Fatec
Araras

Estrutura de Dados

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

Fatec Araras

“First, solve the problem.

Then, write the code.”

John Johnson

Estrutura de Dados

Objetivo da aula

Apresentação do docente

Revisão dos conceitos apresentados anteriormente pelo professor Nilton.

Quem é o docente ?

Orlando Saraiva Júnior

Mestre em Tecnologia (FT / Unicamp, 2013)

MBA em Gestão Estratégica de Negócios (Unifian, 2008)

Tecnólogo em Informática (CESET / Unicamp, 2005)

Assistente suporte acadêmico na Universidade Estadual Paulista (UNESP / campus Rio Claro)

Docente na Fundação Hermínio Ometto (FHO / Uniararas)

Autor do livro "Introdução à Orientação a Objetos com C++ e Python" (ISBN: 978-85-7522-548-6)

Ementa e Objetivos

Fatec Araras

Ementa

Revisão dos conceitos básicos de tipos abstratos de dados. Pilhas, filas, alocação dinâmica, recursividade, listas encadeadas, tabelas de espalhamento e árvores. Aplicações das estruturas de dados em problemas computacionais.

Objetivos Gerais:

Aprofundar conhecimentos sobre criação e manipulação de tipos abstratos de dados: listas, pilhas, filas e árvores.

Objetivos Específicos:

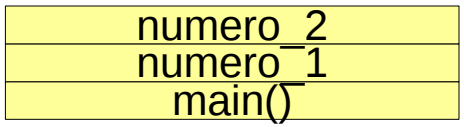
Criar, manipular e aplicar, por meio de uma linguagem de programação, os tipos abstratos de dados: listas, pilhas, filas e árvores.

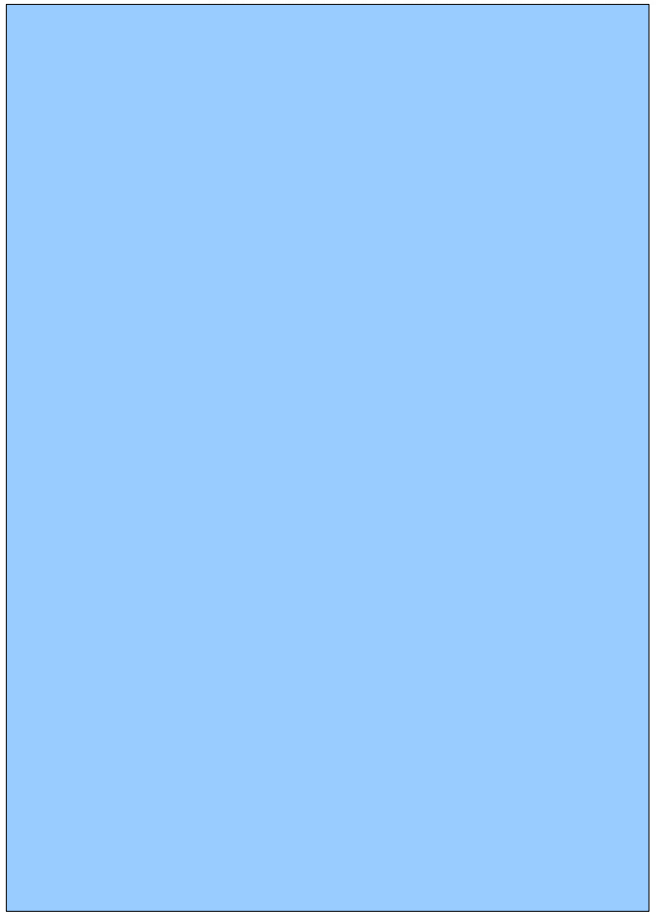
Programa 01

Prof. Me. Orlando Saraiva Júnior

Programa 01.c

debugger na linha 8

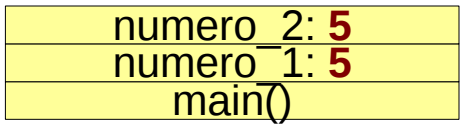




Memória estática | Pilha (stack) | heap

Programa 01.c

debugger na linha 11



Memória estática

Pilha (stack)

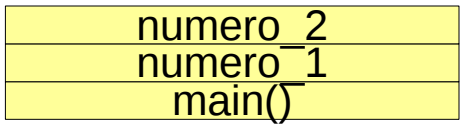
heap

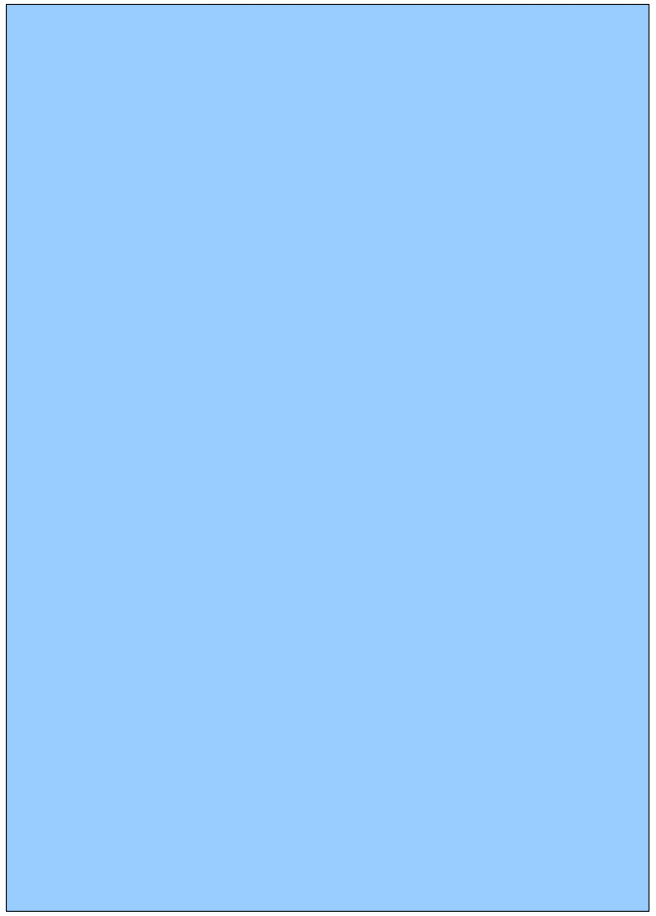
Programa 02

Prof. Me. Orlando Saraiva Júnior

Programa 02.c

debugger na linha 27

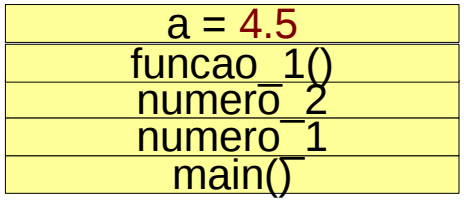


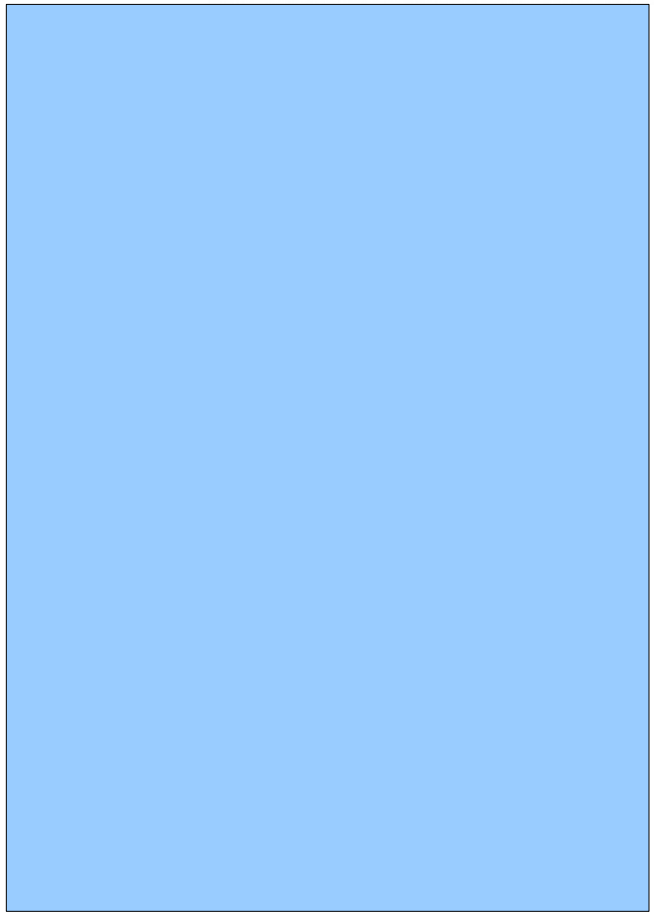


Memória estática | Pilha (stack) | heap

Programa 02.c

debugger na linha 18

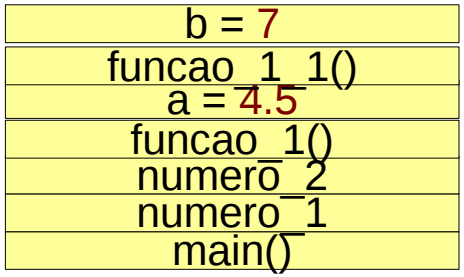


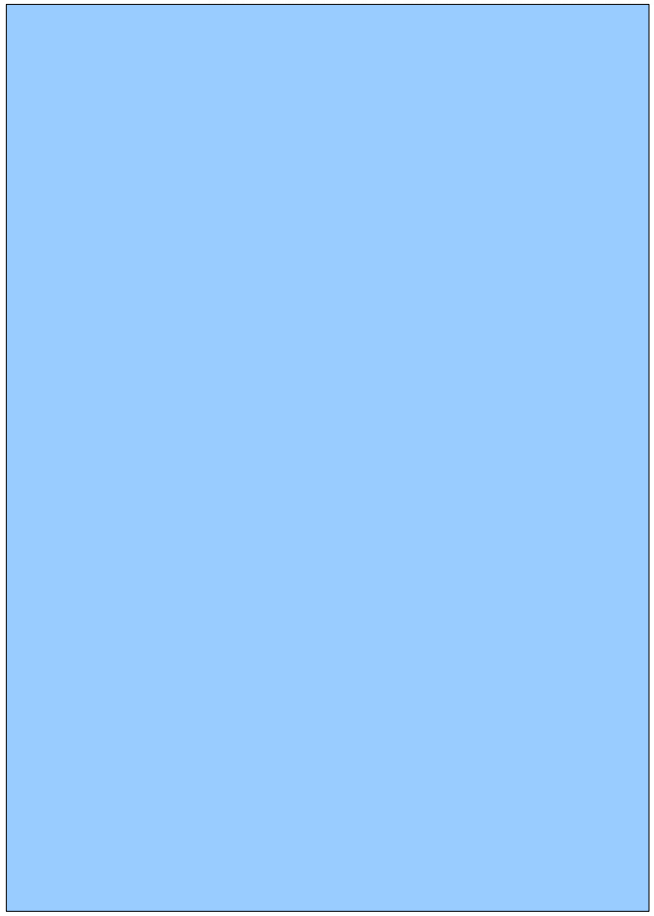


Memória estática | Pilha (stack) | heap

Programa 02.c

debugger na linha 12

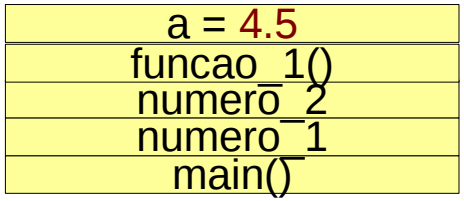


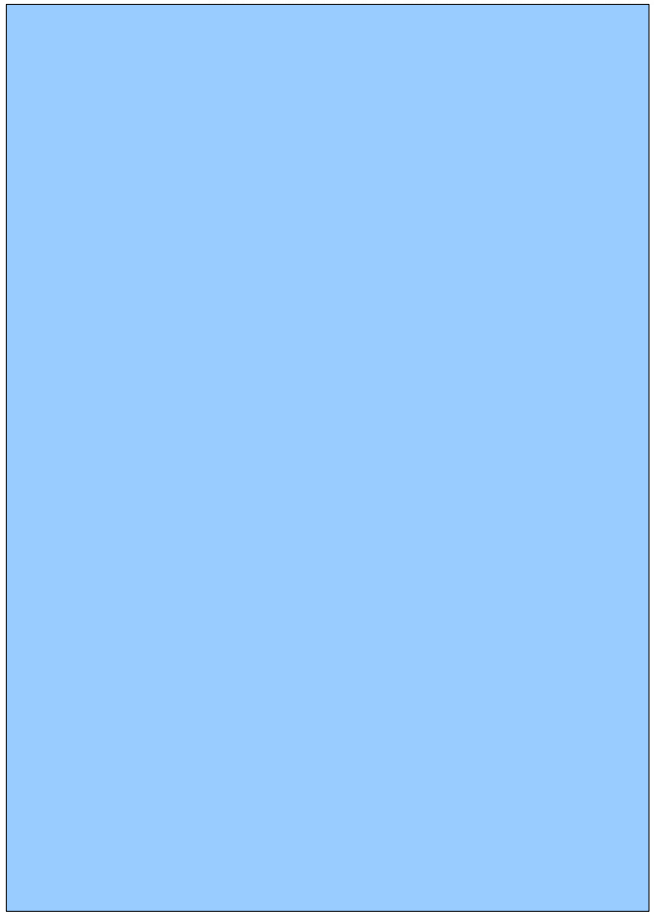


Memória estática | Pilha (stack) | heap

Programa 02.c

debugger na linha 20

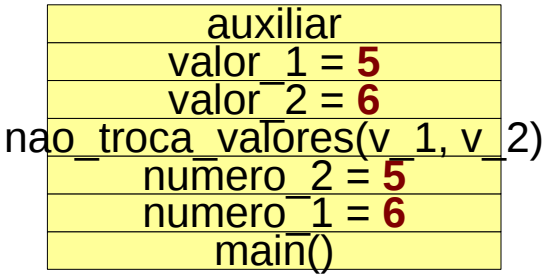


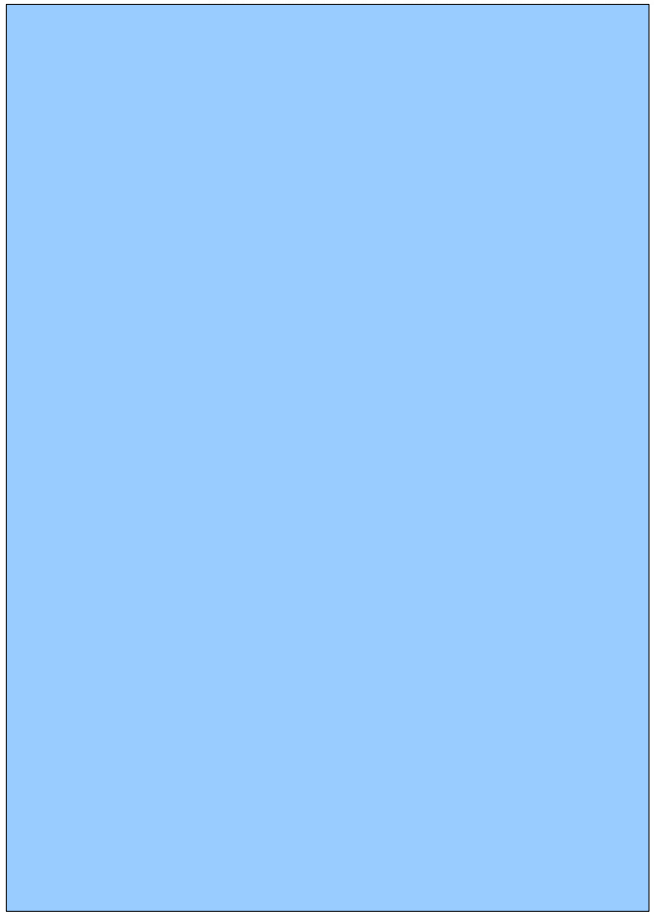


Memória estática | Pilha (stack) | heap

Programa 02.c

debugger na linha 49 (entrou na função na linha 38)

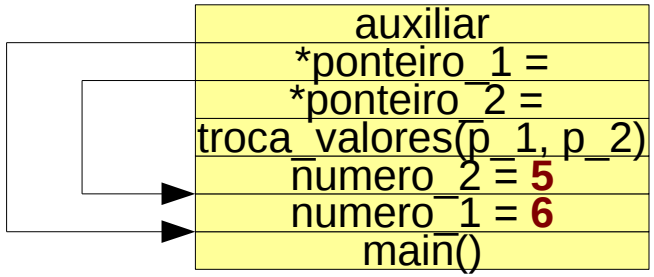




Memória estática | Pilha (stack) | heap

Programa 02.c

debugger na linha 58 (entrou na função na linha 41)



Memória estática

Pilha (stack)

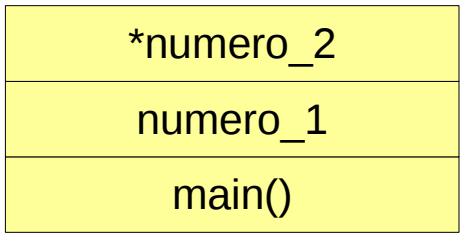
heap

Programa 03

Prof. Me. Orlando Saraiva Júnior

Programa 03.c

debugger na linha 9



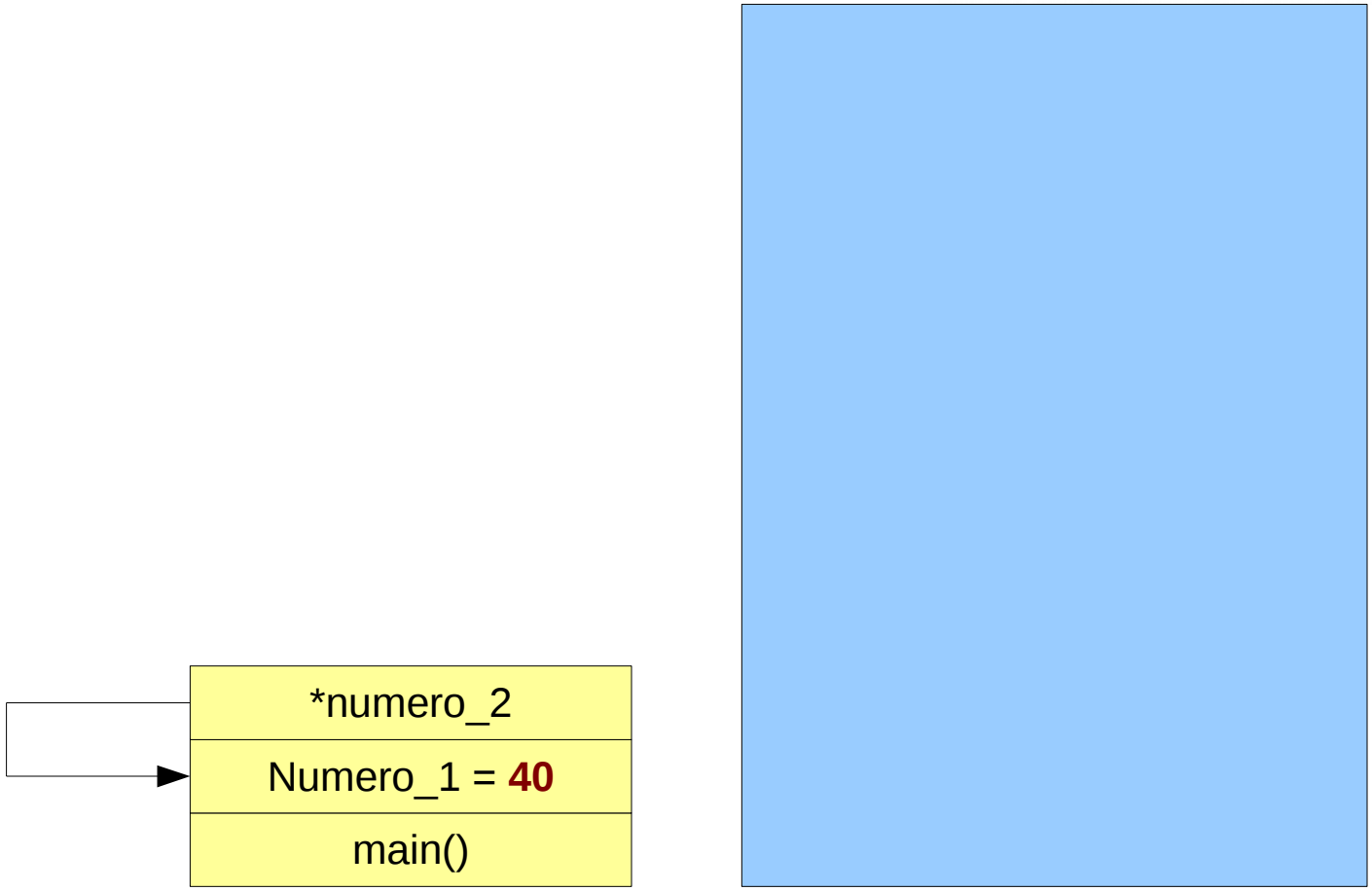
Memória estática

Pilha (stack)

heap

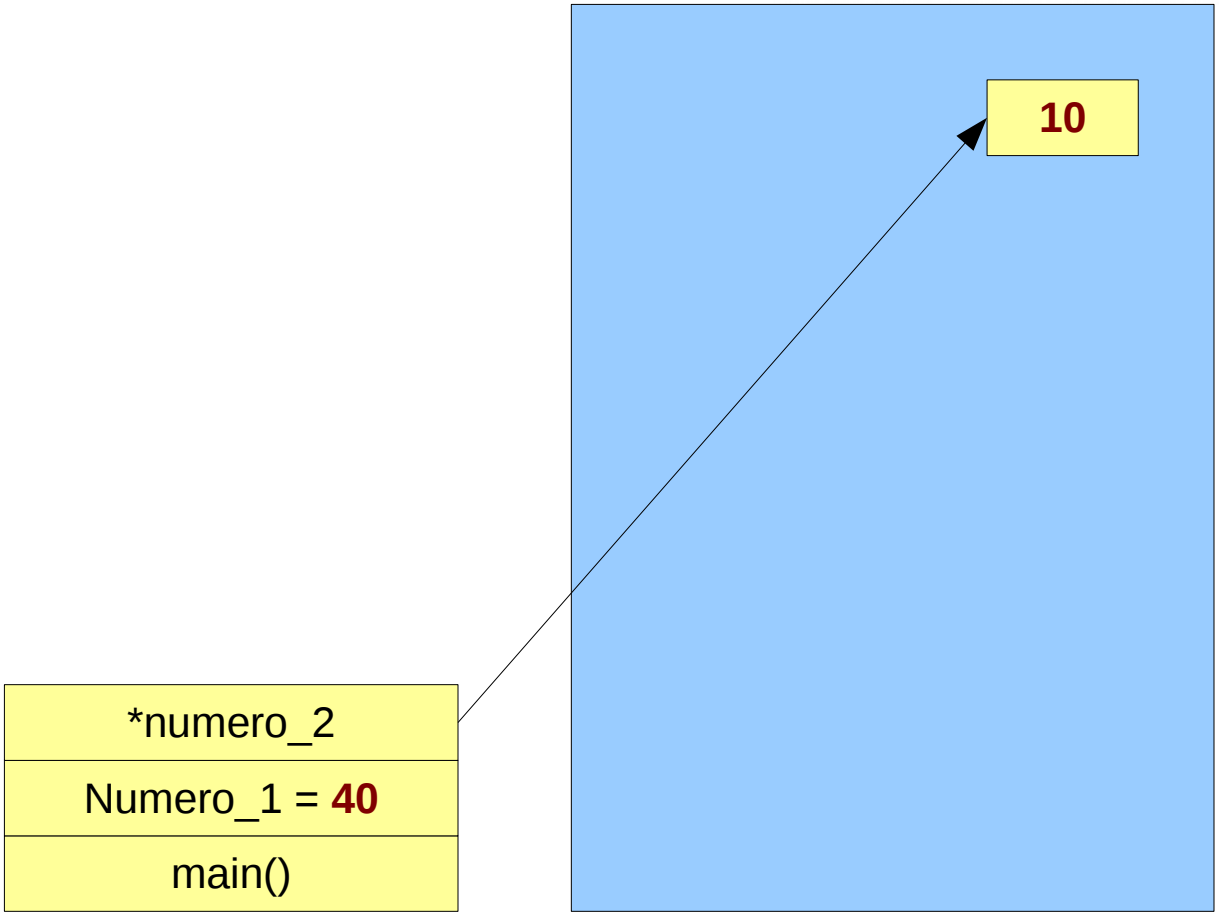
Programa 03.c

debugger na linha 12



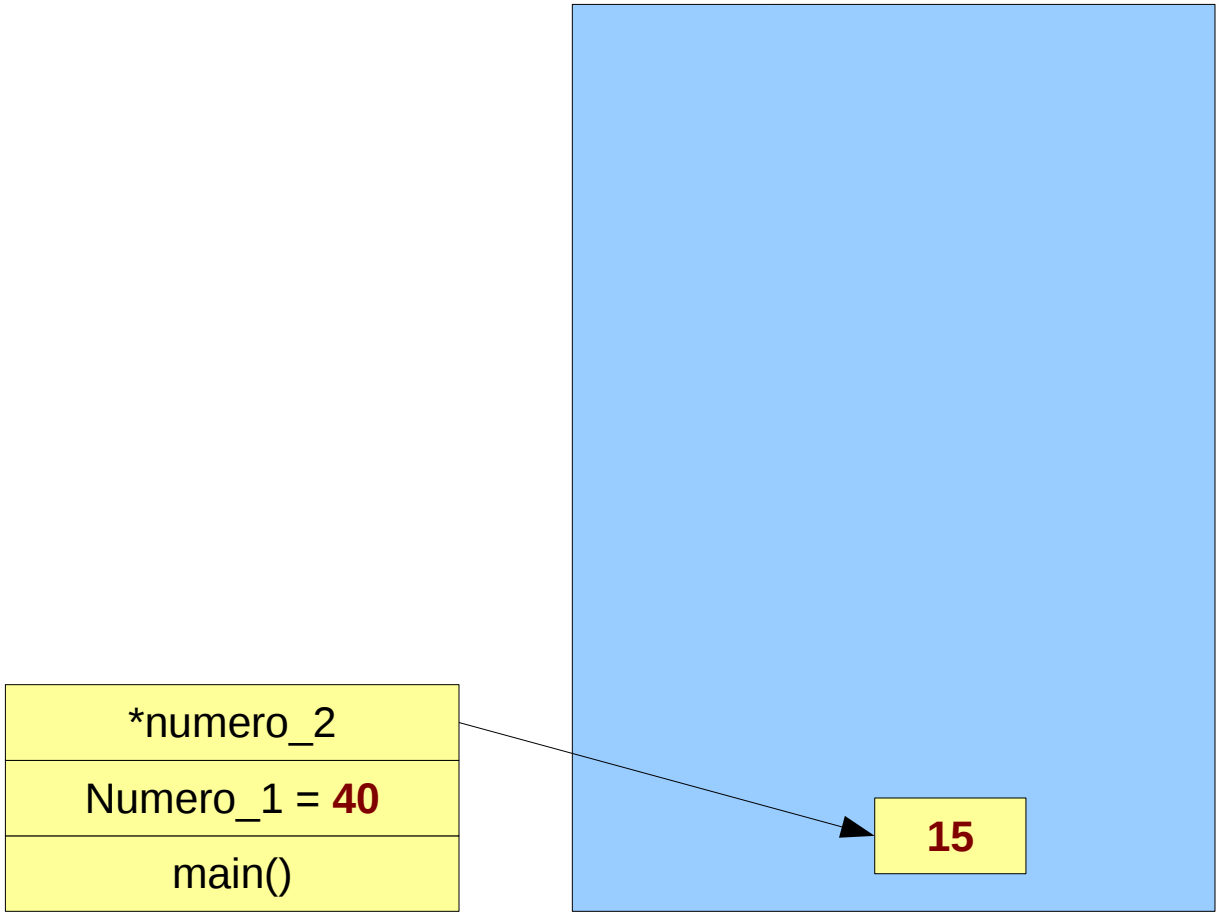
Programa 03.c

debugger na linha 16



Programa 03.c

debugger na linha 21

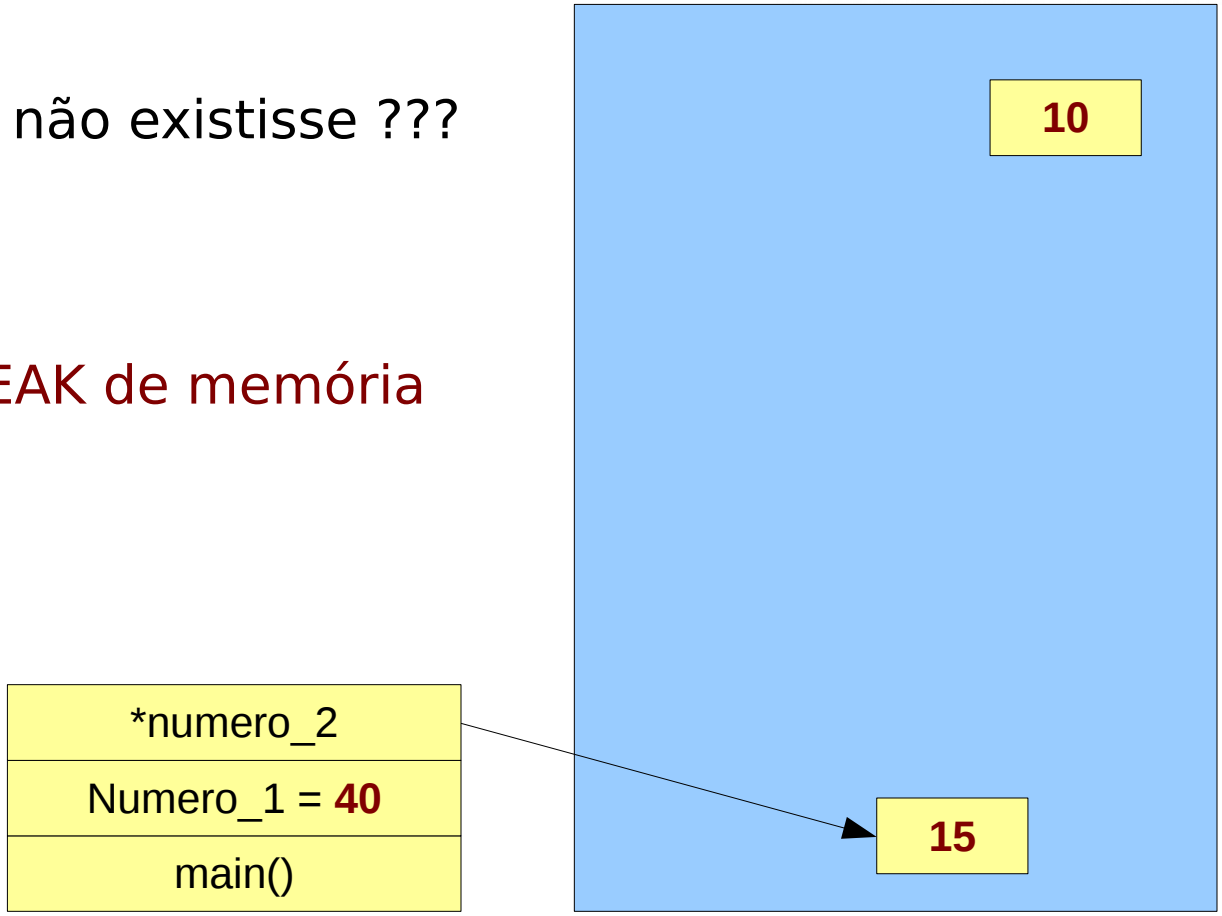


Programa 03.c

debugger na linha 21

E se o free(), linha 18 não existisse ???

LEAK de memória

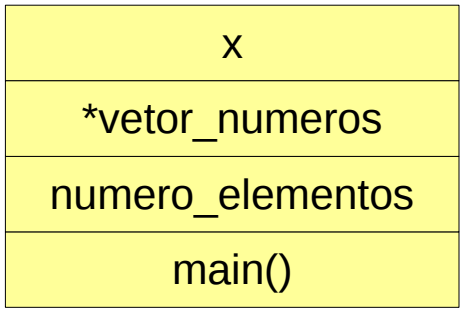


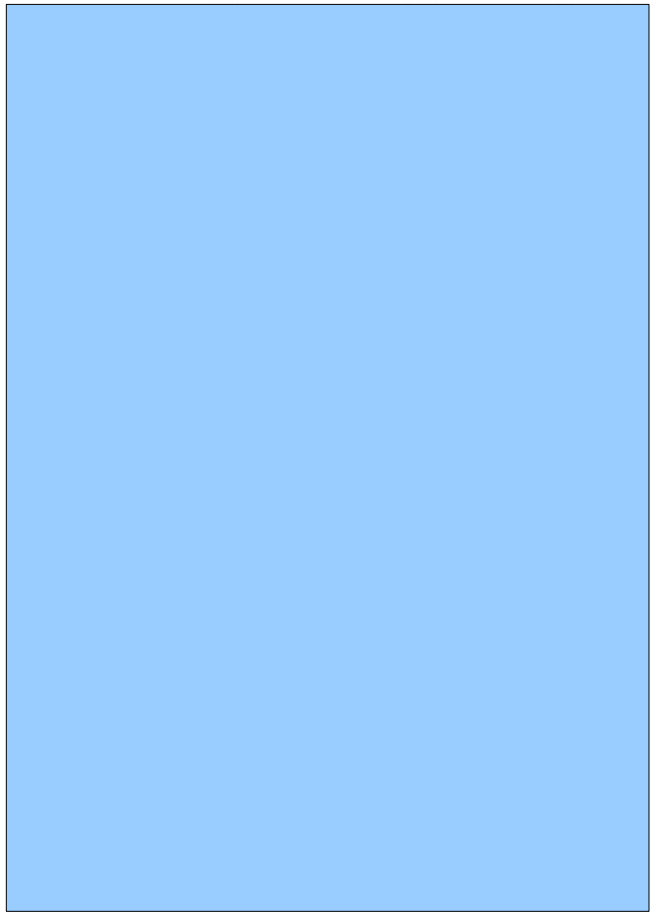
Programa 04

Prof. Me. Orlando Saraiva Júnior

Programa 04.c

debugger na linha 11

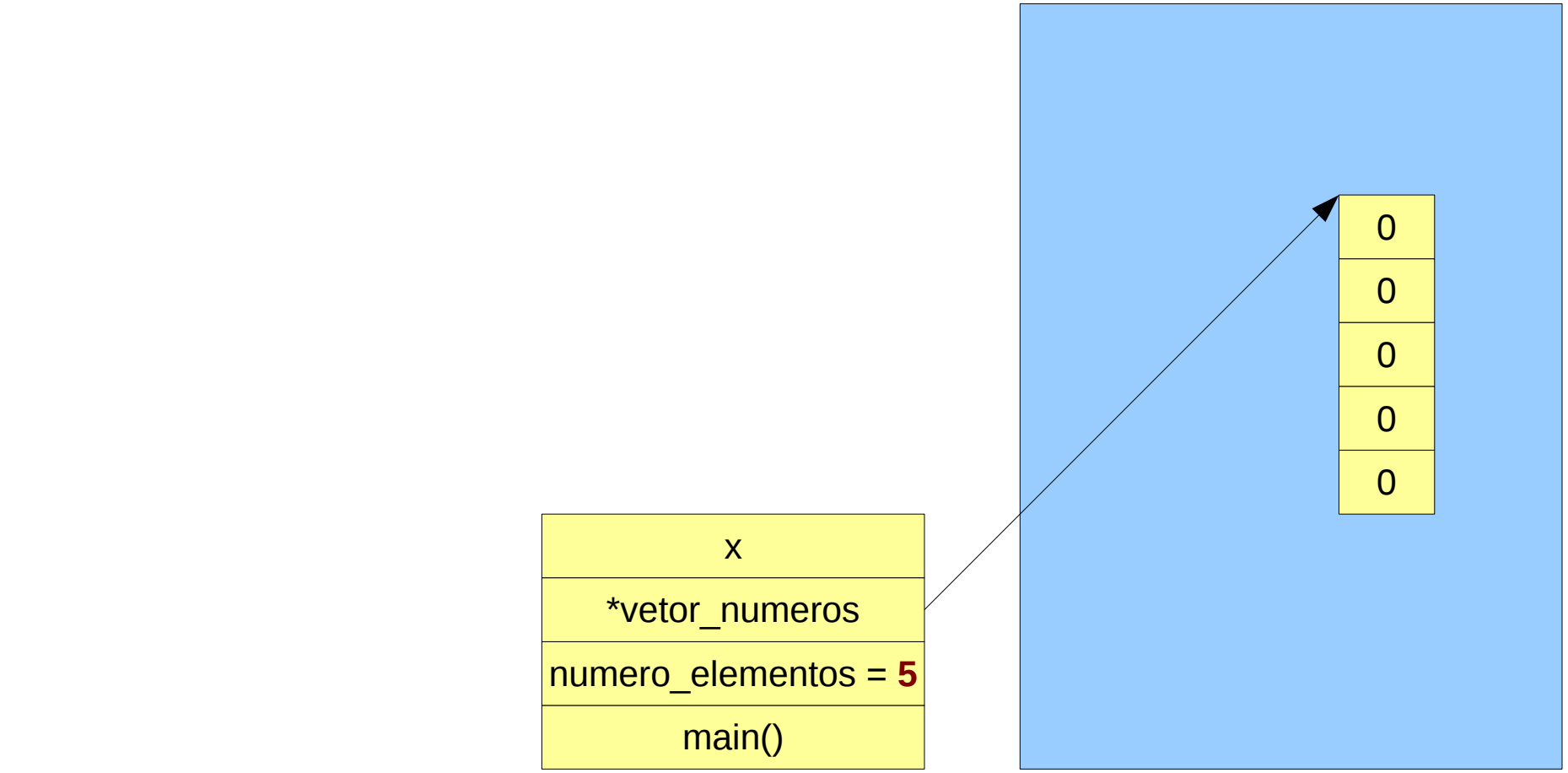




Memória estática | Pilha (stack) | heap

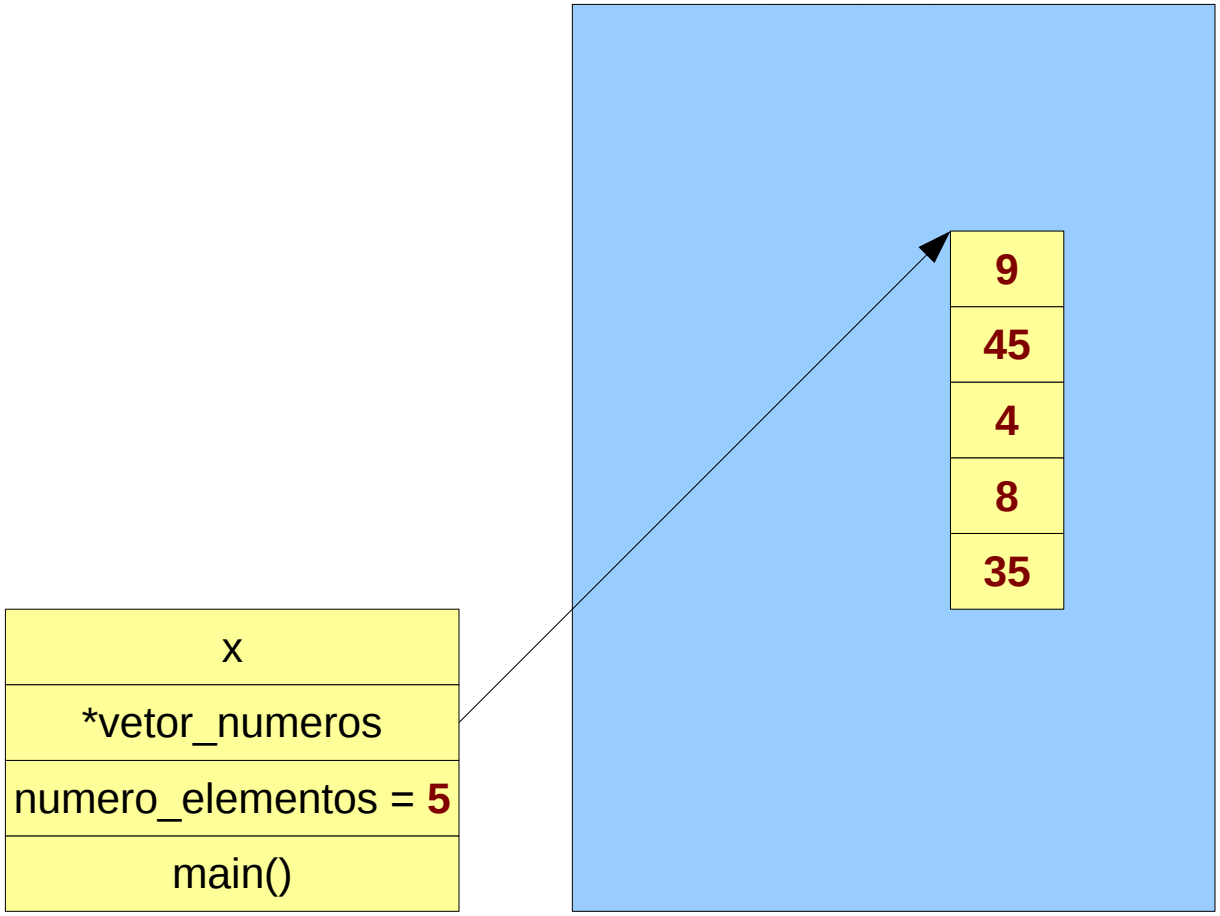
Programa 04.c

debugger na linha 14



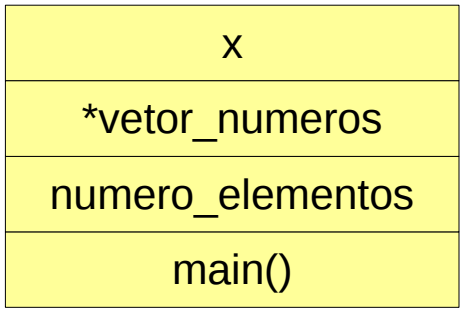
Programa 04.c

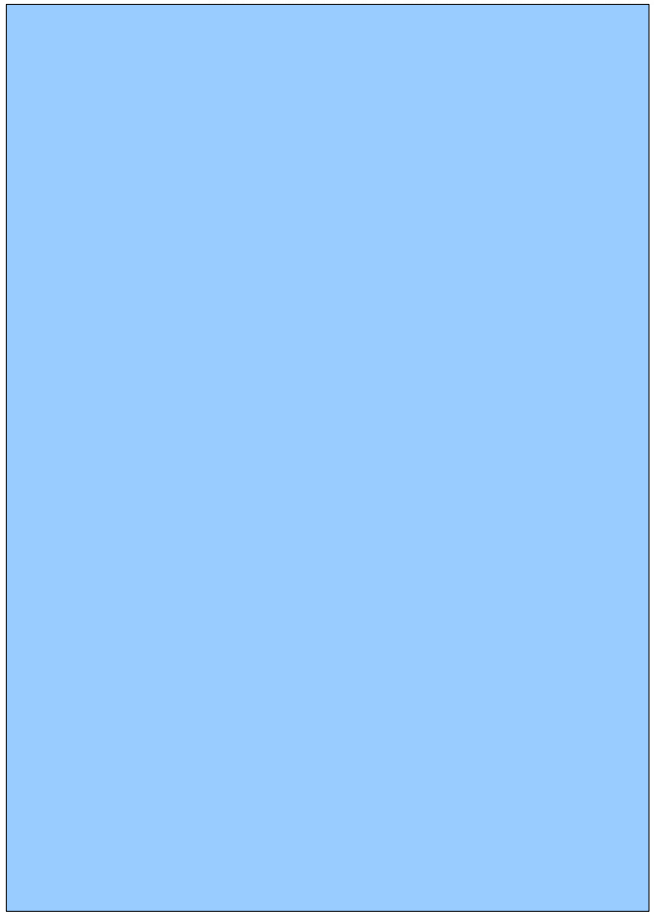
debugger na linha 20



Programa 04.c

debugger na linha 25





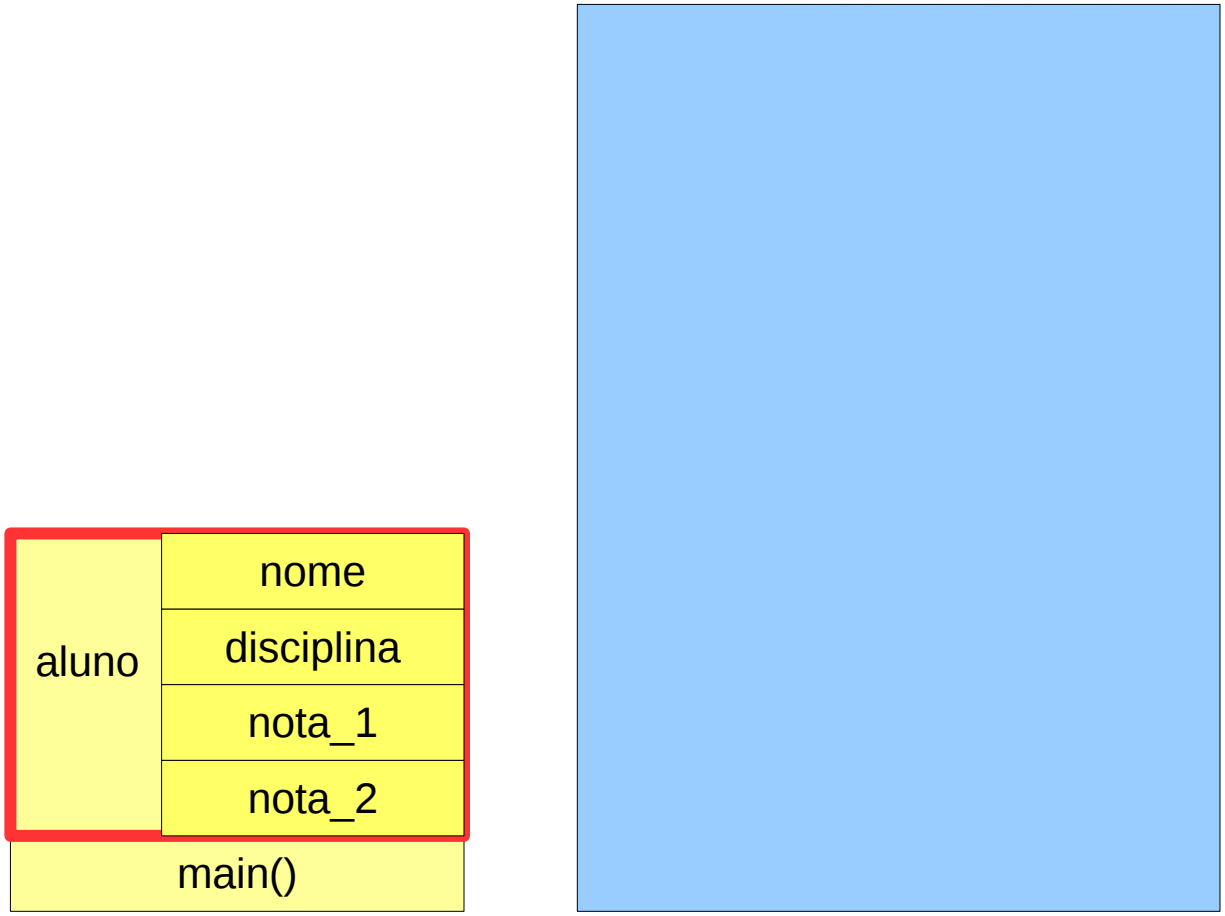
Memória estática | Pilha (stack) | heap

Programa 05

Prof. Me. Orlando Saraiva Júnior

Programa 05.c

debugger na linha 15

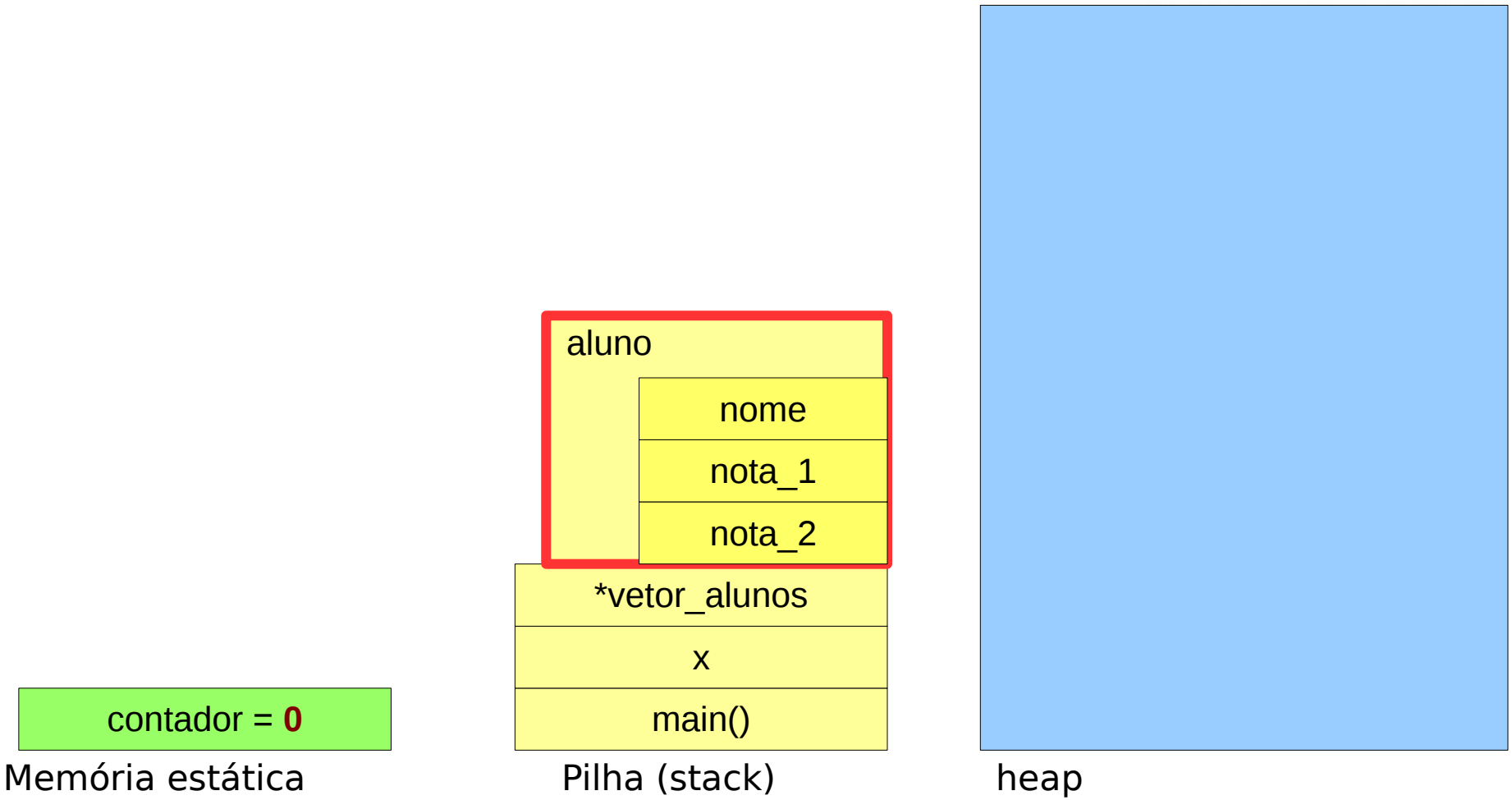


Programa 06

Prof. Me. Orlando Saraiva Júnior

Programa 06.c

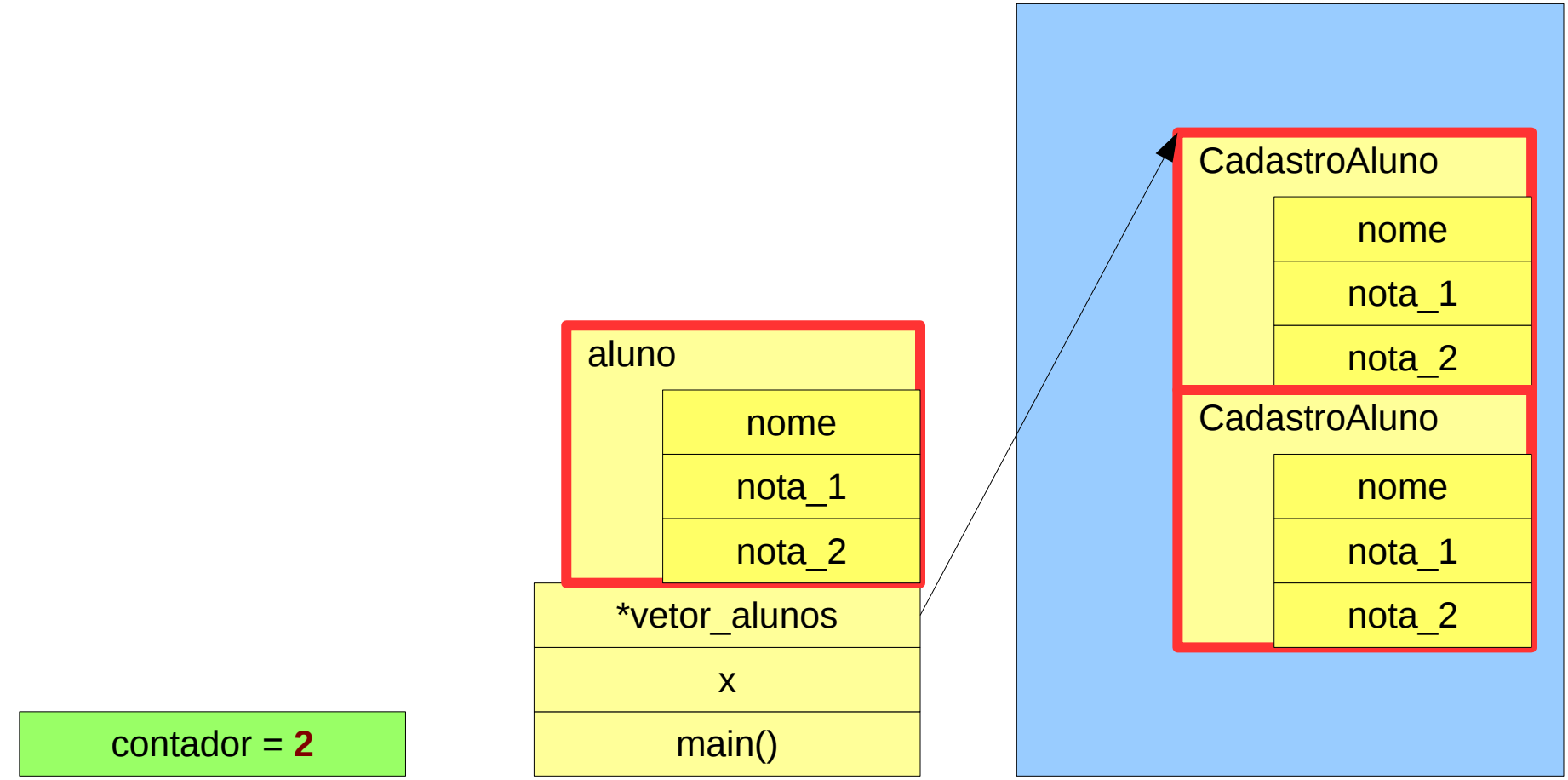
debugger na linha 19



Programa 06.c

debugger na linha 24

Fatec Araras



Fatec
Araras

Dúvidas

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

Fechamento

Programa 02: Altere o programa para que um estouro de pilha ocorra.

Programa 03: Altere o programa e teste ponteiros com outros tipos primitivos (float, double, por exemplo)

Programa 04: Altere o programa, para que preencha os valores do vetor automaticamente. Após esta alteração, teste o programa com diversos parâmetros. Tente causar um erro de execução.

Programa 06: Altere o programa para que o CadastroAluno seja mais sofisticado, permitindo armazenar mais informações do aluno. Faça uso de struct dentro de struct, se for necessário.