Fatec
Araras

Estrutura de Dados

Prof. Orlando Saraiva Júnior
orlando.saraiva@unesp.br

"The only way to learn a new programming language is by writing programs in it."

Dennis Ritchie

Estrutura de Dados

Objetivo da aula

Fatec Araras

Conhecer a estrutura de dados Pilha

Implementar pilha em C com uso de vetor

Implementar pilha em C de forma dinâmica

Implementar pilha em Python

Pilha

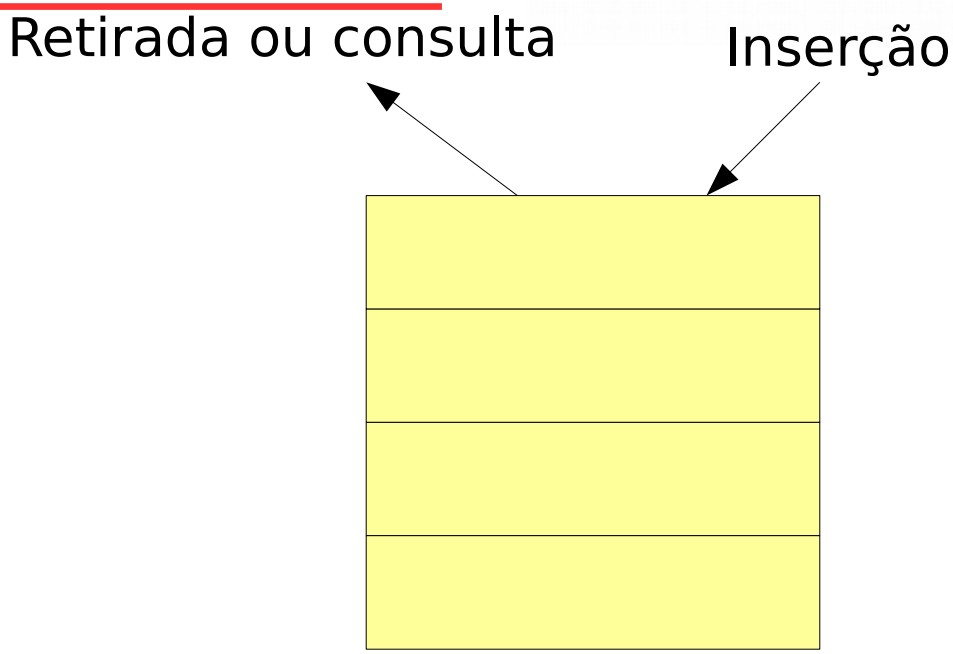
Fatec Araras

Uma **pilha** é um conjunto ordenado de itens no qual novos itens podem ser inseridos a partir do qual podem ser eliminados itens em uma extremidade chamada **topo** da pilha.

Ao contrário do que acontece com o vetor, a definição de pilha compreende a inserção e a eliminação de itens, de moto que uma pilha é um objeto dinâmico, constantemente mutável.

Quando um item é incluído numa pilha, ele é **empilhado** sobre a pilha e, quando um item é removido, ele é **desempilhado.**

Pilha



O primeiro item a ser inserido na pilha é o último a ser removido.

Esta política é conhecida pela sigla LIFO (**L**ast **I**n **F**irst **O**ut)

Sempre que houver uma remoção, o elemento removido é o que está na estrutura há menos tempo.

Exemplo de aplicação

Exemplo 1

Navegadores de Internet armazenam os endereços dos sites visitados recentemente em uma pilha. Cada vez que um usuário visita um novo site, o endereço é “enviado” para a pilha de endereços. O navegador permite o usuário desempilhe o último endereço acessado via botão voltar.

Exemplo 2

Os editores de texto geralmente fornecem um mecanismo de “desfazer” que cancela operações de edição e reverte para estados anteriores de um documento. Esta operação de desfazer pode ser obtida mantendo as alterações de texto em uma pilha.

Entendendo Pilhas

Fatec Araras

As pilhas são as estruturas de dados mais simples, mas também estão entre as mais importantes.

Formalmente, uma pilha é um tipo de dado abstrato (ADT) tal que uma instância S suporta as seguintes funcionalidades:

S.push(elem) → Adiciona o elemento elem ao topo da pilha S

S.pop() → Remove e retorna do topo da pilha S.

Espera-se um erro caso a pilha esteja vazia.

Entendendo Pilhas

Fatec Araras

No exemplo, teremos outros três funcionalidades

S.top() → Retorna a referência do topo da lista, sem remover o elemento da lista.

S.is_empty() → Returna verdadeiro, caso a pilha esteja vazia.

S.tamanho() → Returna o número de elementos da pilha S.

Entendendo Pilhas

Fatec Araras

Operação	Valor de retorno	Conteúdo da pilha
S.push(5)	-	[5]
S.push(3)	-	[5, 3]
S.tamanho()	2	[5, 3]
S.pop()	3	[5]
S.is_empty()	False	[5]
S.pop()	5	[ ]
S.is_empty()	True	[ ]
S.pop()	“erro”	[ ]
S.push(7)	-	[7]
S.push(9)	-	[7, 9]
S.push(4)	-	[7, 9 , 4]
S.tamanho()	3	[7, 9 , 4]
S.push(8)	-	[7, 9 , 4 , 8]

Entendendo Pilhas

Qual o conteúdo da pilha ?

Operação	Valor de retorno	Conteúdo da pilha
S.push(6)	-	[7 , 9 , 4 , 8 , 6]
S.push(1)	-	? ?
S.tamanho()	6	? ?
S.pop()	1	? ?
S.is_empty()	False	? ?
S.pop()	6	? ?
S.pop()	8	? ?
S.pop()	4	? ?
S.pop()	9	? ?
S.push(1)	-	? ?
S.pop()	1	? ?
S.pop()	7	[ ]
S.pop()	“erro”	[ ]

Entendendo Pilhas
(continuação)

Fatec Araras

Operação	Valor de retorno	Conteúdo da pilha
S.push(6)	-	[7 , 9 , 4 , 8 , 6]
S.push(1)	-	[7 , 9 , 4 , 8 , 6 , 1]
S.tamanho()	6	[7 , 9 , 4 , 8 , 6 , 1]
S.pop()	1	[7 , 9 , 4 , 8 , 6]
S.is_empty()	False	[7 , 9 , 4 , 8 , 6]
S.pop()	6	[7 , 9 , 4 , 8]
S.pop()	8	[7 , 9 , 4]
S.pop()	4	[7 , 9]
S.pop()	9	[7]
S.push(1)	-	[7, 1]
S.pop()	1	[7]
S.pop()	7	[ ]
S.pop()	“erro”	[ ]

Implementando pilha com C
Primeira versão

Prof. Me. Orlando Saraiva Júnior

Pilha em C: Primeira versão

Fatec Araras

Há várias formas de se representar uma pilha. Examinaremos a mais simples delas.

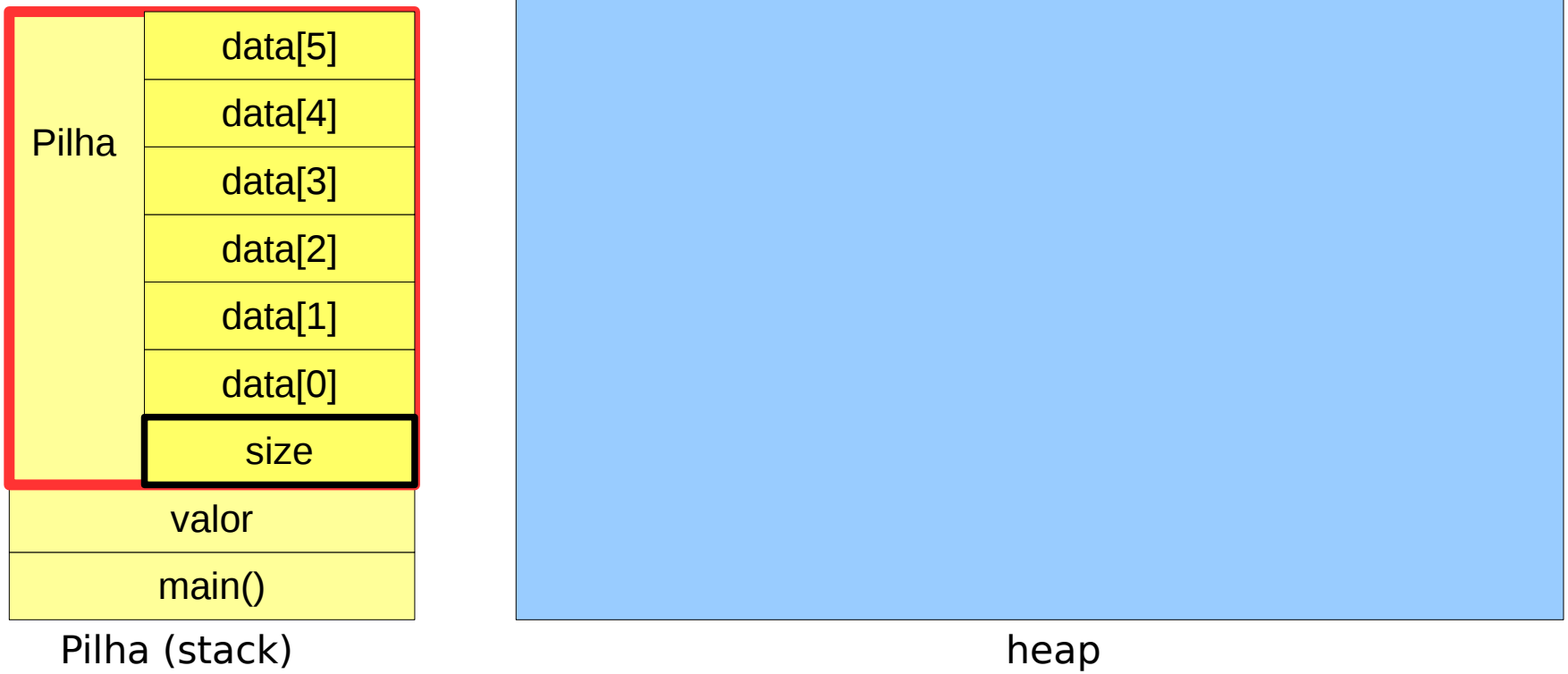
Uma pilha é um conjunto ordenado de itens, e em C já contém um tipo de dado que representa um conjunto ordenado de itens: o vetor.

Contudo, um vetor e uma pilha são entidades totalmente diferentes. O número de elementos de um vetor é fixo. Em termos gerais, o usuário não pode alterar este número. Uma pilha é um objeto dinâmico, cujo tamanho está sempre mudando, a medida que os itens são empilhados e desempilhados.

Uma pilha em C pode ser declarada como uma estrutura contendo dois objetos: um vetor para armazenar os elementos e um inteiro para indicar a posição da pilha.

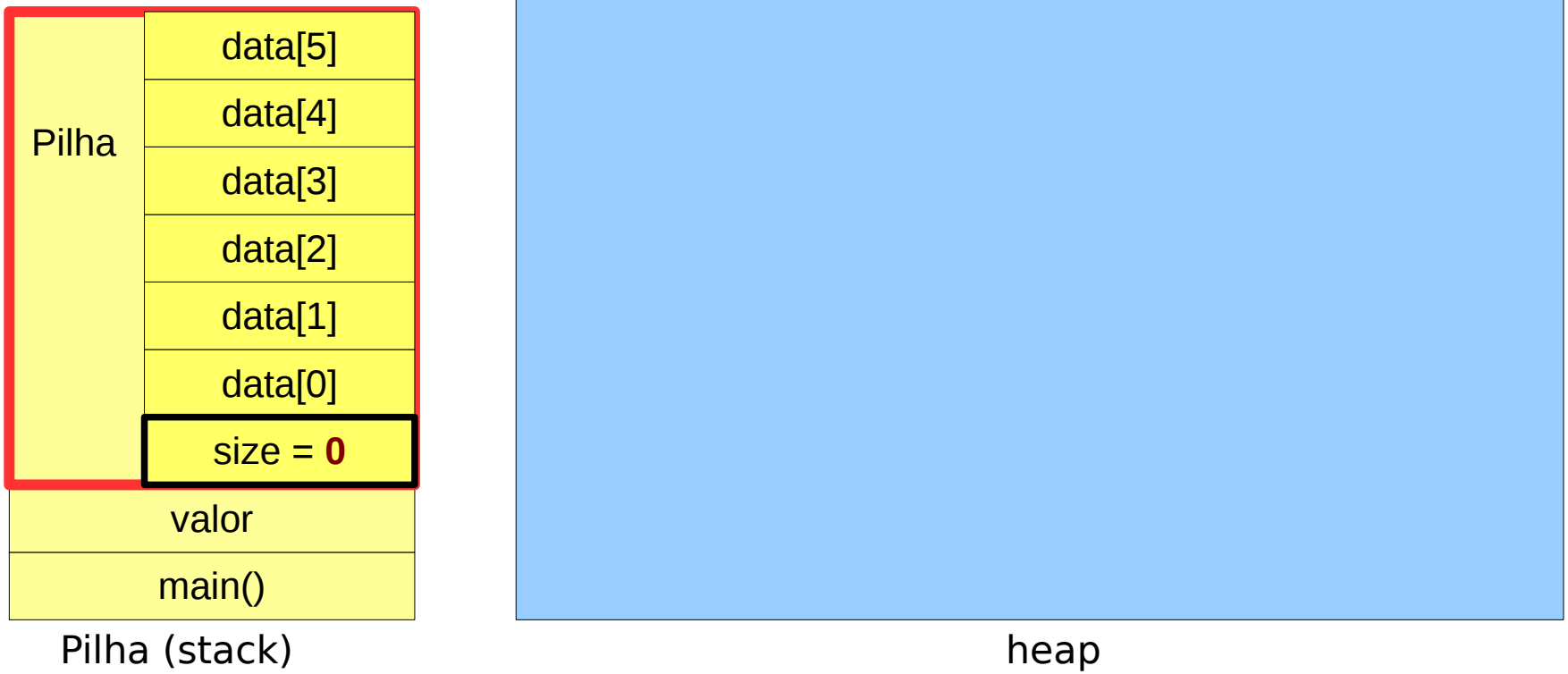
pilha1.c

Variáveis



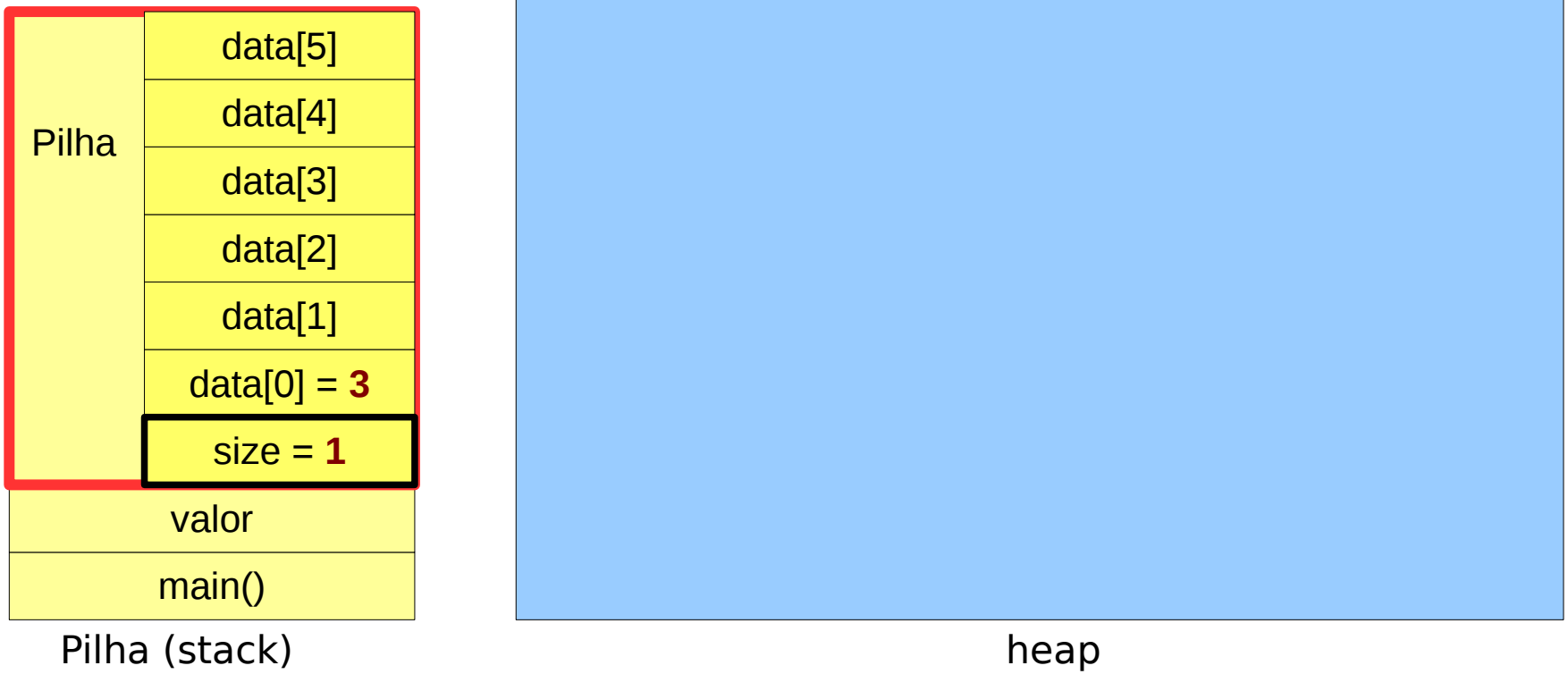
pilha1.c

Variáveis



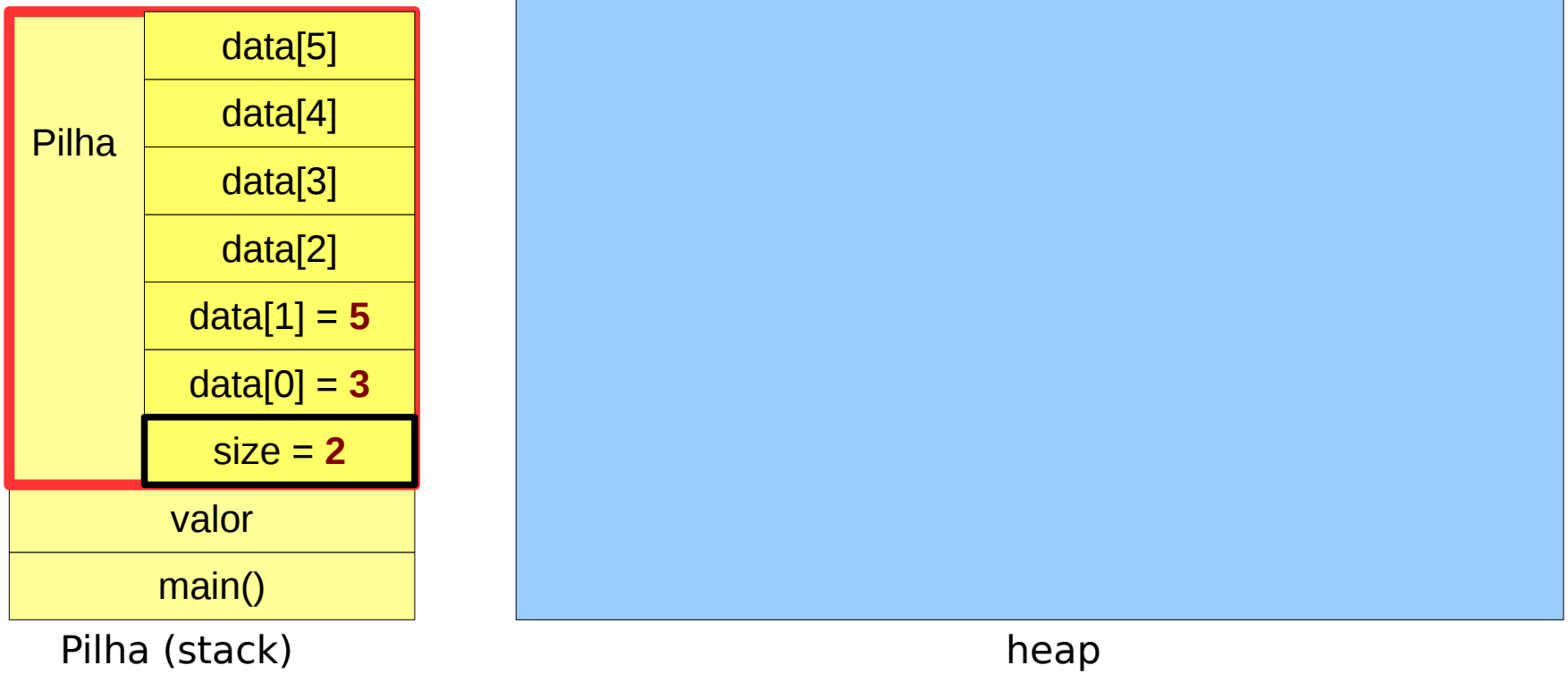
pilha1.c

Variáveis



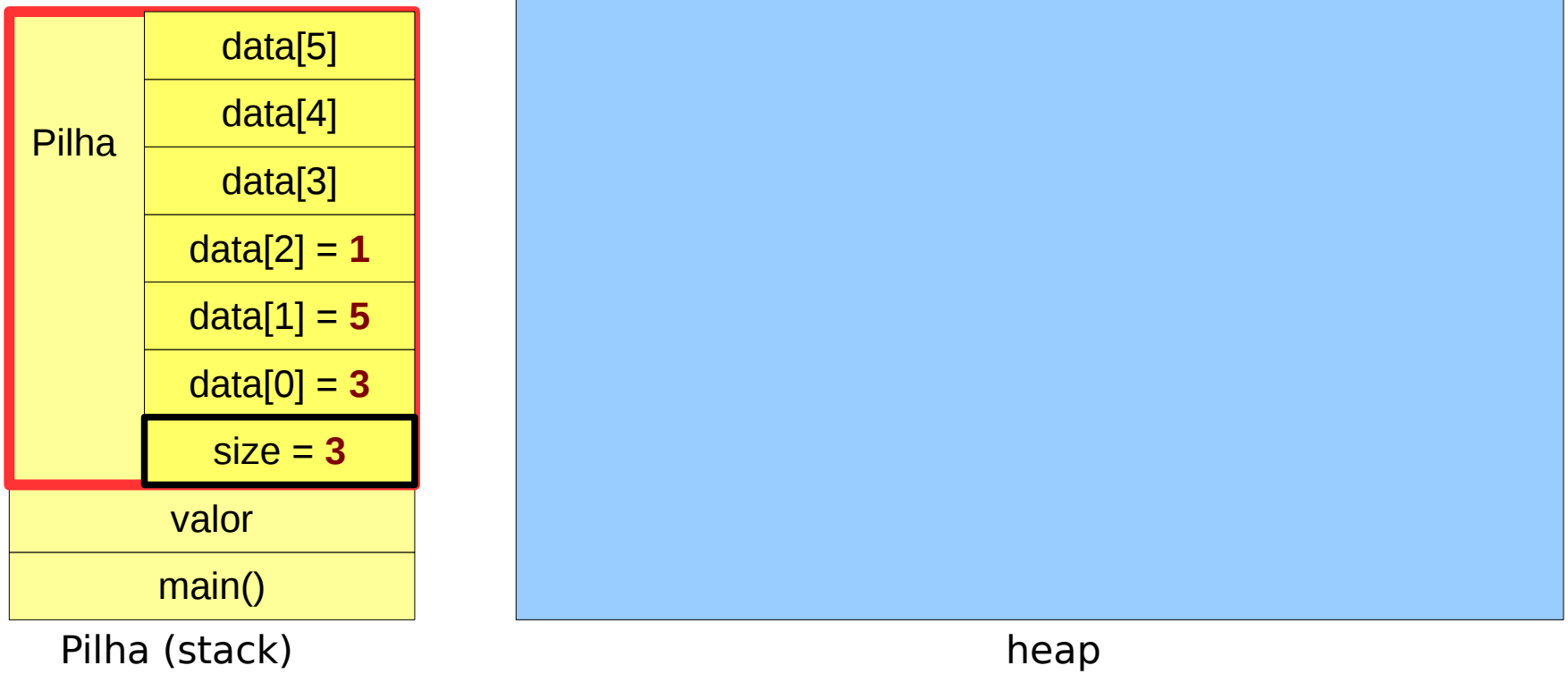
pilha1.c

Variáveis



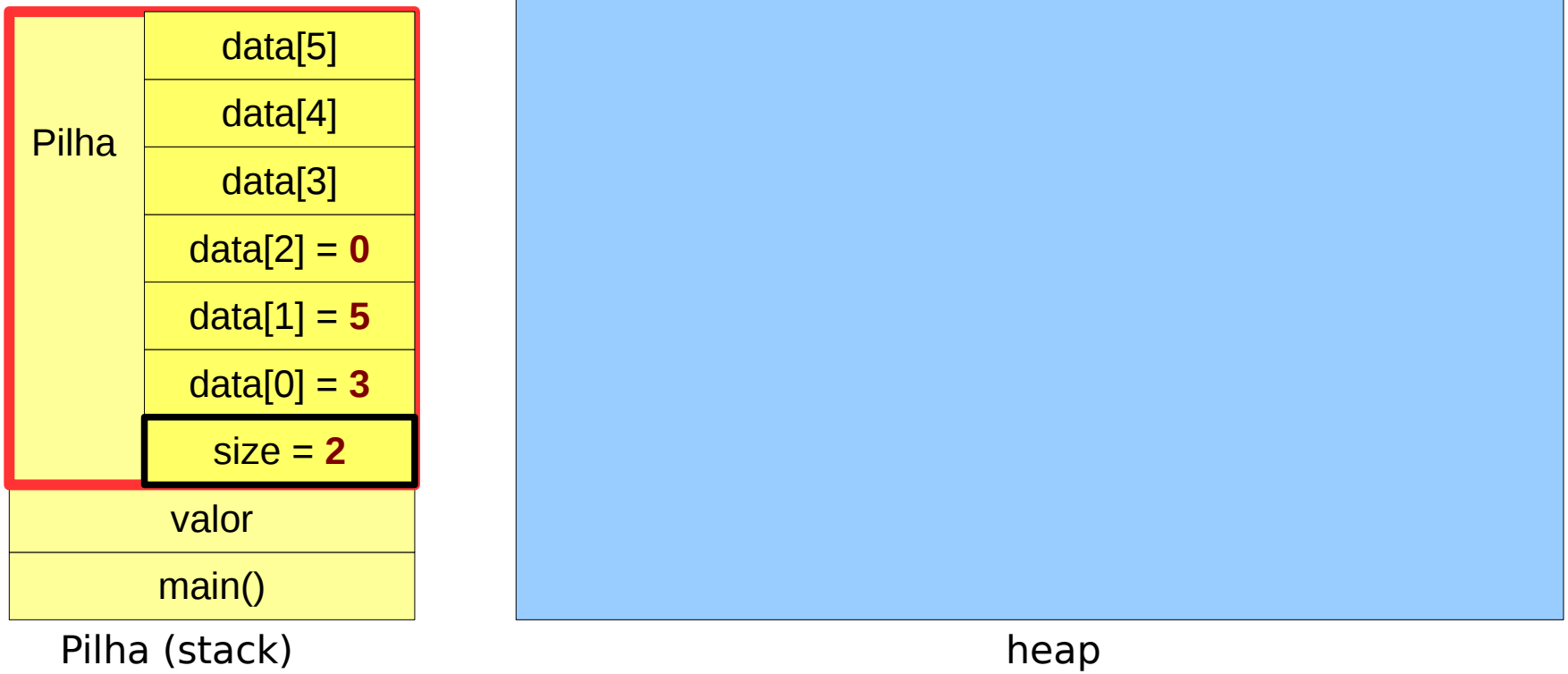
pilha1.c

Variáveis



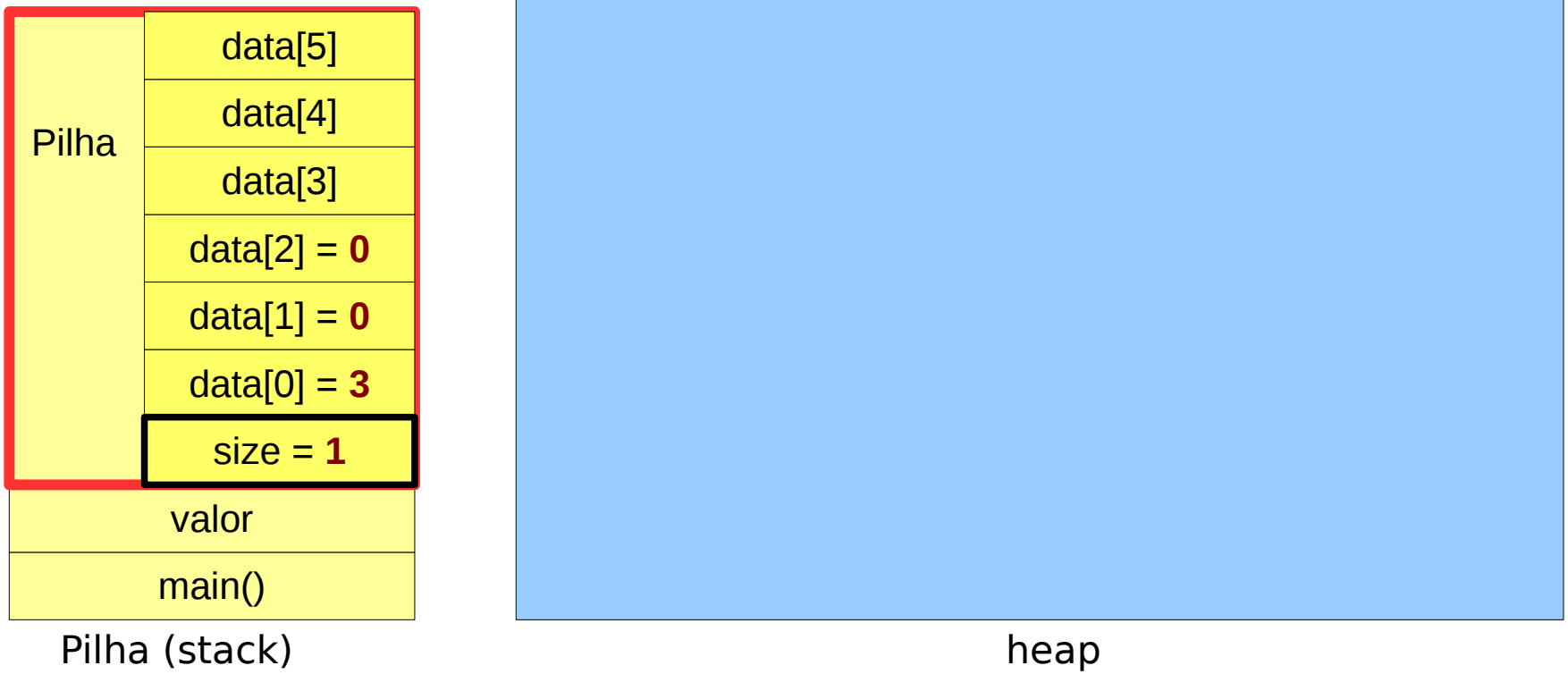
pilha1.c

Variáveis



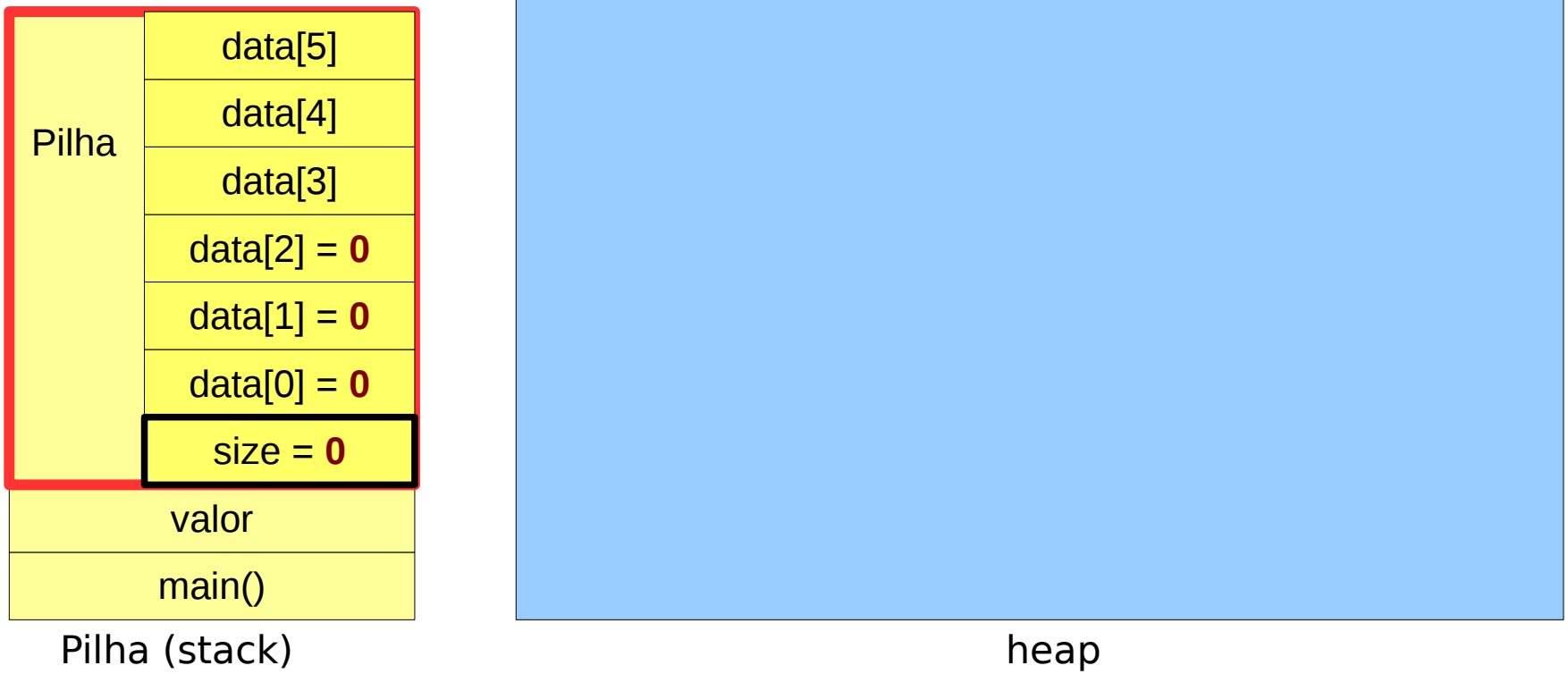
pilha1.c

Variáveis



pilha1.c

Variáveis



Implementando pilha com C
Segunda versão

Prof. Me. Orlando Saraiva Júnior

Pilha em C: Segunda versão

Fatec Araras

Nesta segunda versão, não há um limite para o número de elementos que podem ser empilhados.

Uma estrutura (Node) serve para armazenar o valor empilhado. Uma outra estrutura (Pilha) contém um ponteiro para o nó (Node) e o tamanho atual da pilha.

Diferente da versão anterior, nesta implementação não há um limite de elementos possíveis.

Ao empilhar um elemento (operação push), instancia um novo nó, empilhando-o na estrutura Pilha.

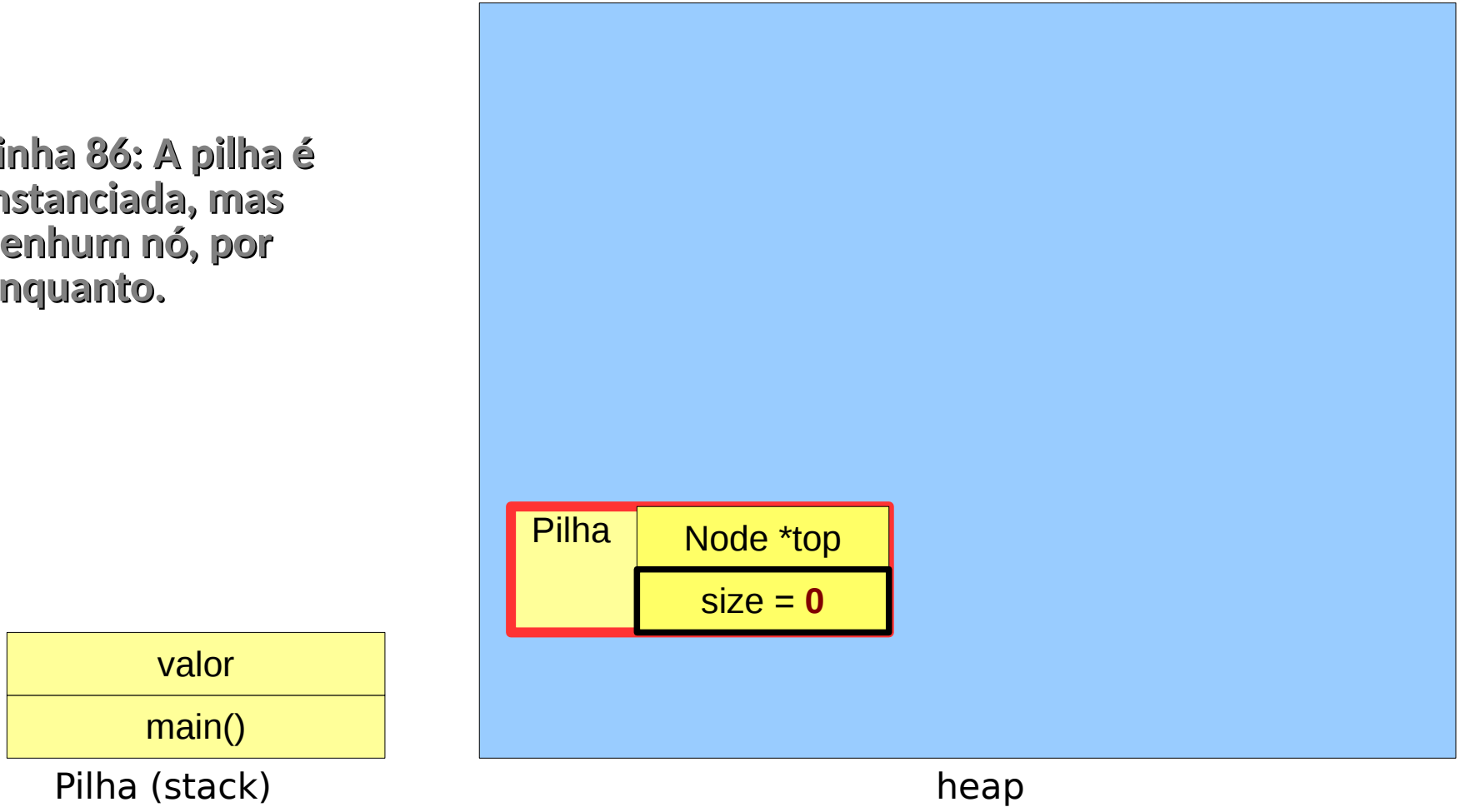
Ao desempilhar um elemento (operação pop), o topo da pilha é desempilhado e o nó no topo é desvinculado (via chamada de sistema free()).

pilha2.c

Nenhum nó alocado (ainda)

Fatec Araras

Linha 86: A pilha é instanciada, mas nenhum nó, por enquanto.

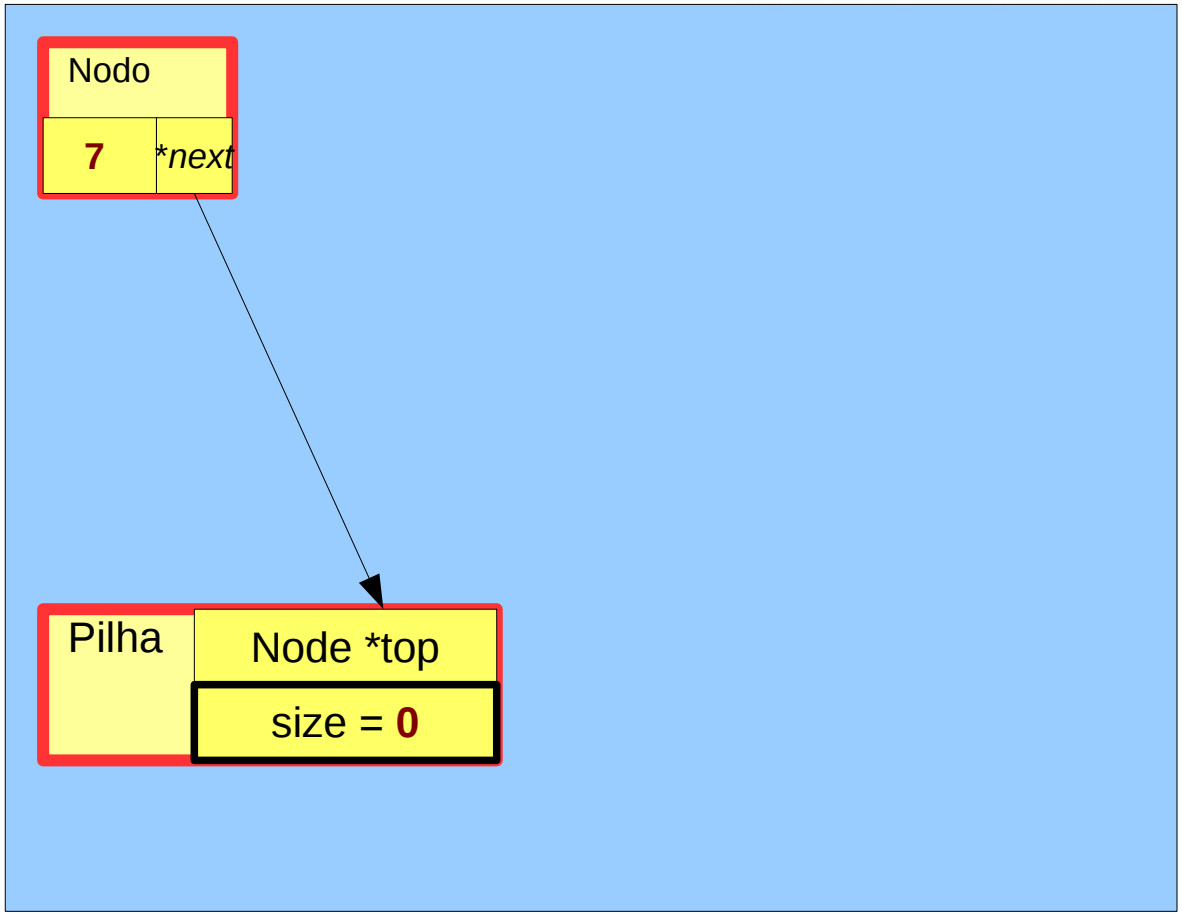


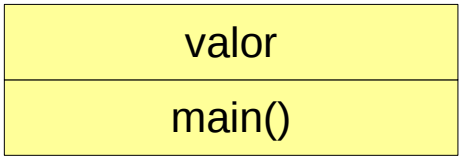
pilha2.c

Primeiro push

Fatec Araras

Linha 36





Pilha (stack)

heap

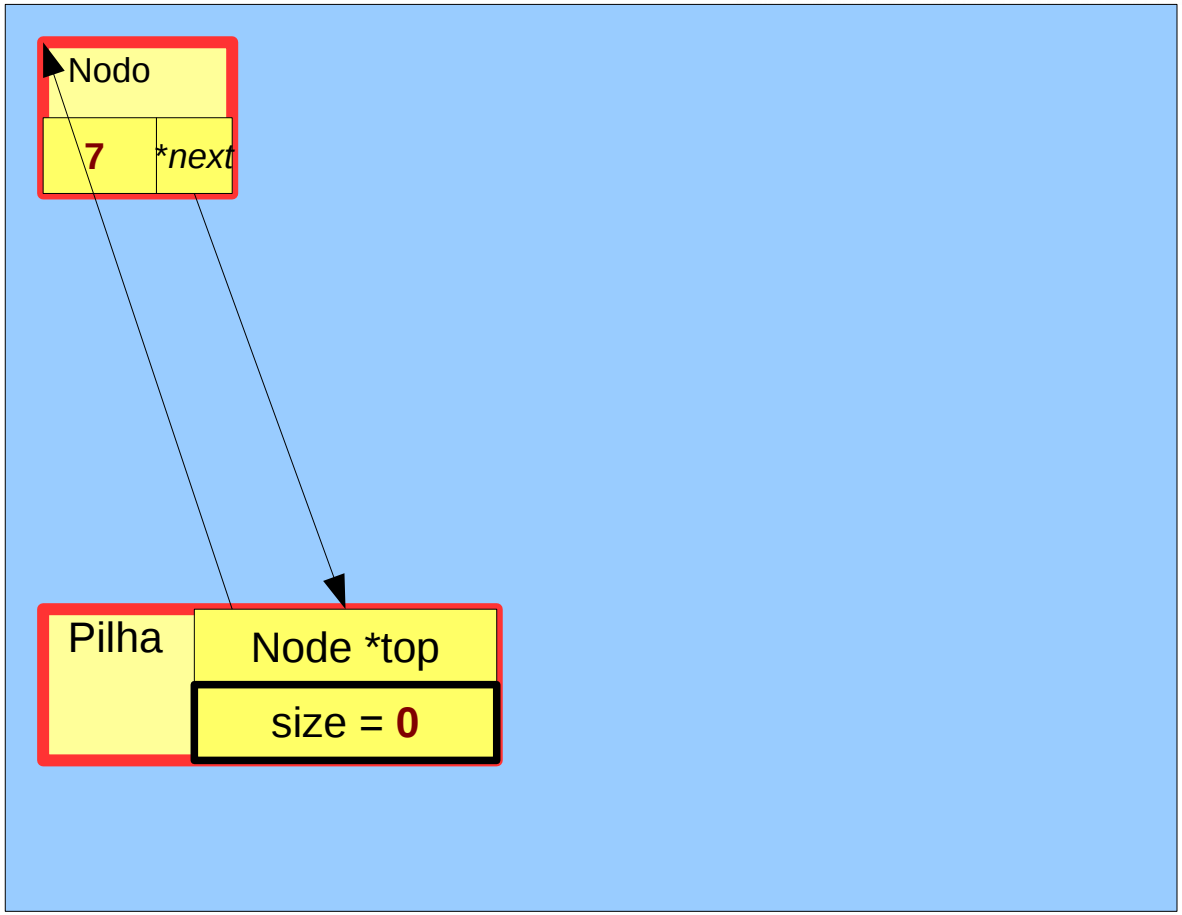
pilha2.c

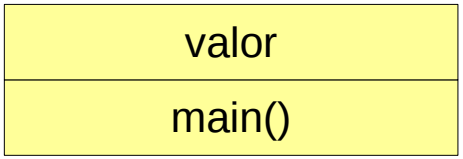
Primeiro push

Fatec Araras

Linha 37

O contador size será incrementado na linha 39



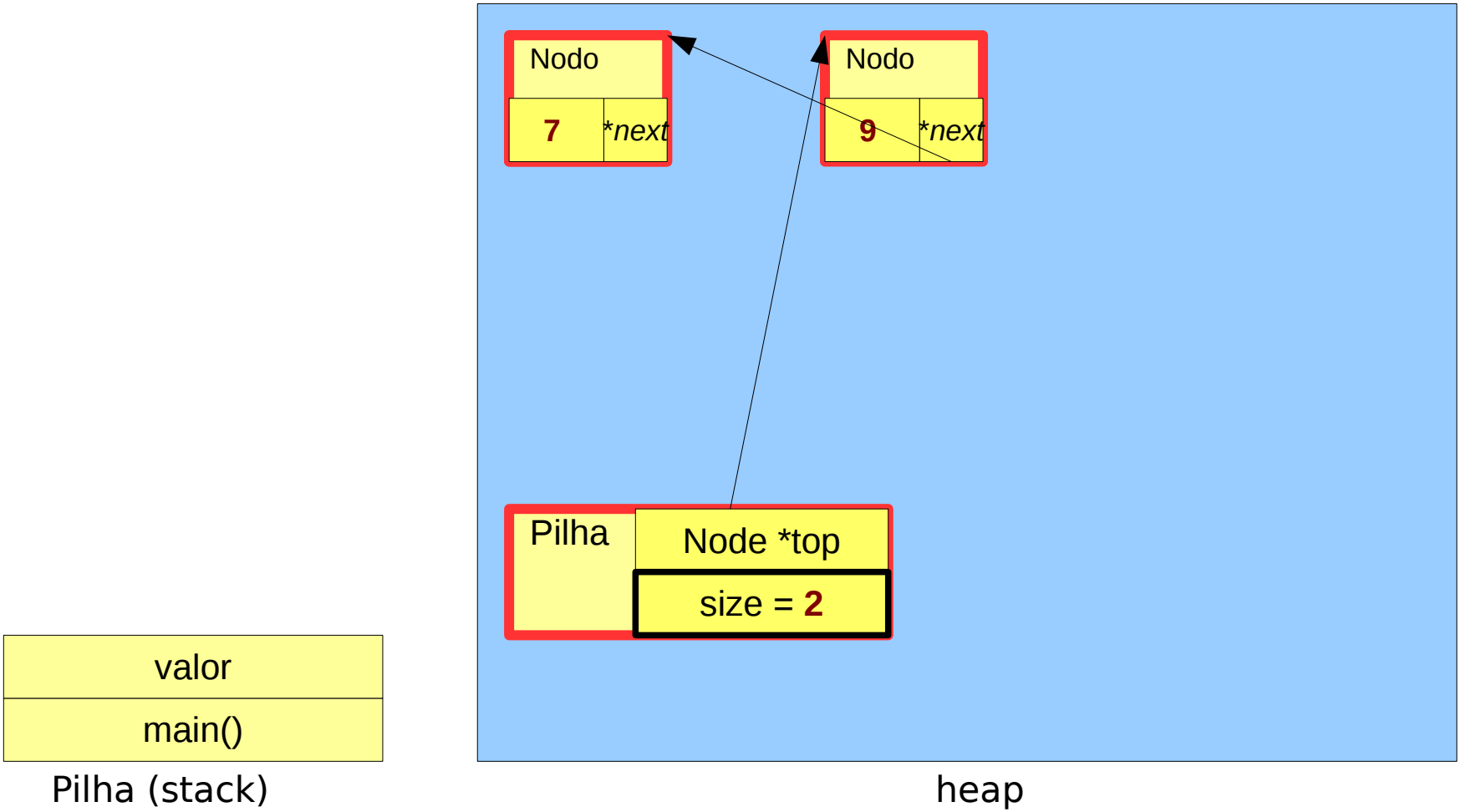


Pilha (stack)

heap

pilha2.c

Segundo push

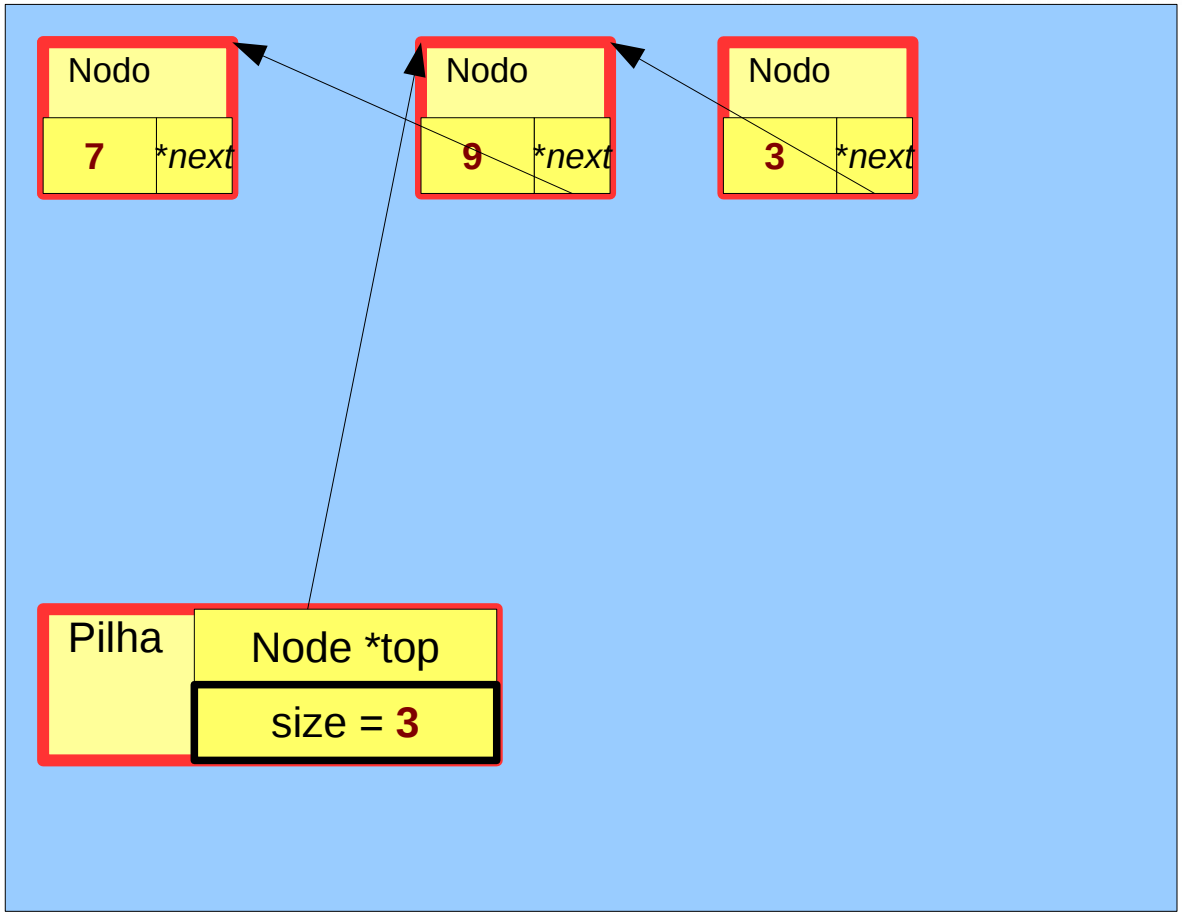


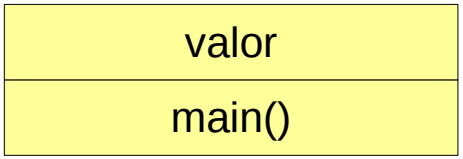
pilha2.c

Terceiro push

Fatec Araras

Linha 36





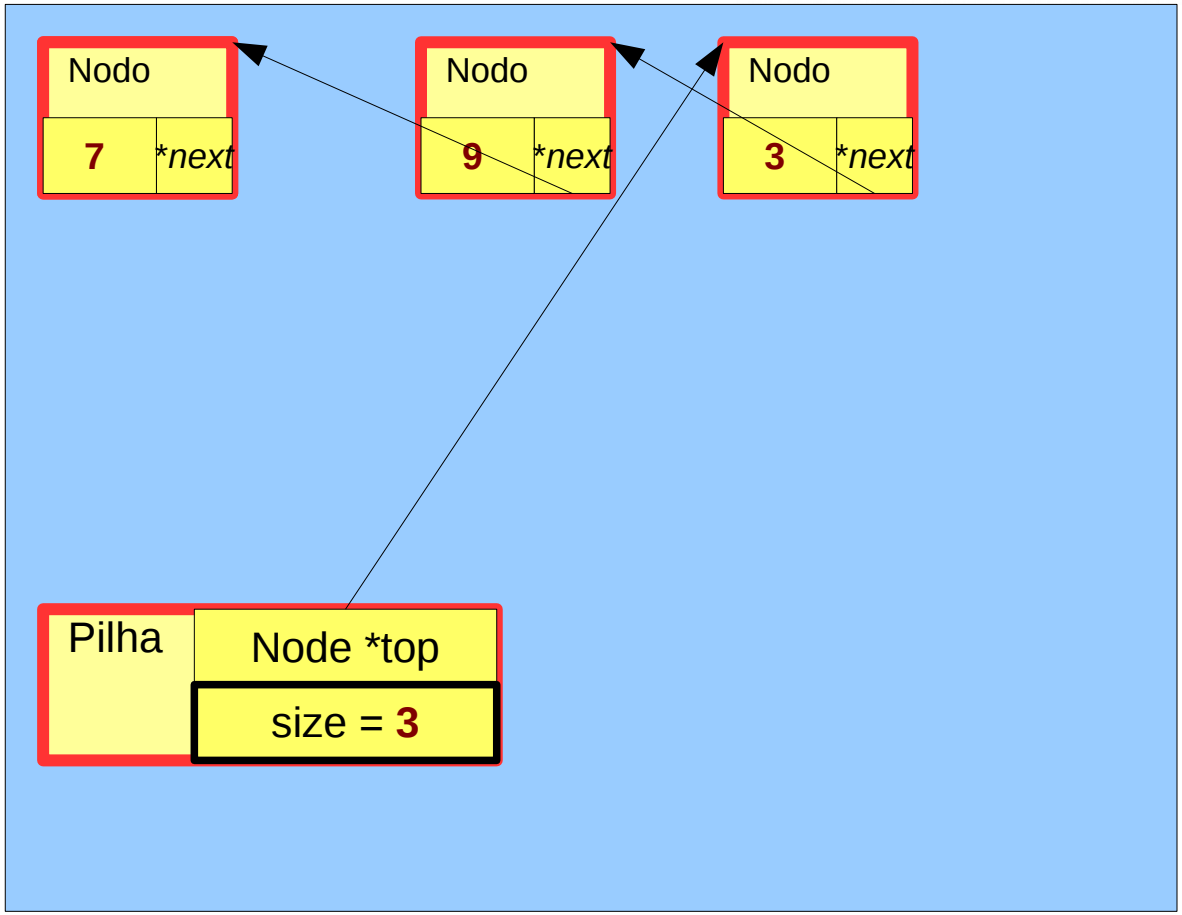
Pilha (stack)

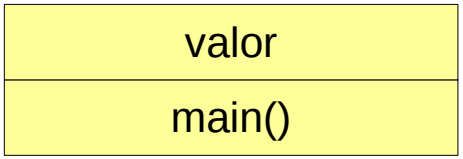
heap

pilha2.c

Terceiro push

Linha 37





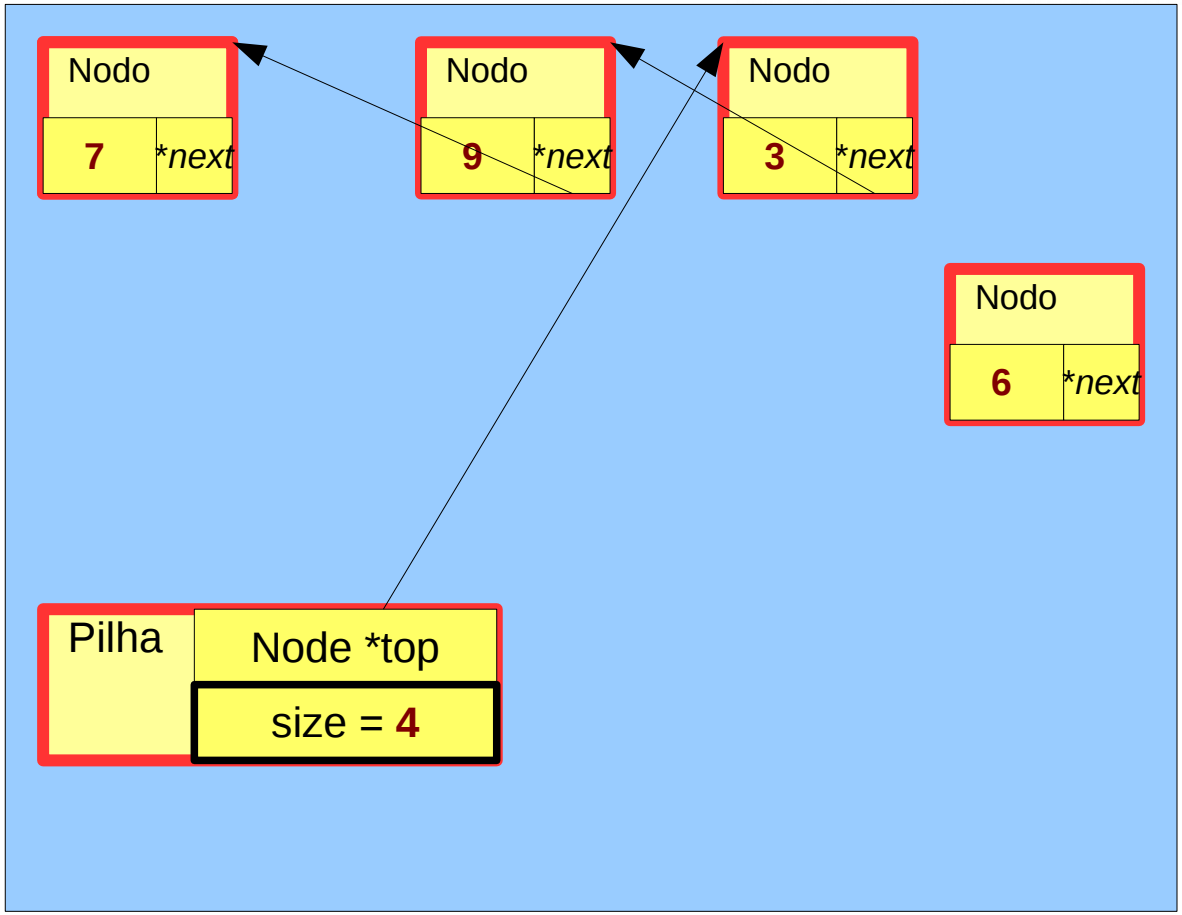
Pilha (stack)

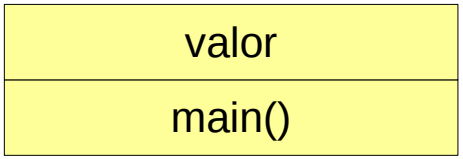
heap

pilha2.c

Quarto push

Linha 35





Pilha (stack)

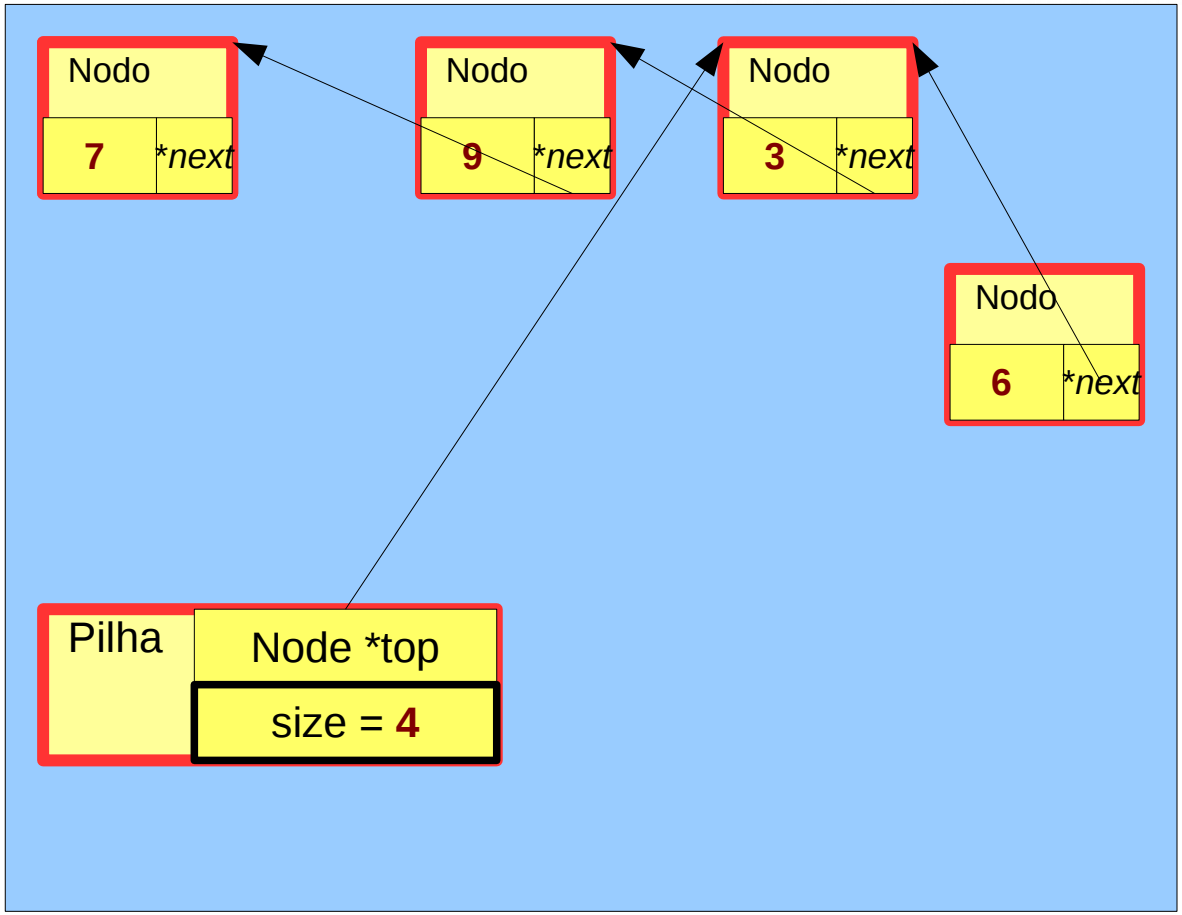
heap

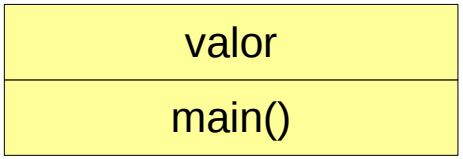
pilha2.c

Quarto push

Fatec Araras

Linha 36





Pilha (stack)

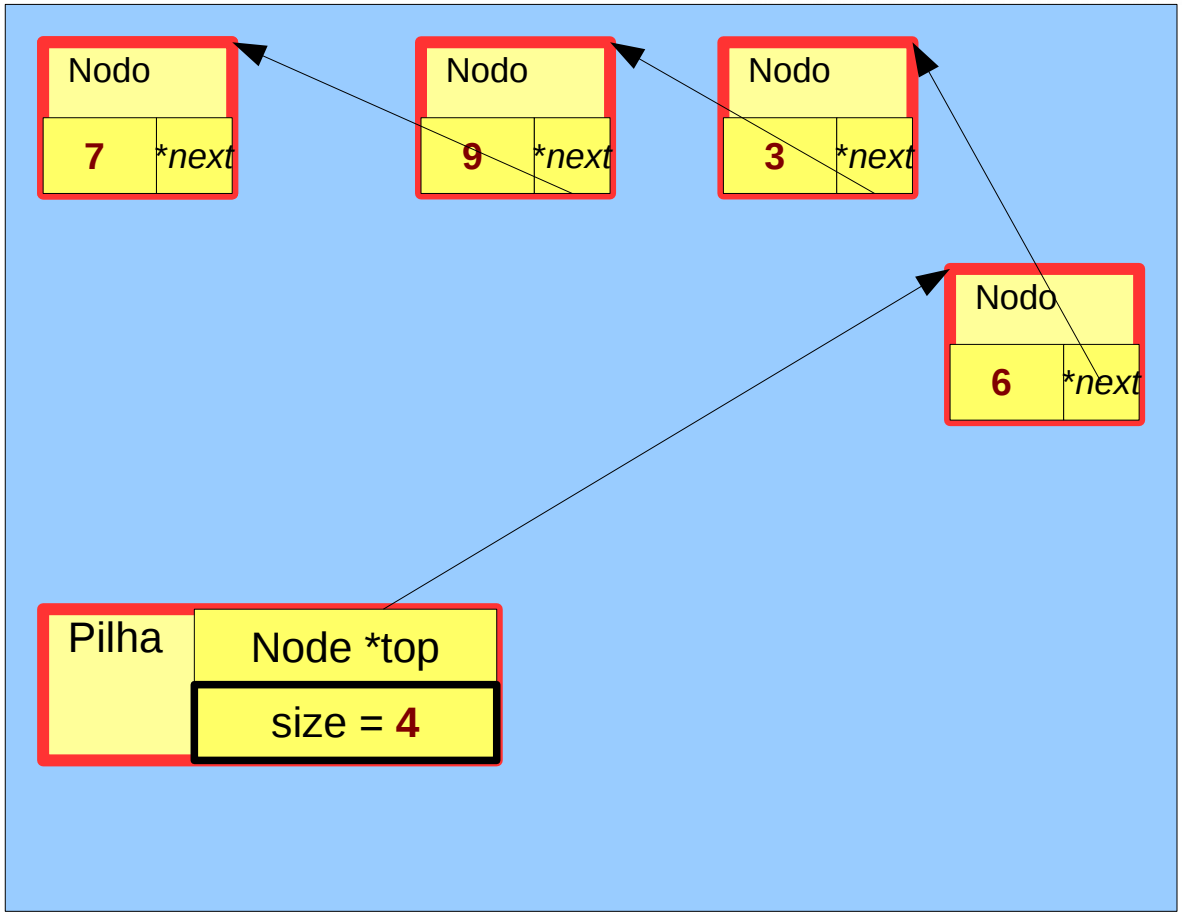
heap

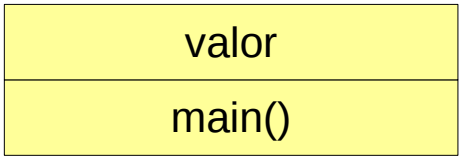
pilha2.c

Quarto push

Fatec Araras

Linha 37



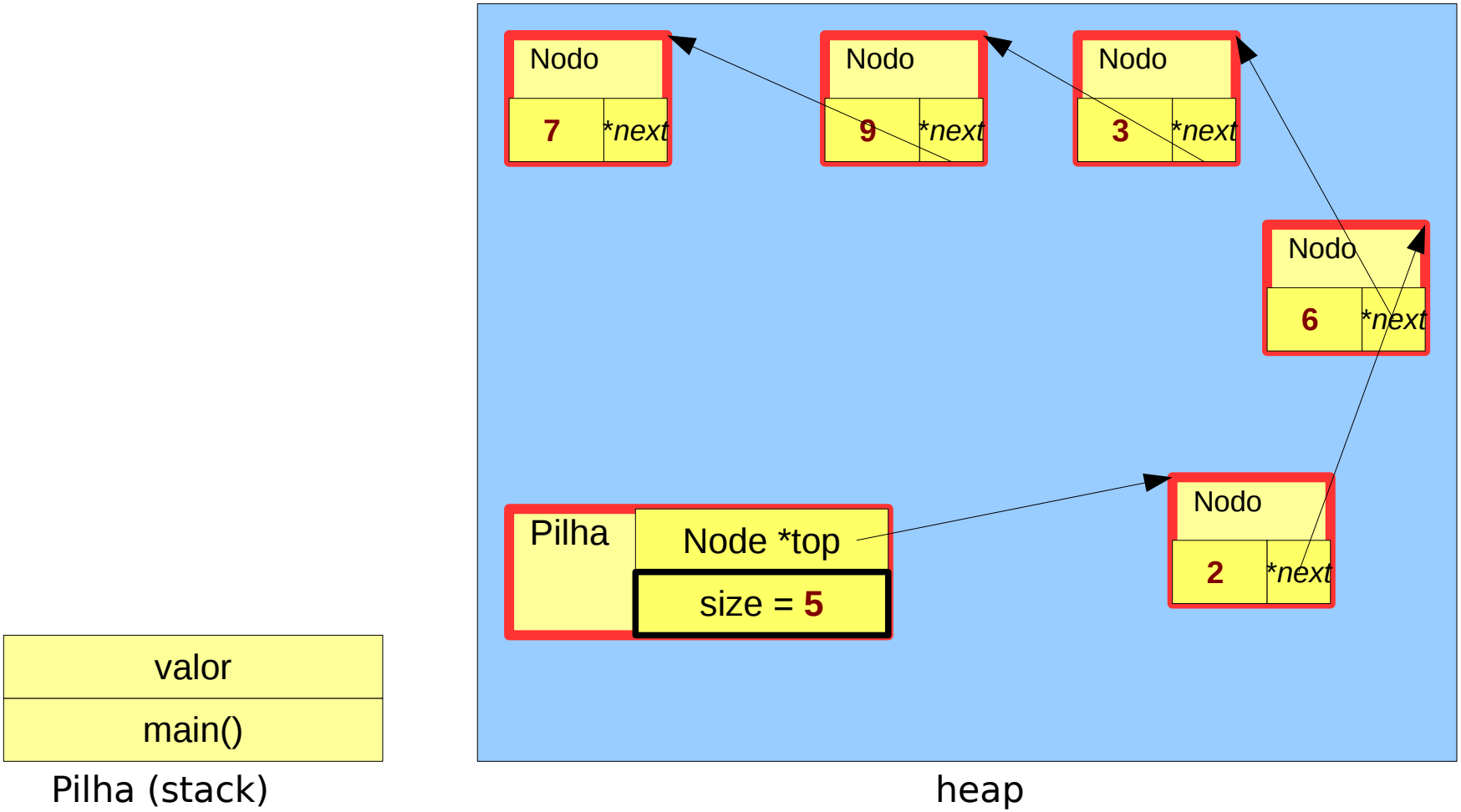


Pilha (stack)

heap

pilha2.c

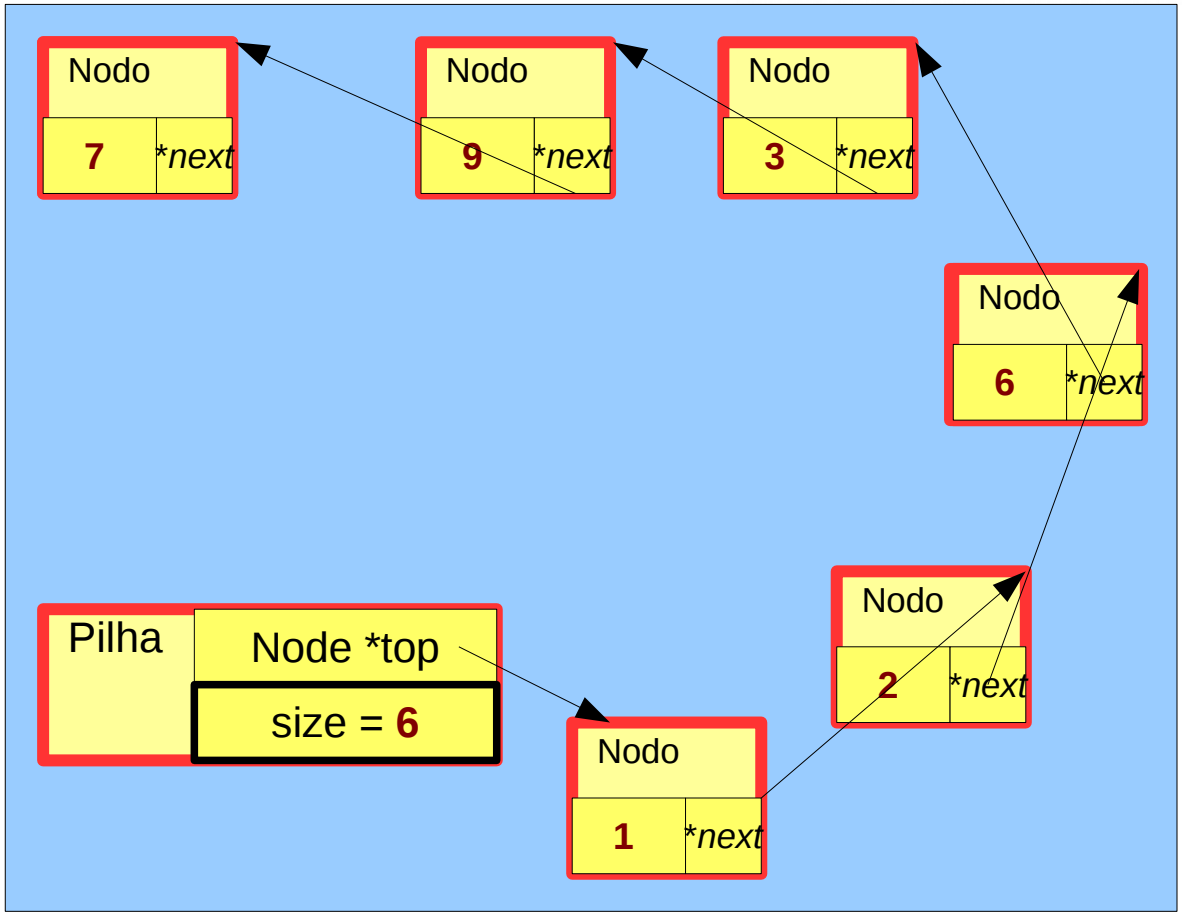
Quinto push

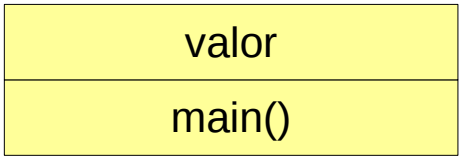


pilha2.c

Sexto push

Fatec Araras



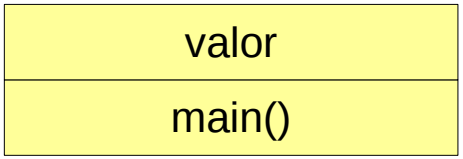


Pilha (stack)

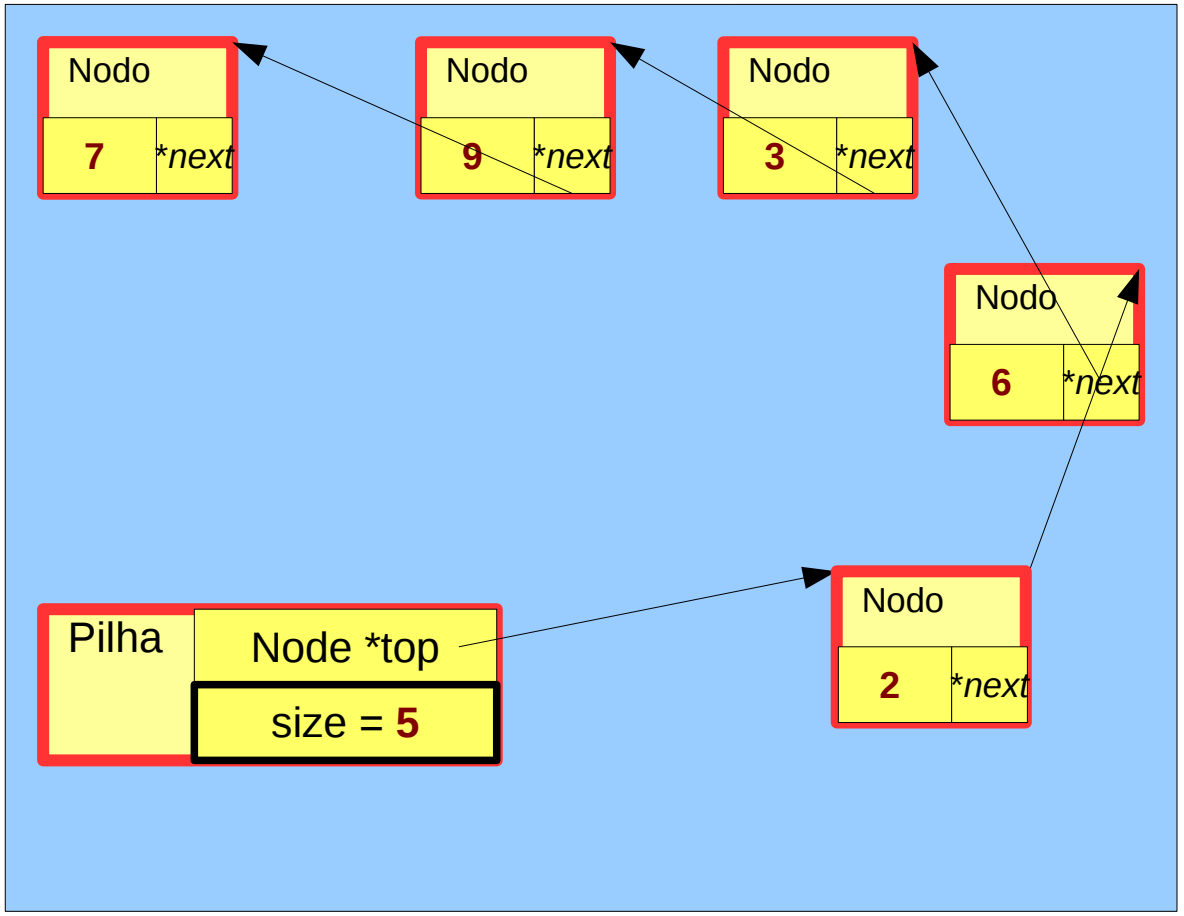
heap

pilha2.c

Primeiro POP



Pilha (stack)



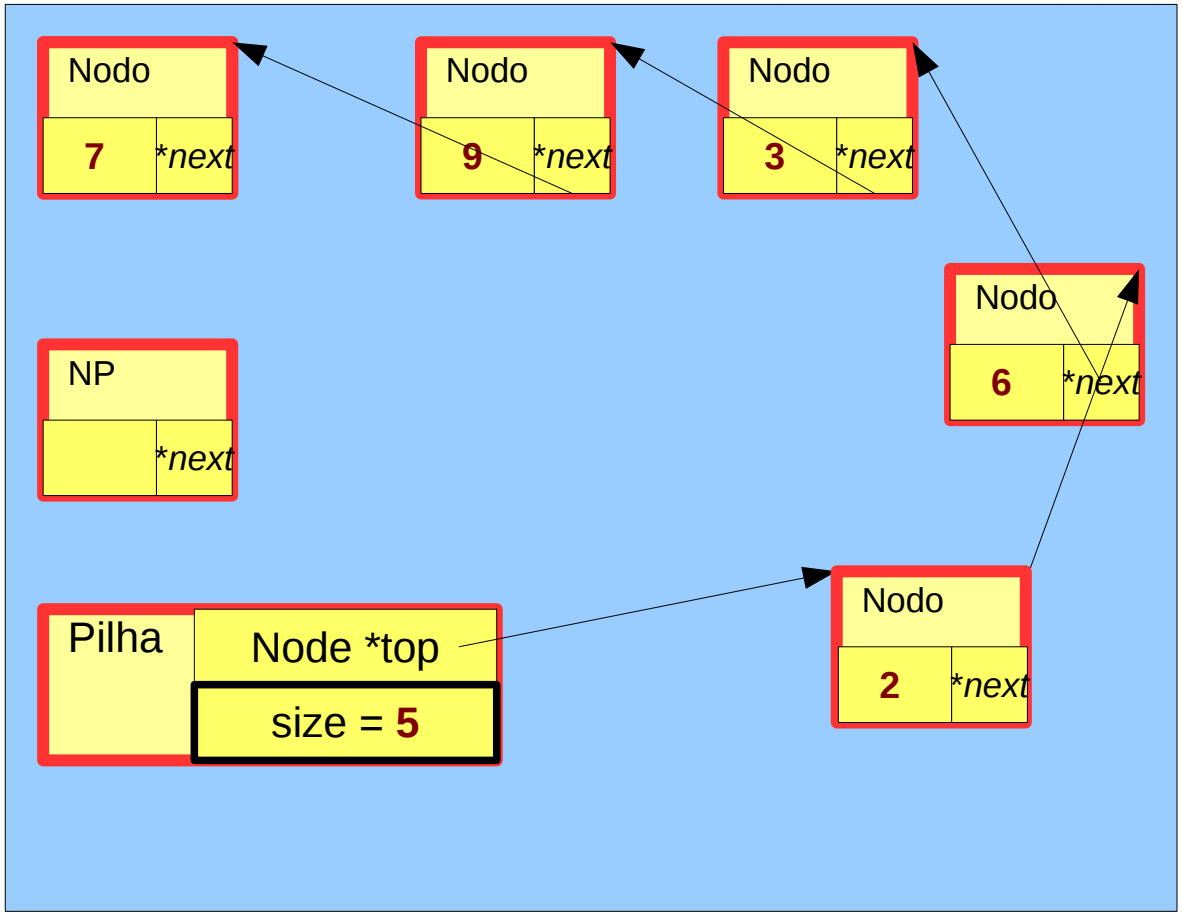
heap

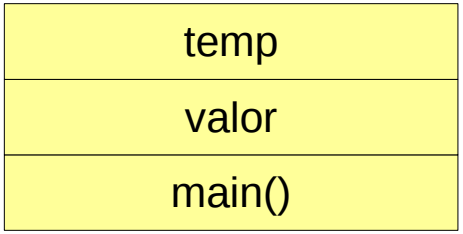
pilha2.c

Segundo POP

Fatec Araras

Linha 44





Pilha (stack)

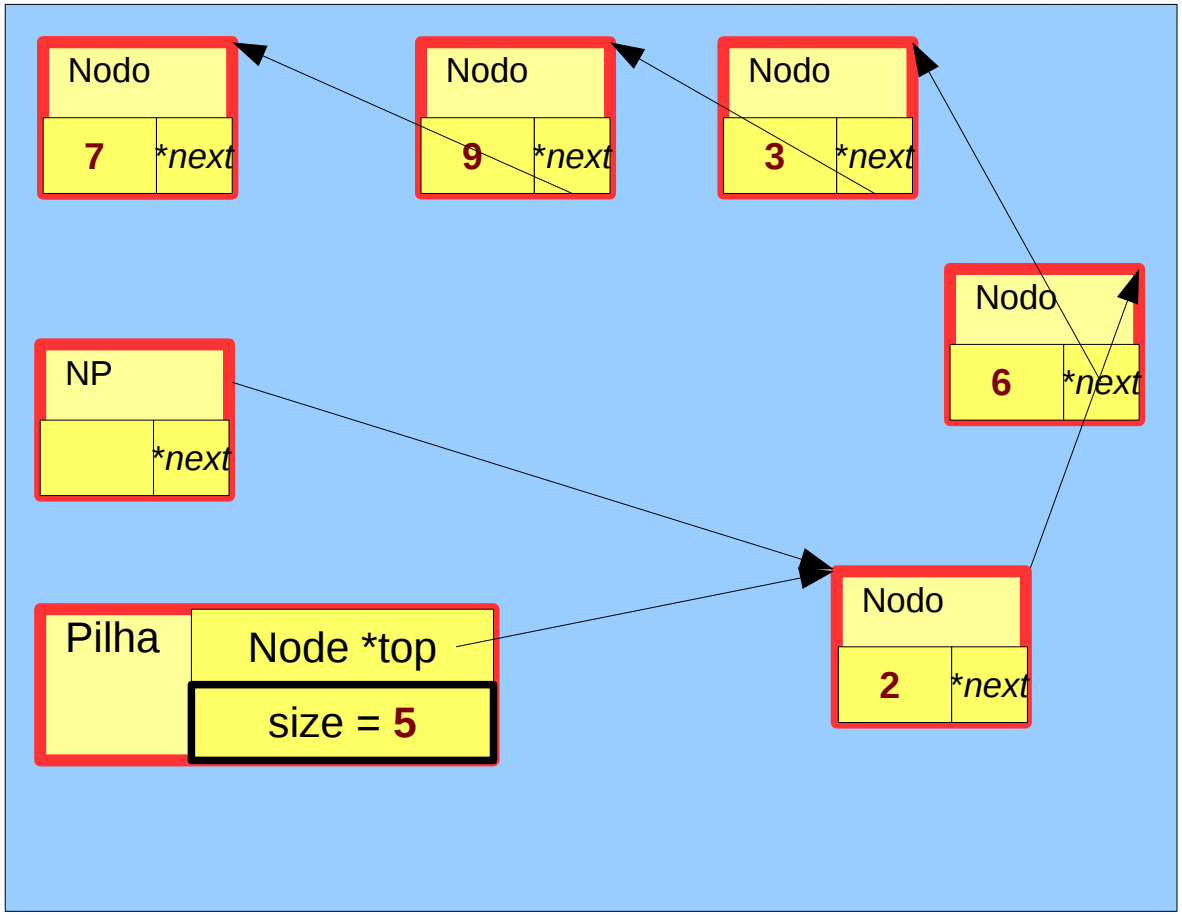
heap

pilha2.c

Segundo POP

Fatec Araras

Linha 51



temp = **2**
valor
main()

Pilha (stack)

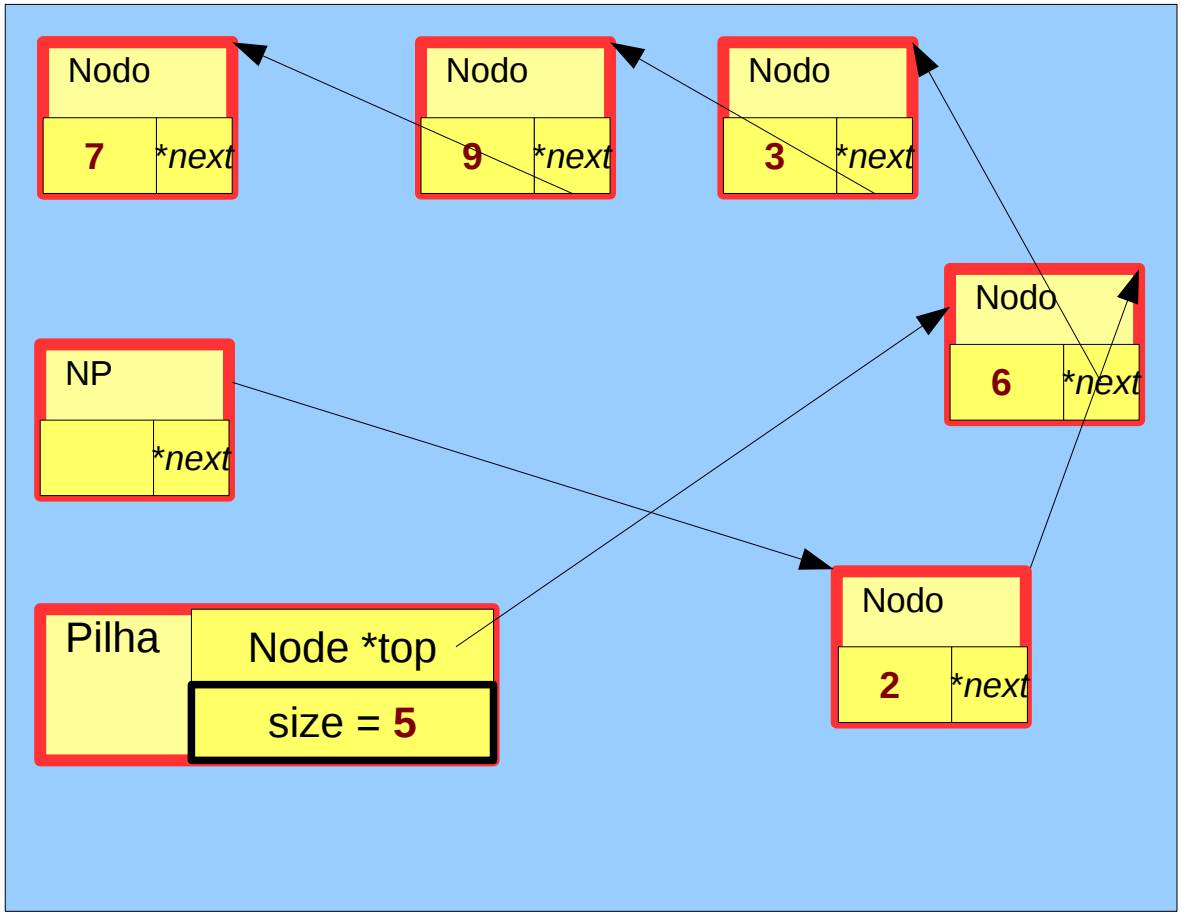
heap

pilha2.c

Segundo POP

Fatec Araras

Linha 52



temp = **2**
valor
main()

Pilha (stack)

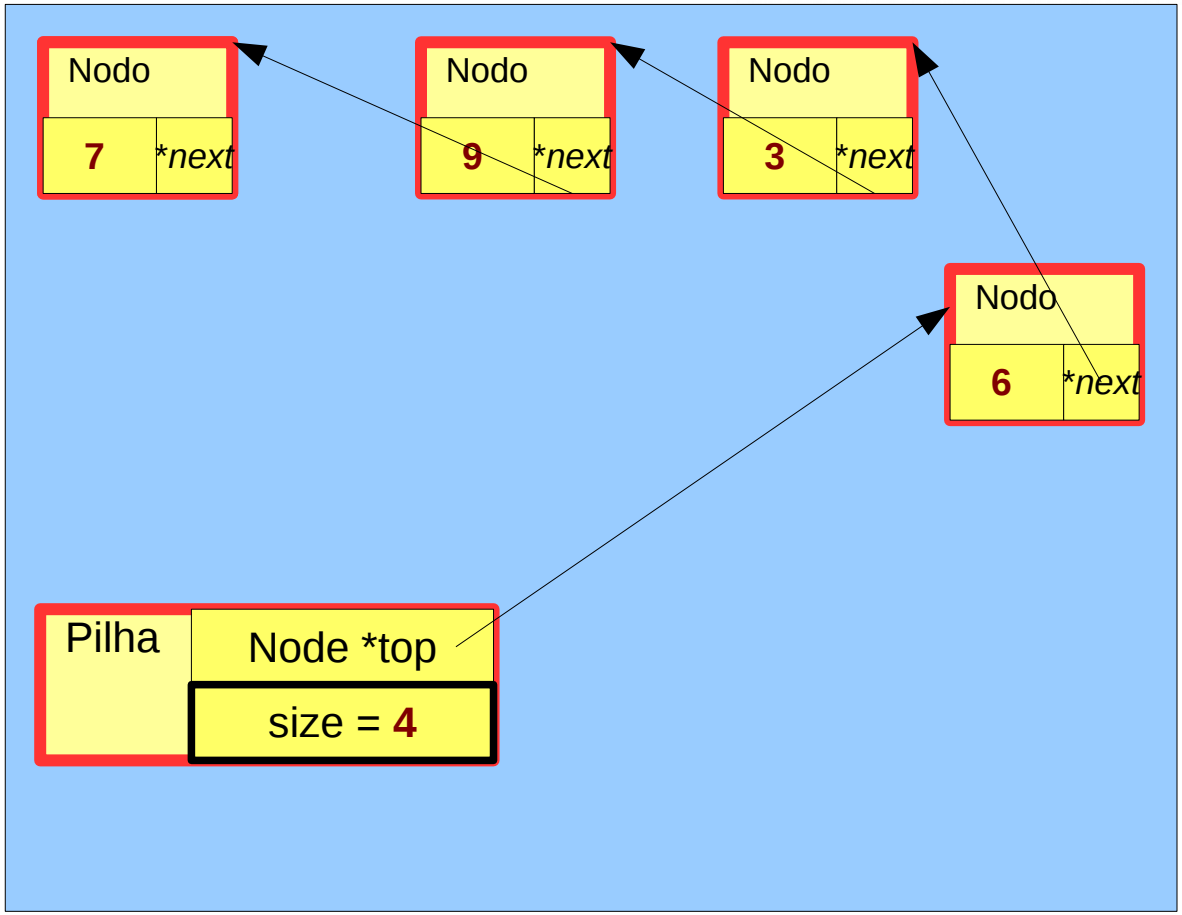
heap

pilha2.c

Segundo POP

Fatec Araras

Linha 54



temp = **2**
valor
main()

Pilha (stack)

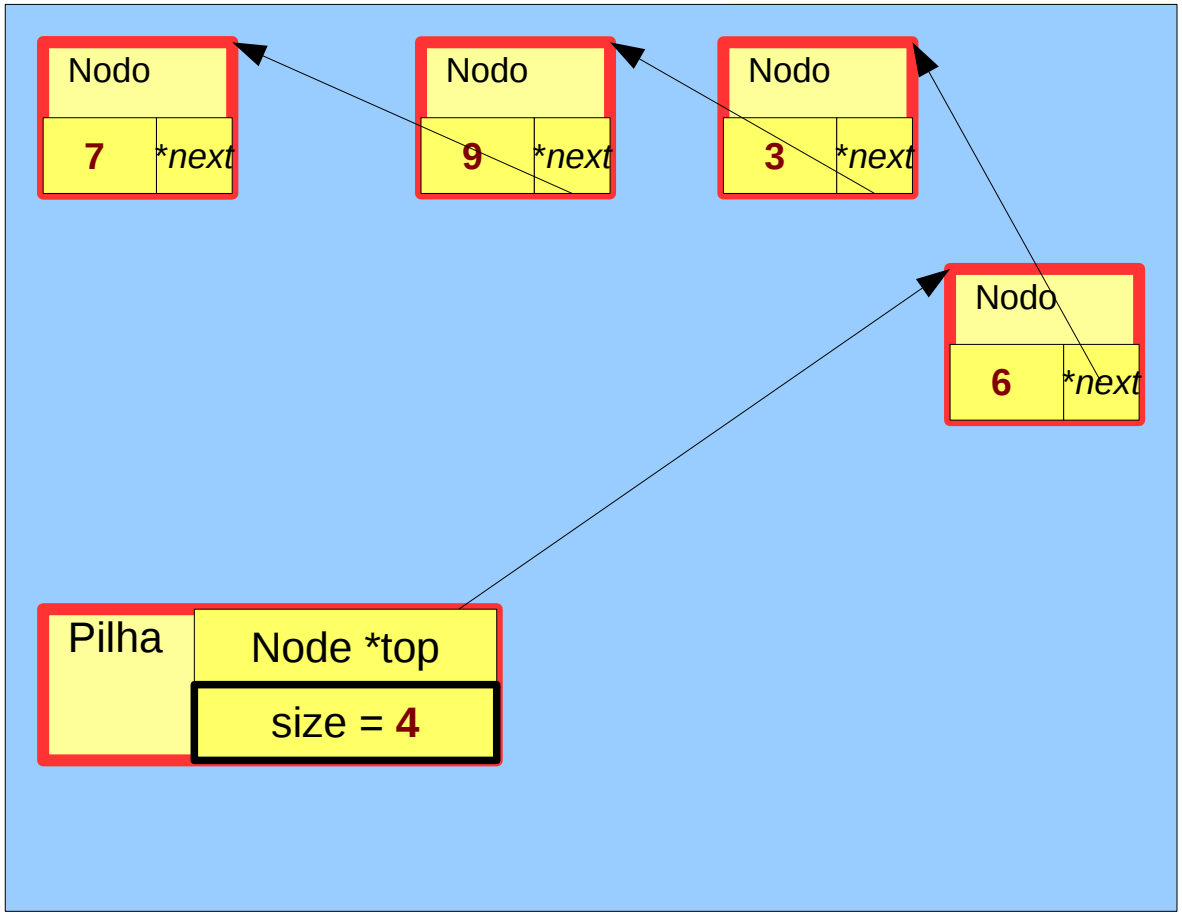
heap

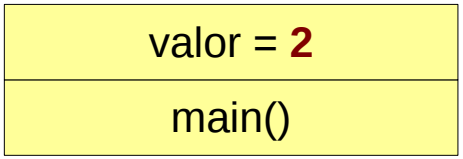
pilha2.c

Returno do segundo POP

Fatec Araras

Linha 99
O retorno da função pop invocado no *main* retorna o valor da variável *temp*





Pilha (stack)

heap

Implementando pilha com Python

Prof. Me. Orlando Saraiva Júnior

Pilha com Python

Podemos implementar uma pilha facilmente armazenando seus elementos em uma lista interna. Python possui tipos *built-in* para tipos sequencia.

A o tipo ***list*** já suporta a adição de um elemento ao final com o método ***append***, e removendo o último elemento com o método ***pop***.

Observe a implementação proposta, comparando-o com a linguagem C.

Fatec
Araras

Dúvidas

Prof. Orlando Saraiva Júnior
orlando.saraiva@unesp.br

Fechamento

Exercício 1 em C

Fatec Araras

Estacionamento

Um estacionamento possui uma única entrada/saída e a capacidade máxima para guardar dez carros. Faça um programa que simule a entrada (empilhar) e saída (desempilhar) de carros deste estacionamento.

Armazene a placa, modelo e ano de cada carro estacionado. A entrada de dados deve ocorrer dentro da função de empilhamento.

Funções esperadas: Estacionar carro

Chek-out de carro

Listar carros estacionados

Fechamento

Exercício 2 em C

Fatec Araras

Histórico do Browser

Ao navegar na internet, a cada site acessado, o endereço é armazenado em uma pilha. Ao clicar em voltar, pode-se acessar novamente o endereço anteriormente visitado.

Faça um programa que simule as funcionalidades de acessar um site e voltar. Ao empilhar, armazenar o endereço http. Ao desempilhar, retornar o endereço http. Não há limite de sites que podem ser acessados.

Armazene o endereço http. A entrada de dados deve ocorrer dentro da função de empilhamento.