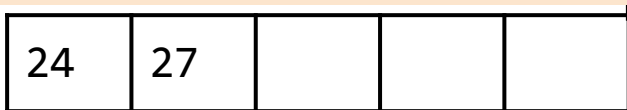
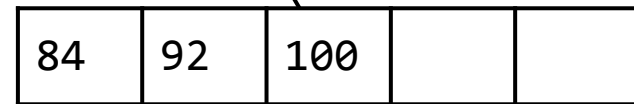
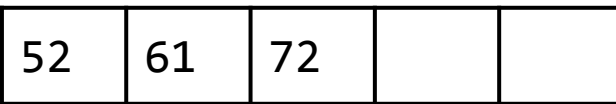




Removendo 45

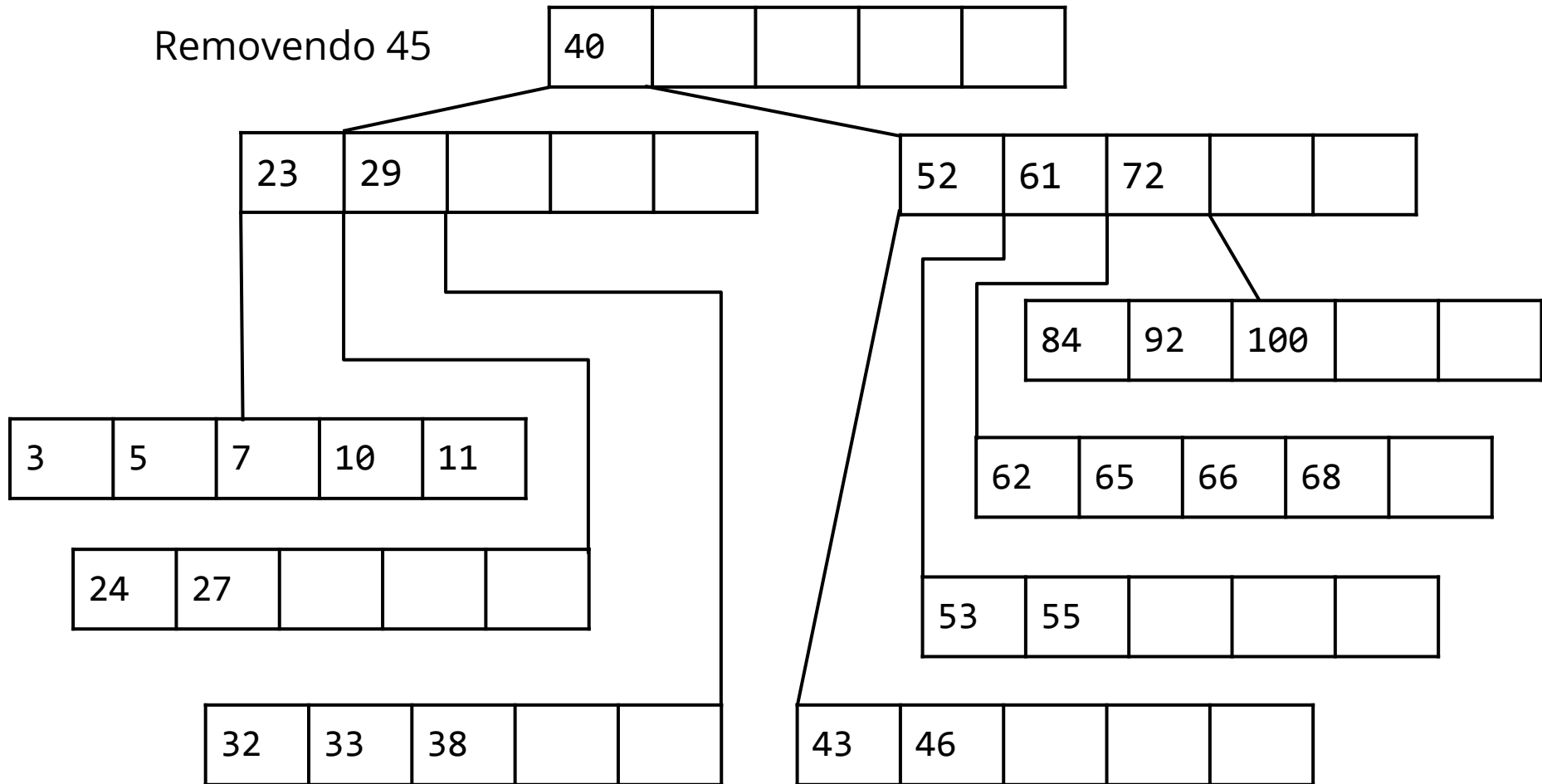


Vejam que o dado com valor 52 subiu para o bloco "pai", fazendo com que o 46 descesse ao nó em que houve a remoção...



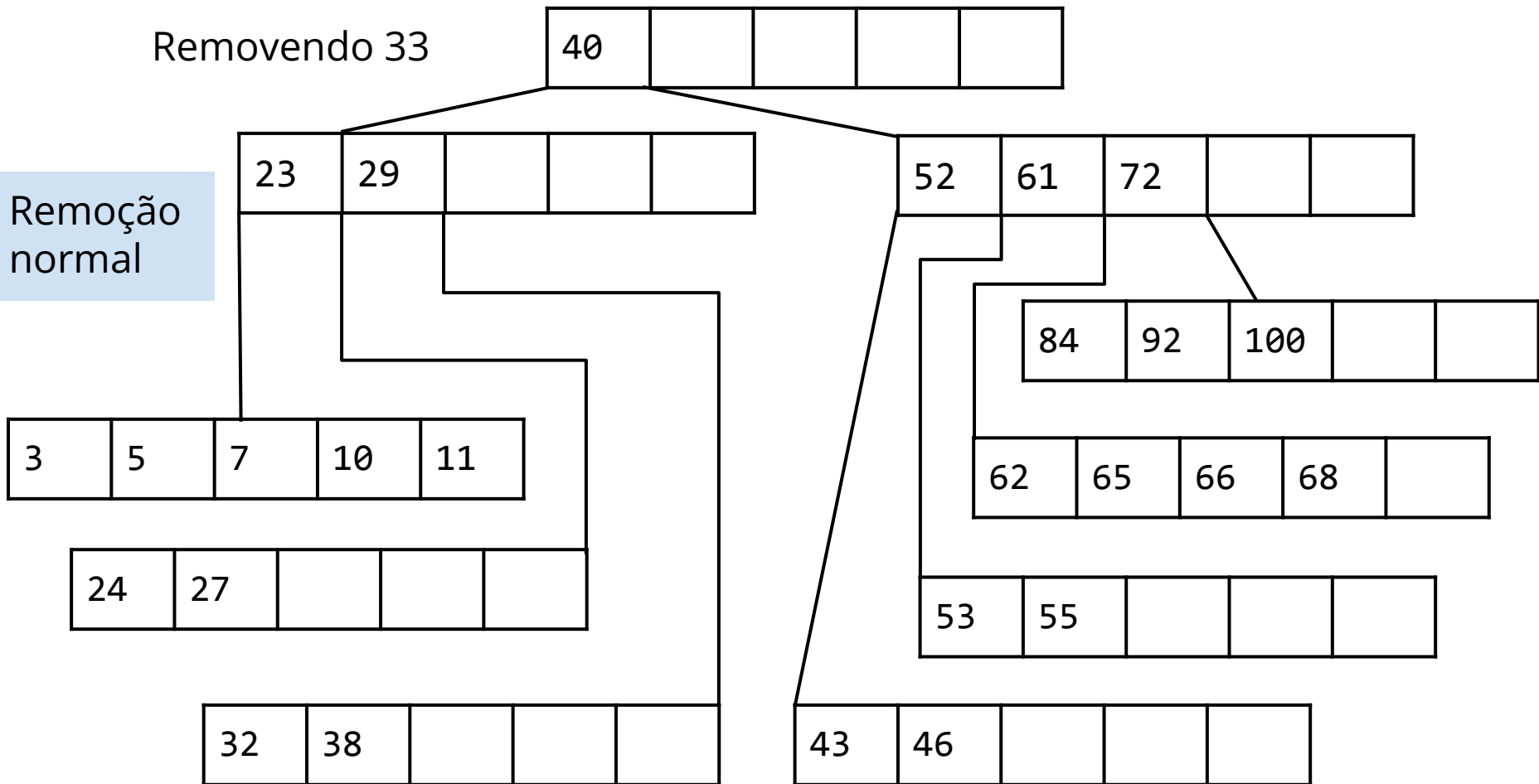
3 As propriedades da árvore foram mantidas.

Removendo 45

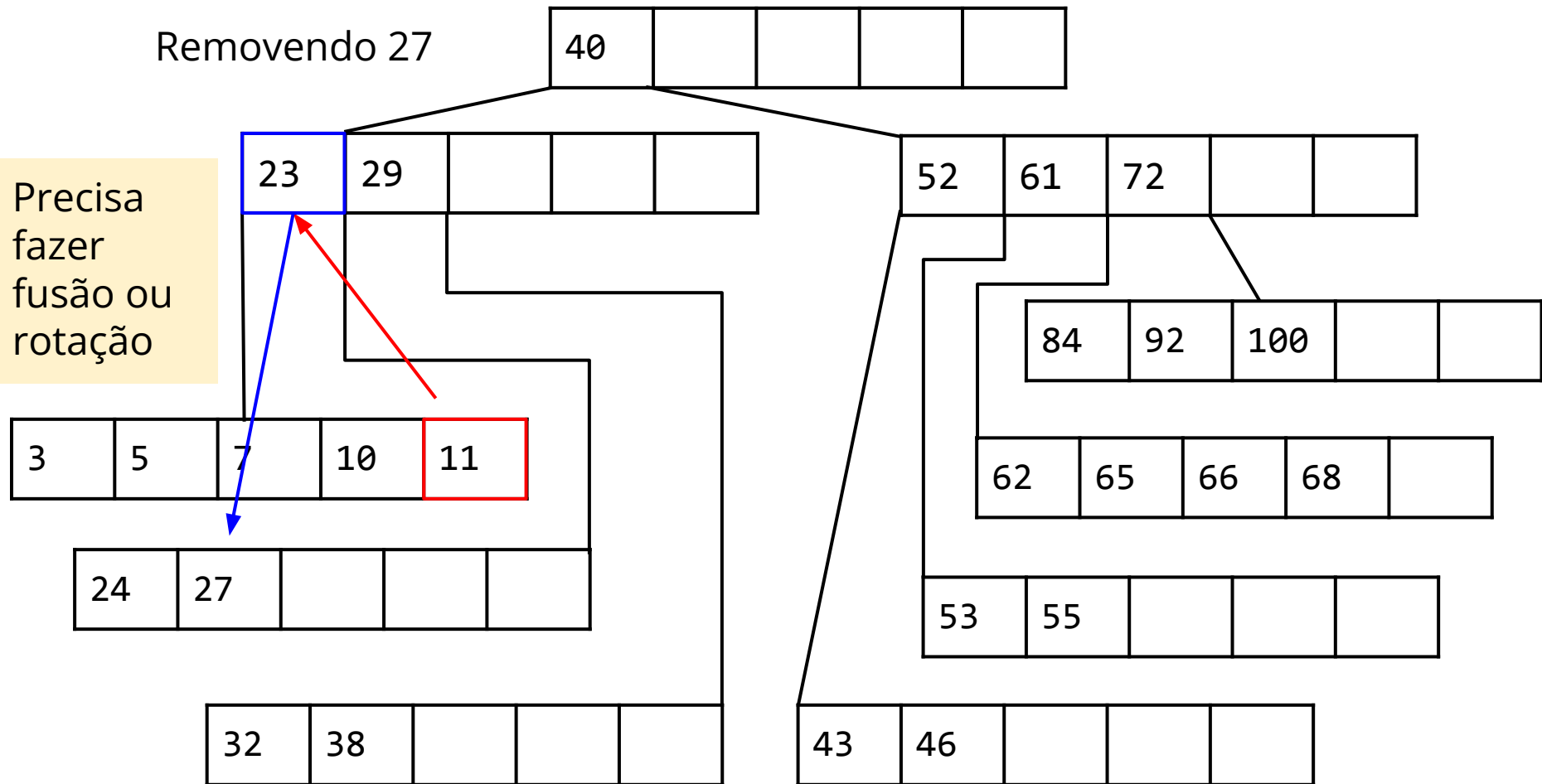


Removendo 33

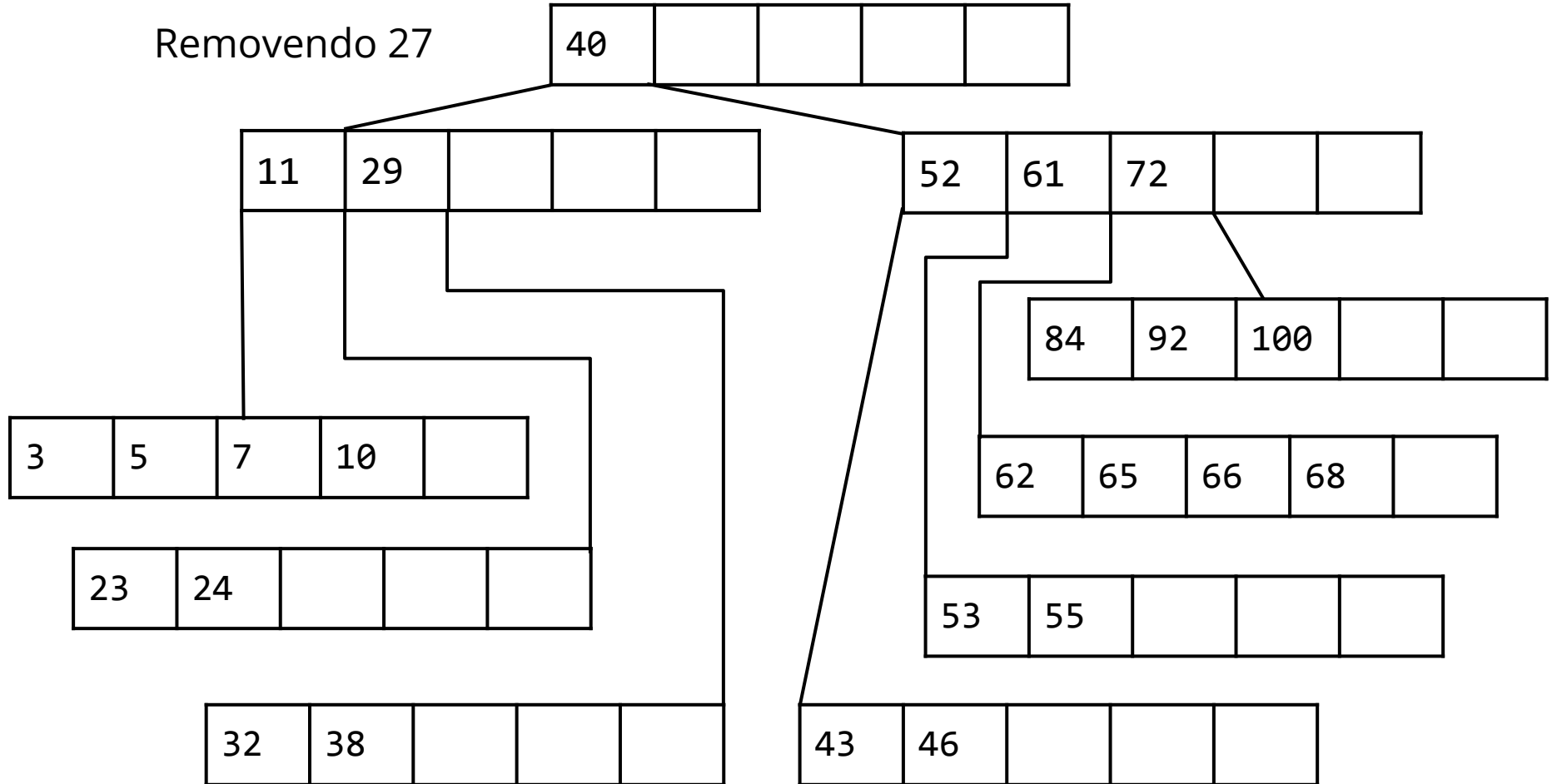
Remoção  
normal



Removendo 27

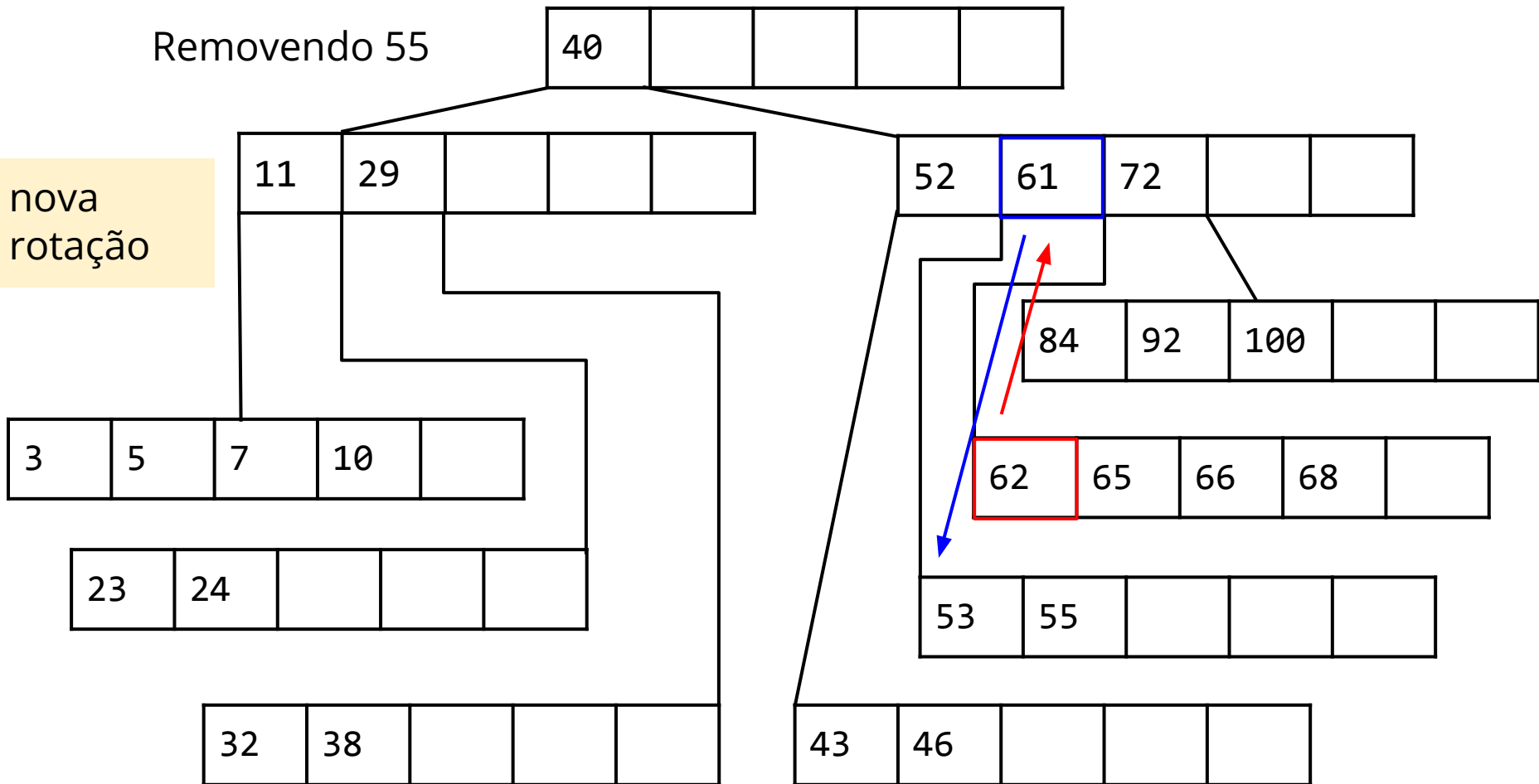


Removendo 27

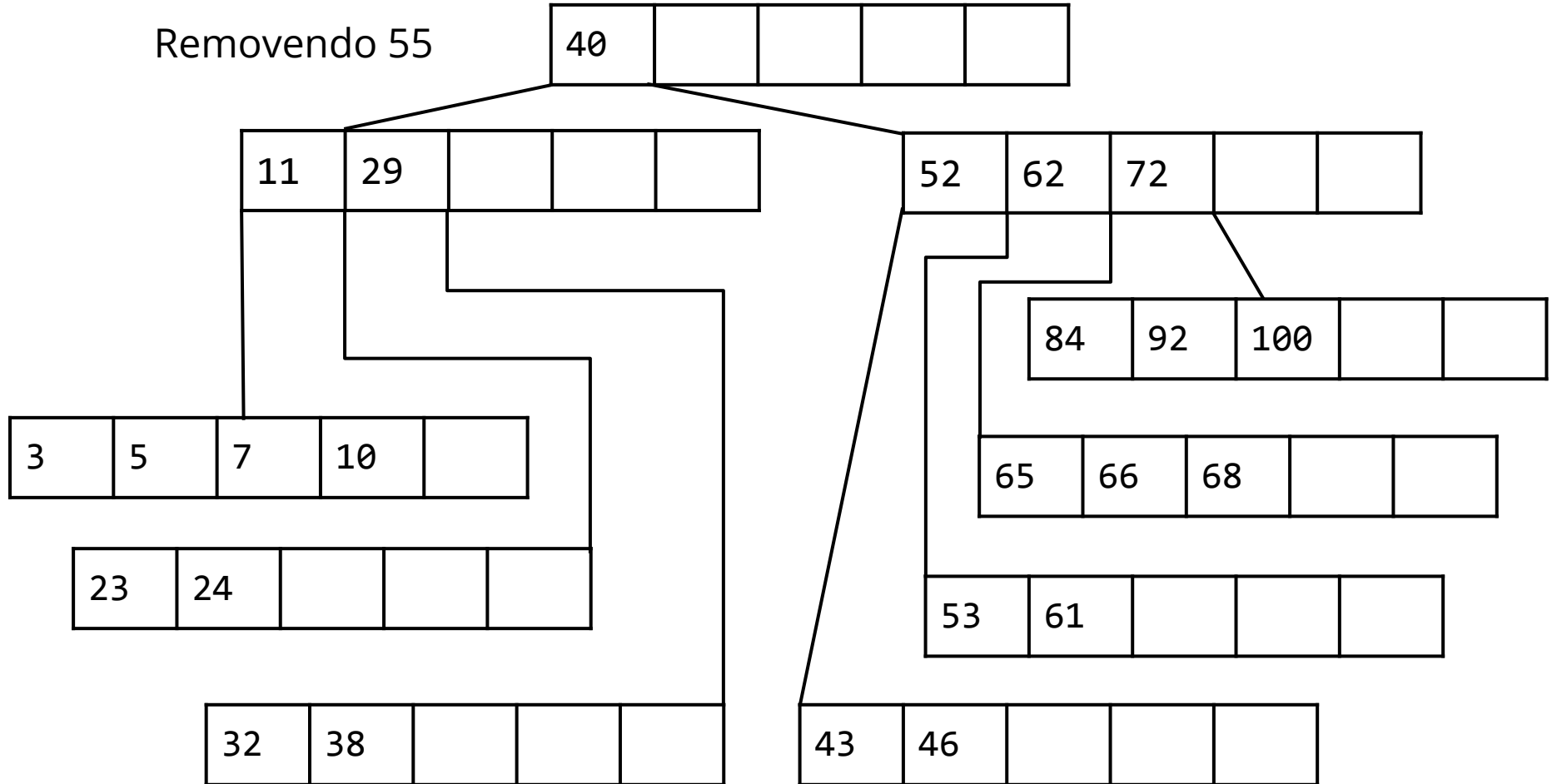


Removendo 55

nova  
rotação

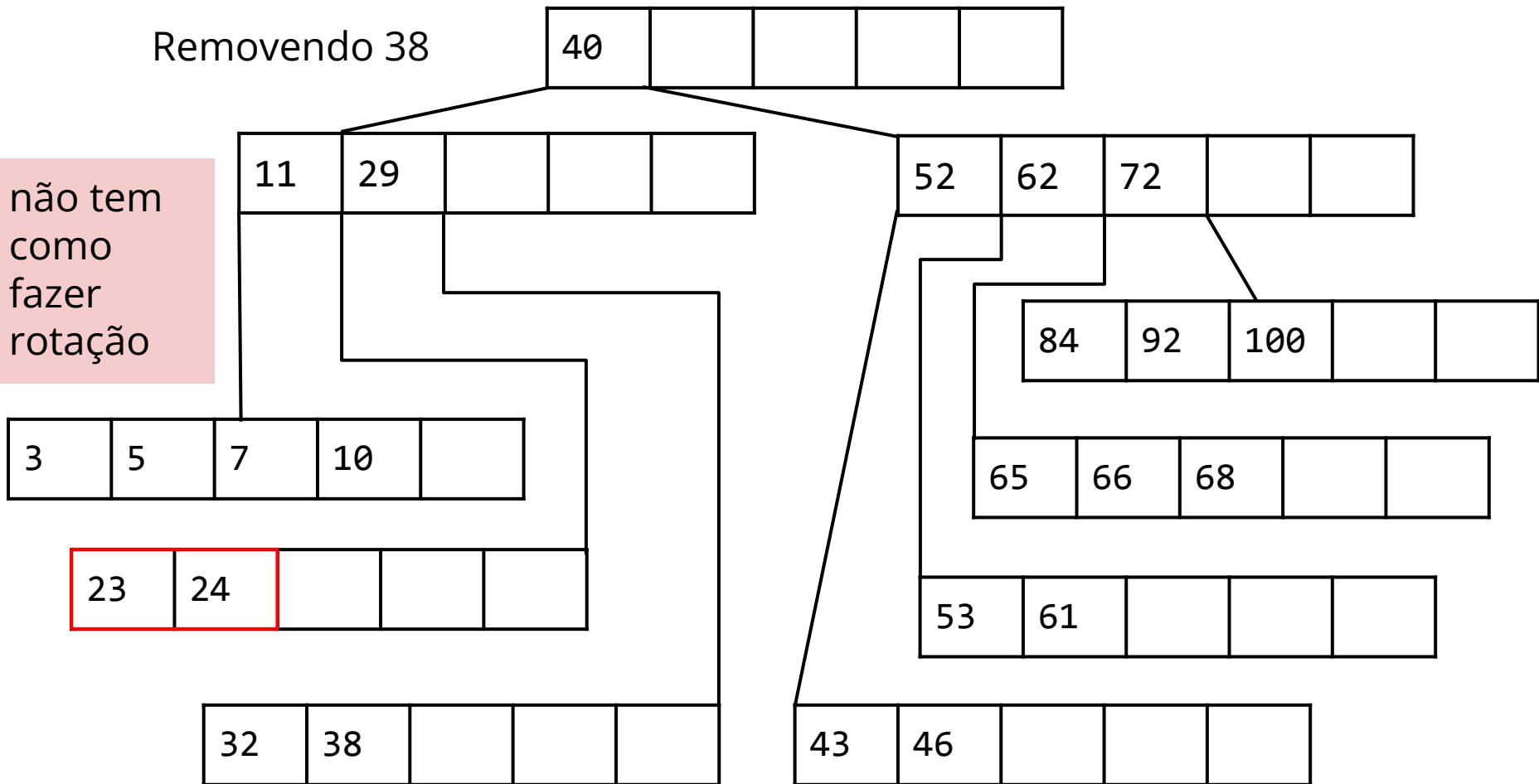


Removendo 55



Removendo 38

não tem  
como  
fazer  
rotação





Removendo 38

40

faz-se  
fusão!

1

3

5

7

23

24

32

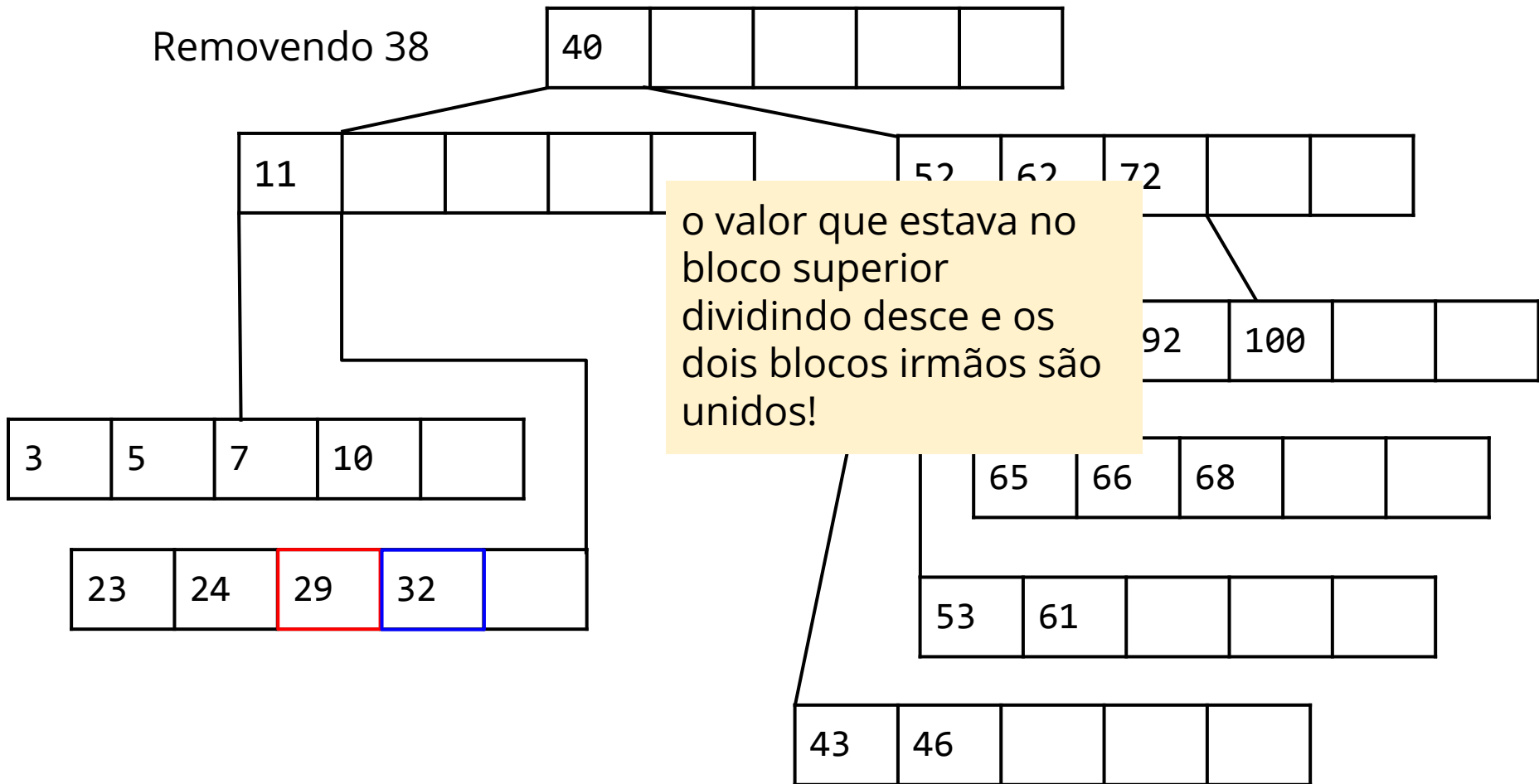
38

43

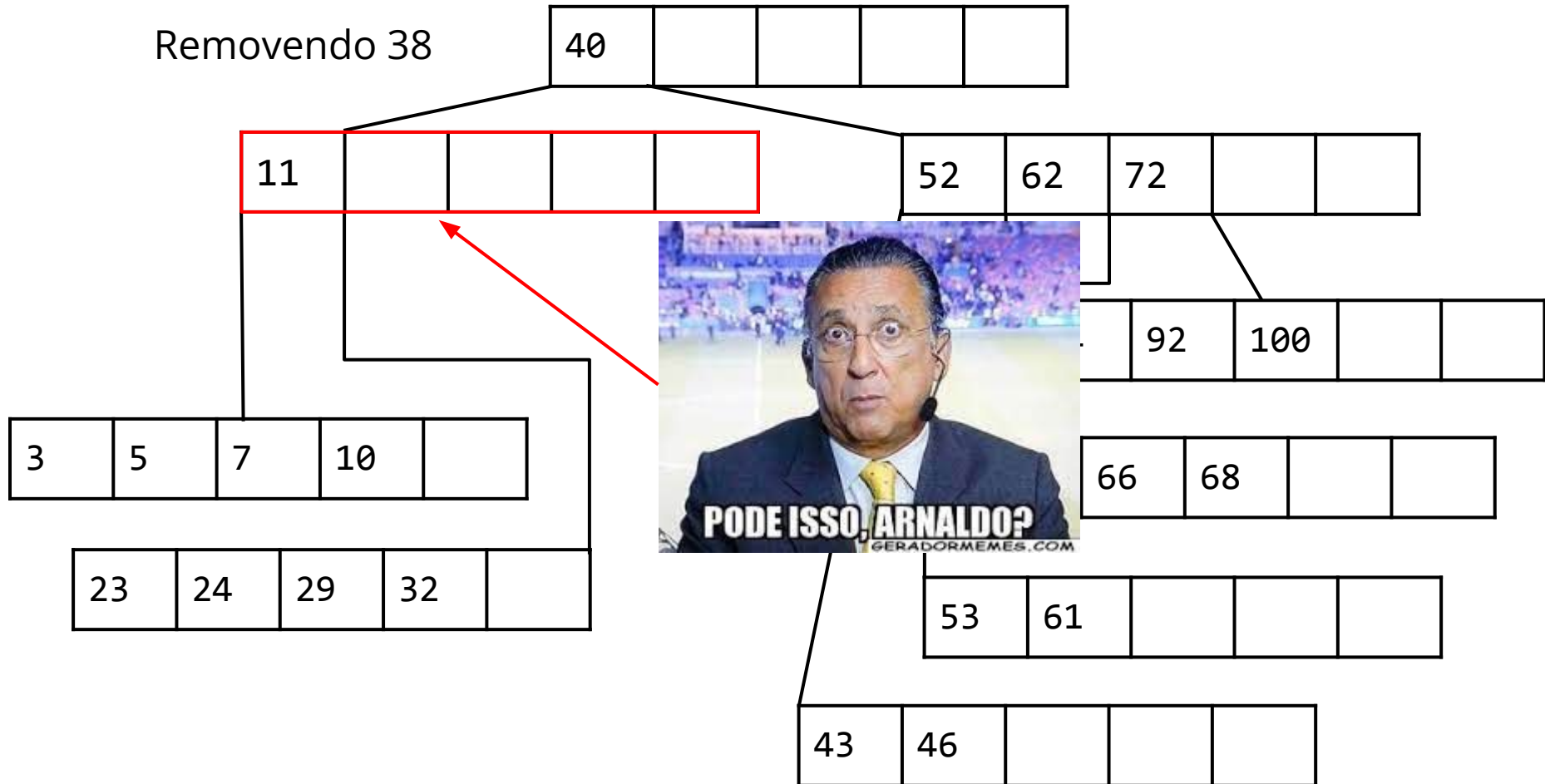
46



Removendo 38



Removendo 38



Removendo 38



Removendo 38

40

11

52

62

72

84

92

100

65

66

68

53

61

43

46

faz-se  
rotação!

3

5

7

10

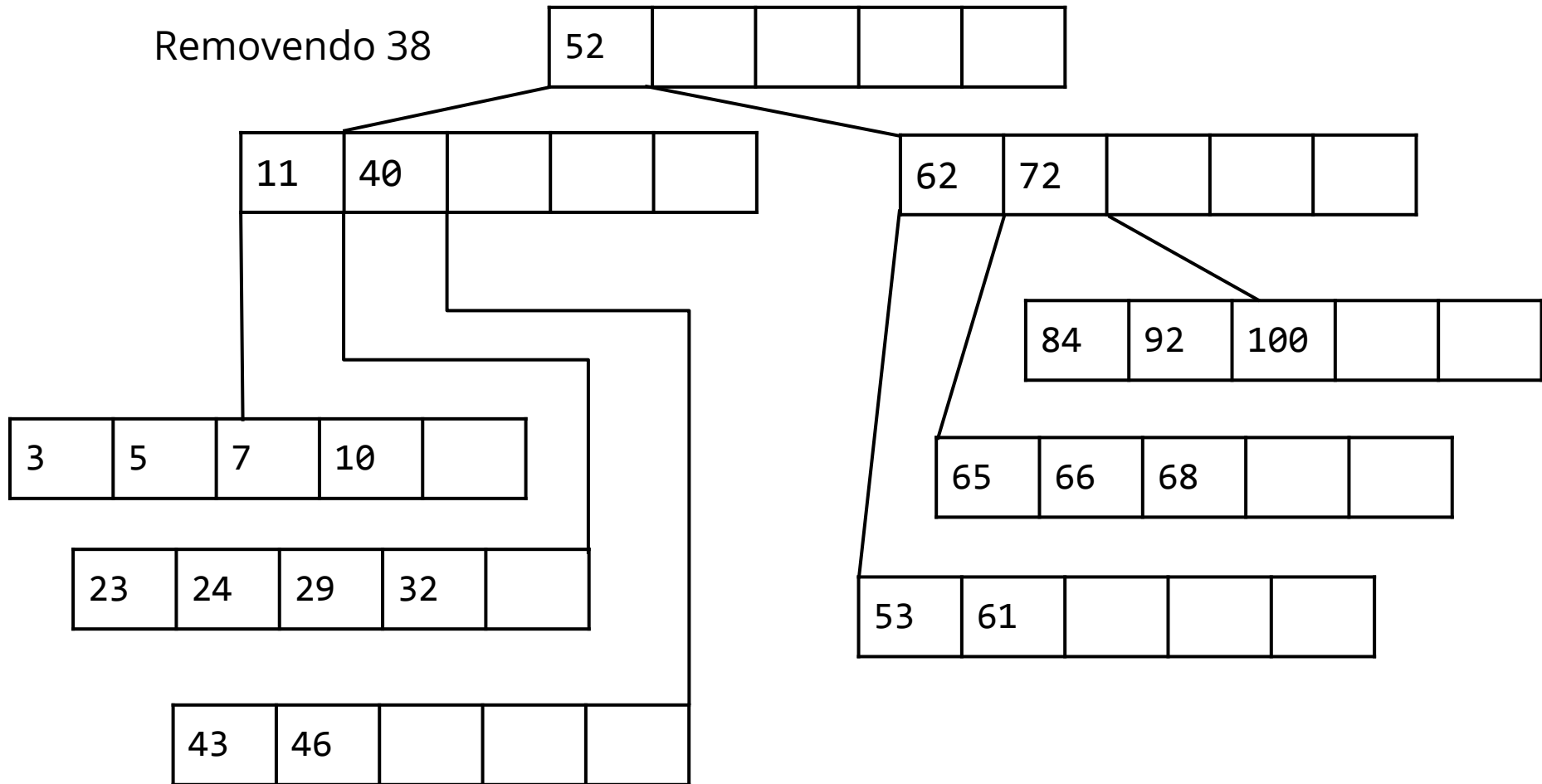
23

24

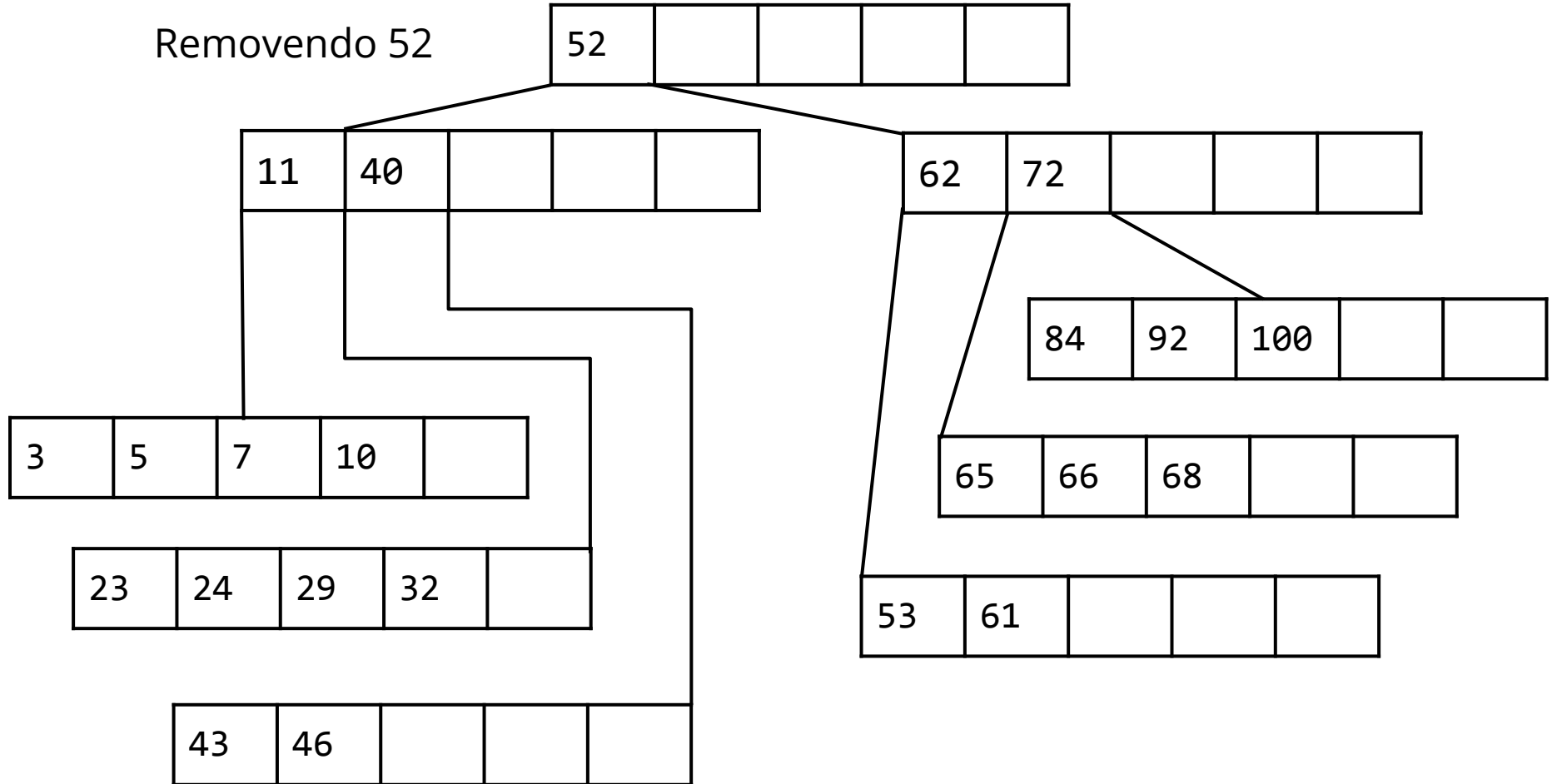
29

32

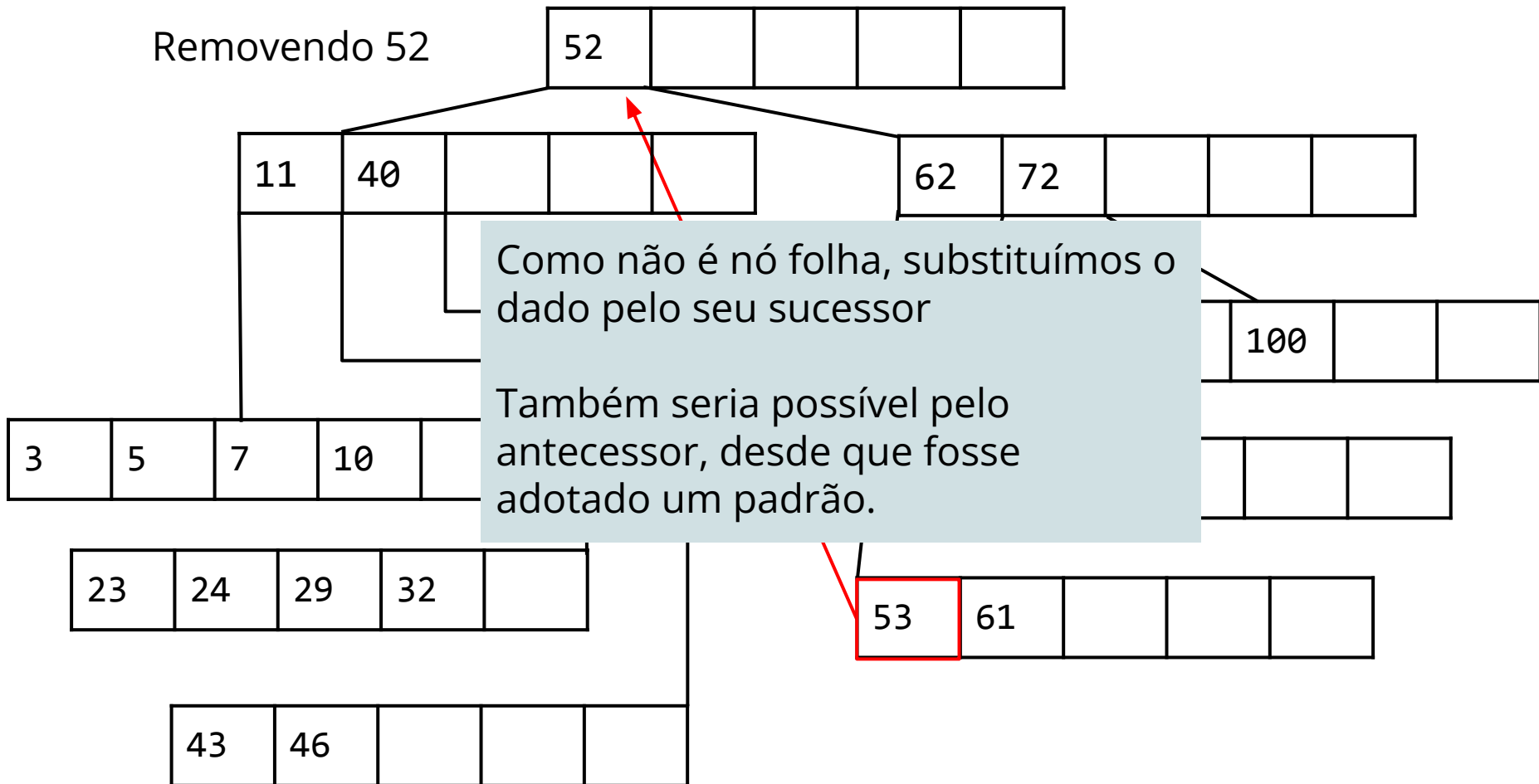
Removendo 38



Removendo 52

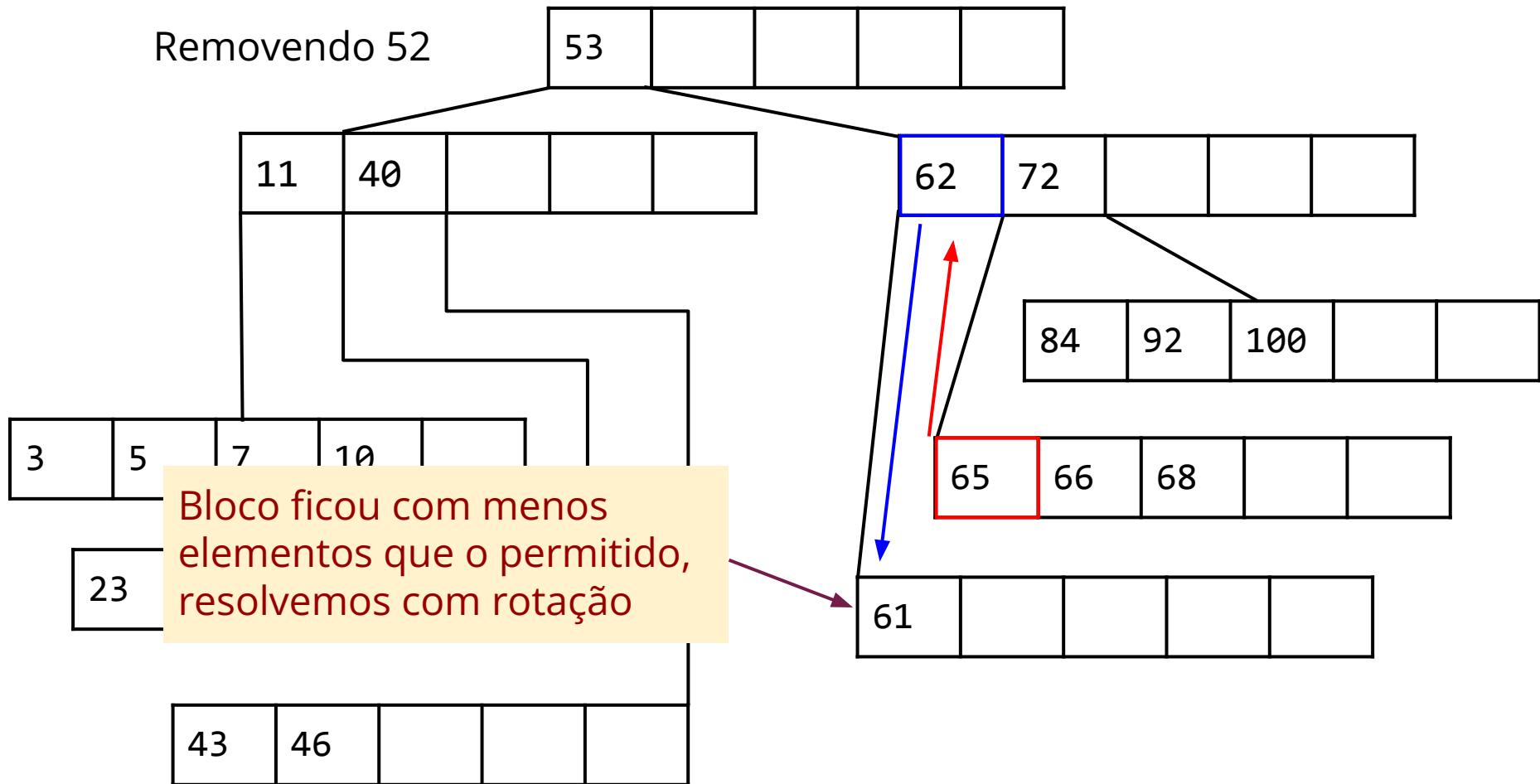


Removendo 52

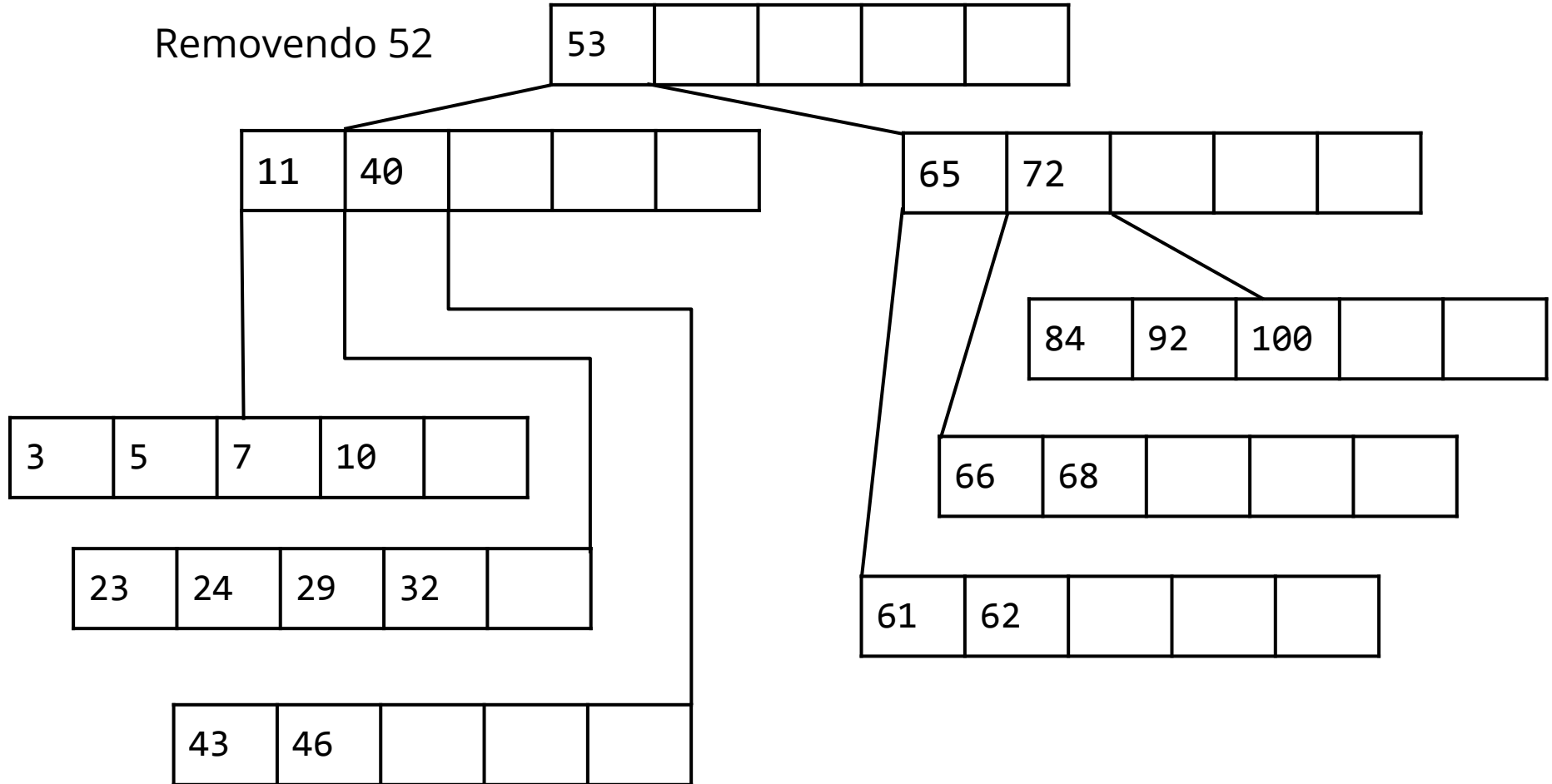




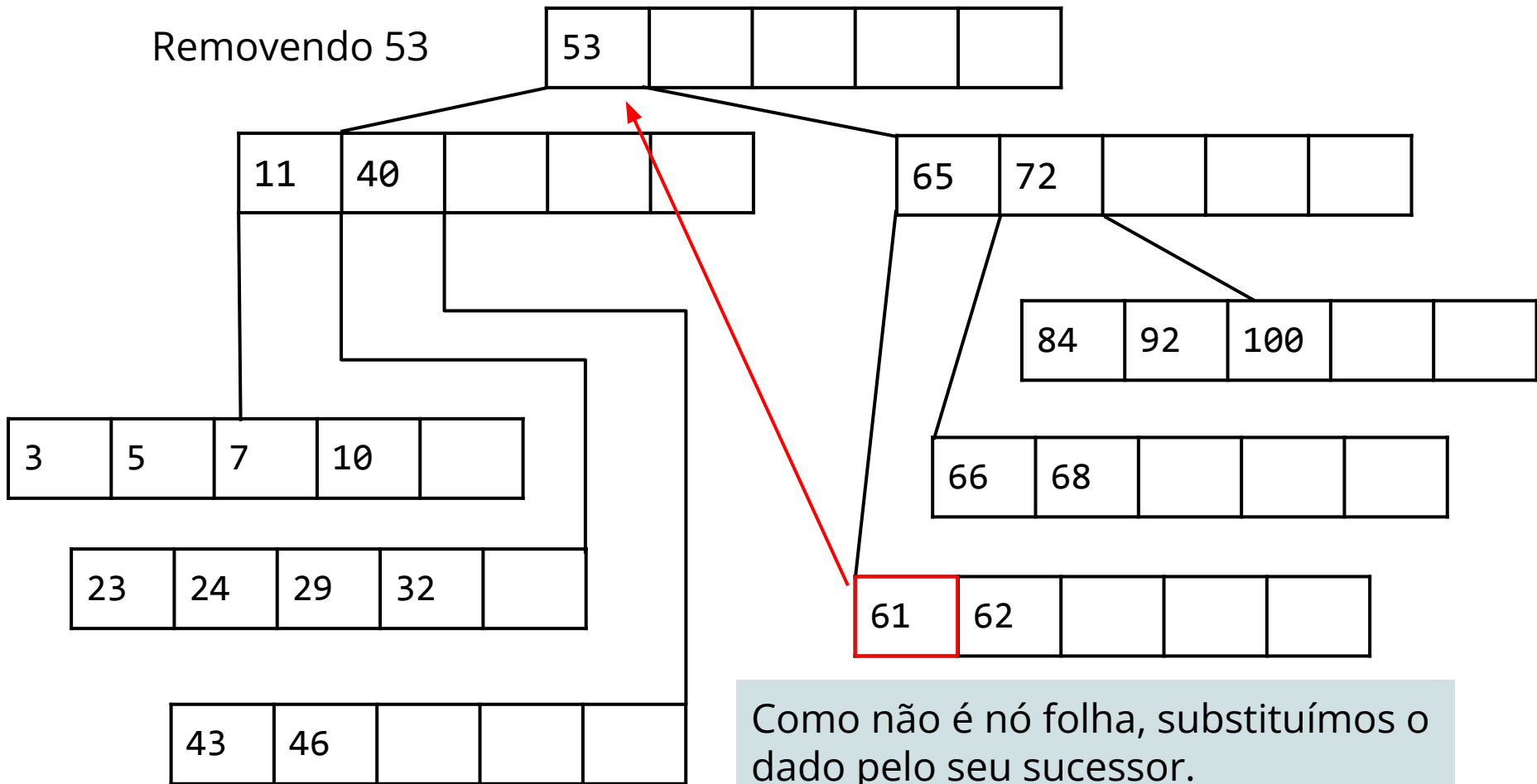
Removendo 52



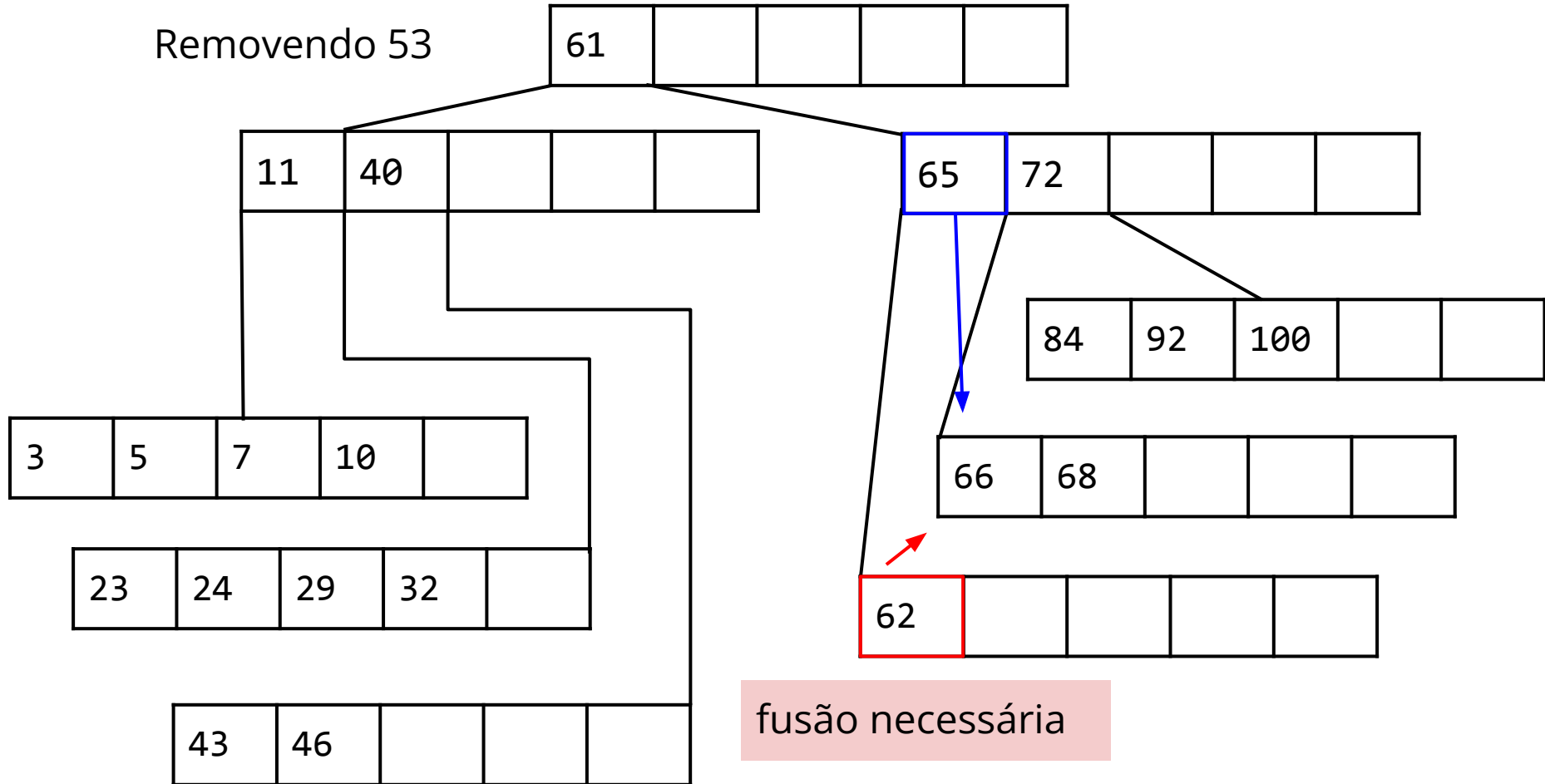
Removendo 52



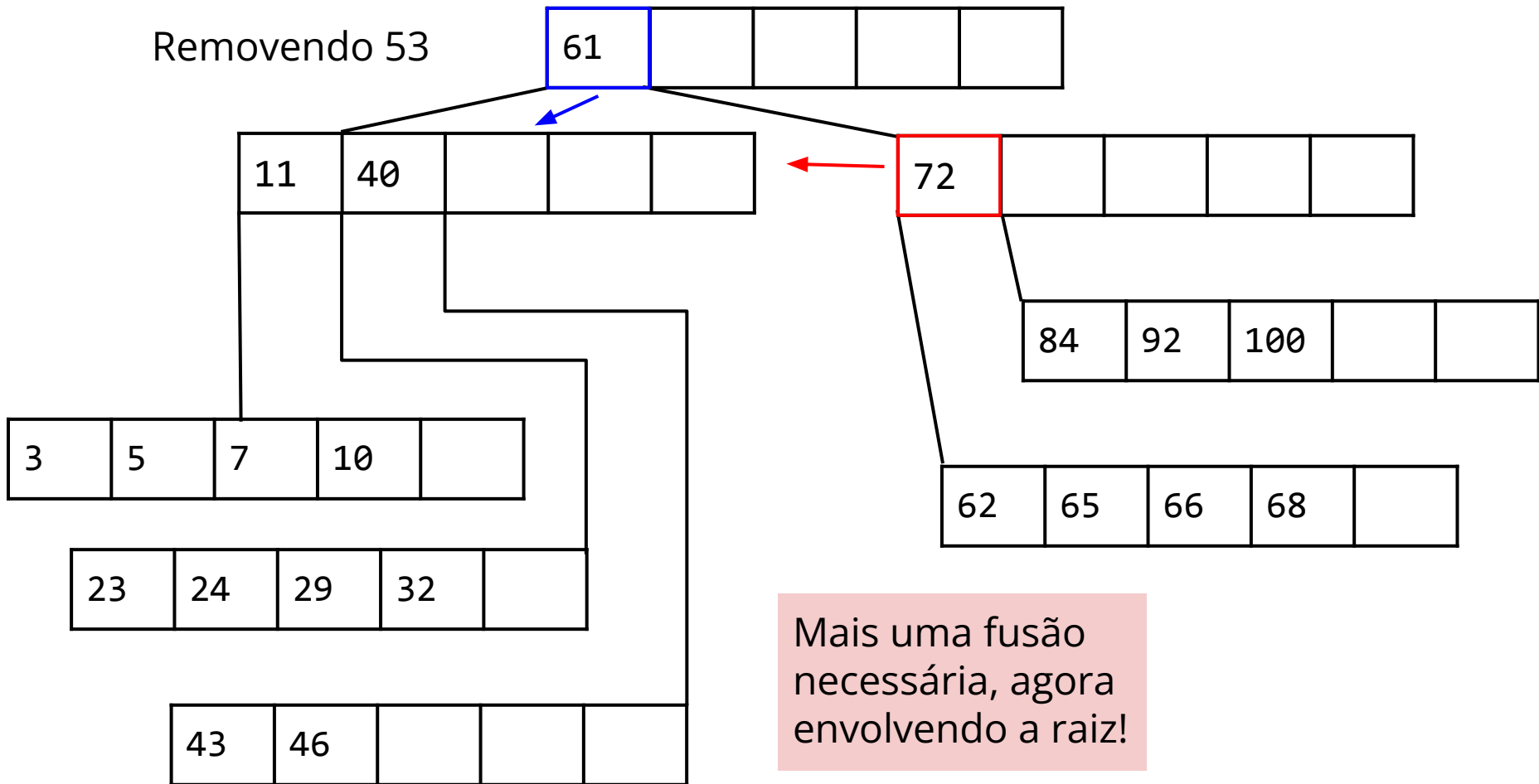
Removendo 53



Removendo 53



Removendo 53



## Árvore Final

