Programação Estruturada

Prof. Rodrigo Hausen
http://progest.compscinet.org

Listas Ligadas
(Listas Encadeadas)

PILHA versus FILA

Uma **pilha** é uma estrutura de dados onde, de modo geral, acessamos primeiramente o **último dado** que foi inserido.

LIFO = **L**ast **I**n, **F**irst **O**ut
Último a entrar, primeiro a sair

Uma **fila** é uma estrutura de dados onde, de modo geral, acessamos primeiramente o **primeiro dado** que foi inserido.

FIFO = **F**irst **I**n, **F**irst **O**ut
Primeiro a entrar, primeiro a sair

PILHA versus FILA

Podemos implementar pilhas e filas usando **arrays** (vimos como implementar uma pilha na aula passada).

Com arrays, para transformarmos a estrutura em **dinâmica** (cresce conforme os dados são recebidos), temos que alocar um novo array e copiar todos os dados para essa nova região de memória.

Podemos criar pilhas e filas dinâmicas sem realocação nem cópia de dados, usando uma estrutura chamada **lista ligada** (ou **lista encadeada**).

LISTA LIGADA (linked list)

Uma lista ligada é composta por **nós**. Cada nó armazena:

- **dado** relacionado a ele; e
- endereço (ponteiro) para **próximo** nó na lista.

LISTA LIGADA (linked list)

Uma lista ligada é composta por **nós**. Cada nó armazena:

- **dado** relacionado a ele; e
- endereço (ponteiro) para **próximo** nó na lista.

Definindo como struct:

```
struct NoLista {
  tipo dado;
  struct NoLista *prox;
};
```

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

struct NoInt *inicio = NULL;
```

(lista vazia)

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = NULL;
```

(lista vazia)

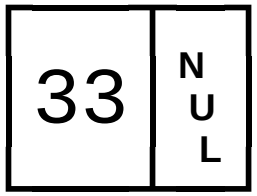
inserir 33

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = NULL;
```



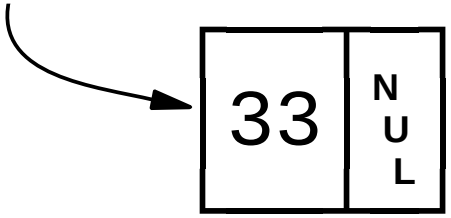
inserir 33

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = NULL;
```



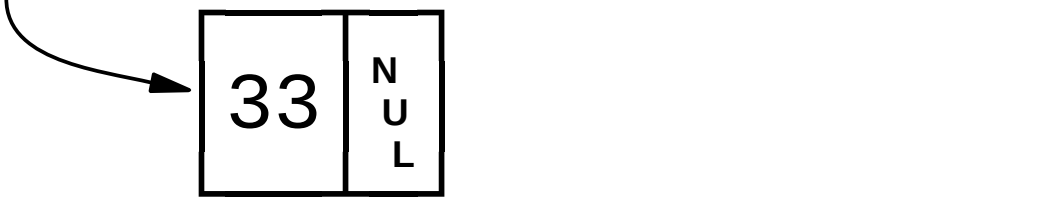
inserir 33

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



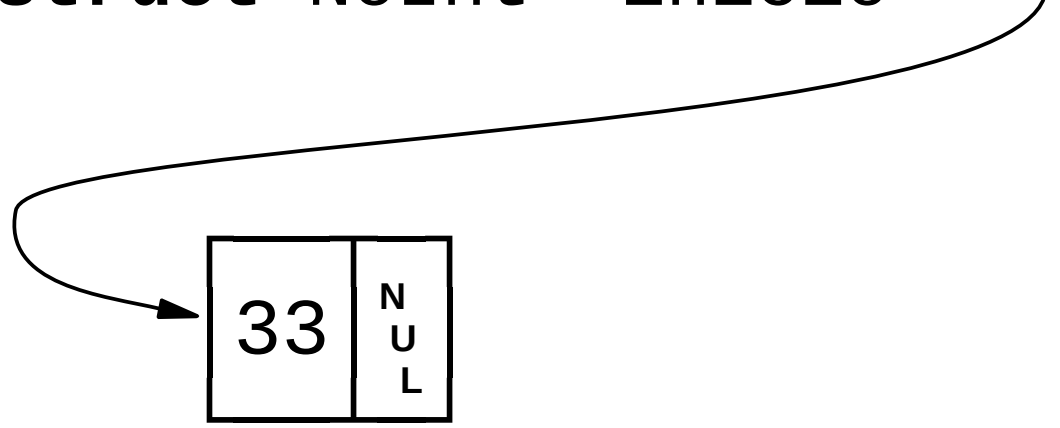
inserir 33

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



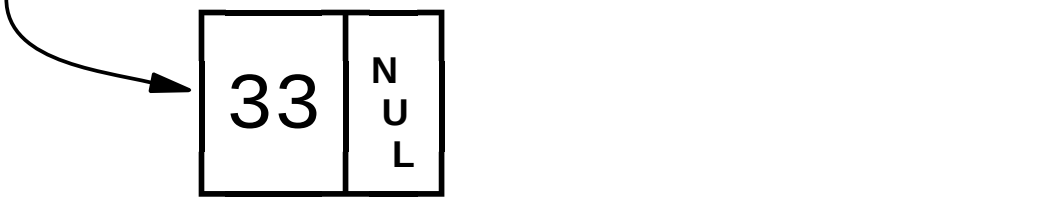
33 inserido

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



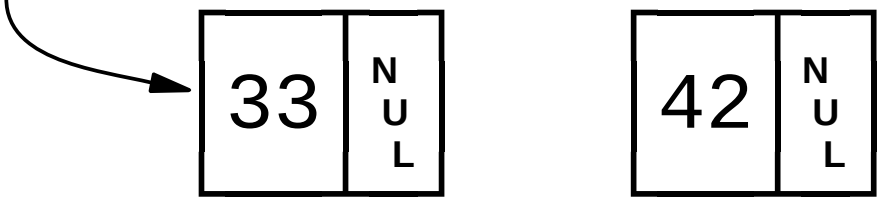
inserir 42 no final

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



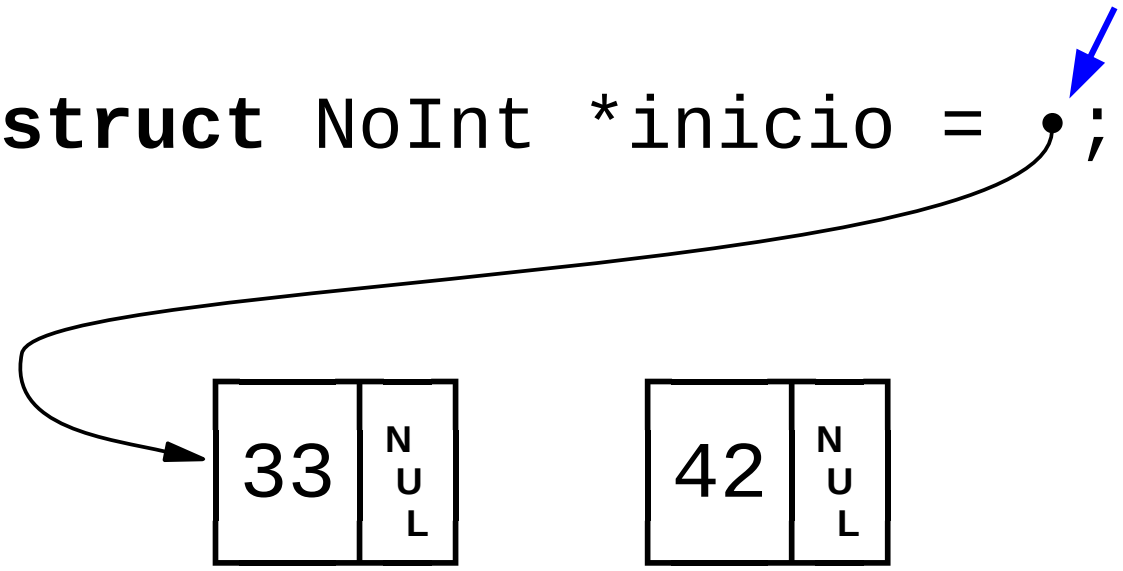
inserir 42 no final

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



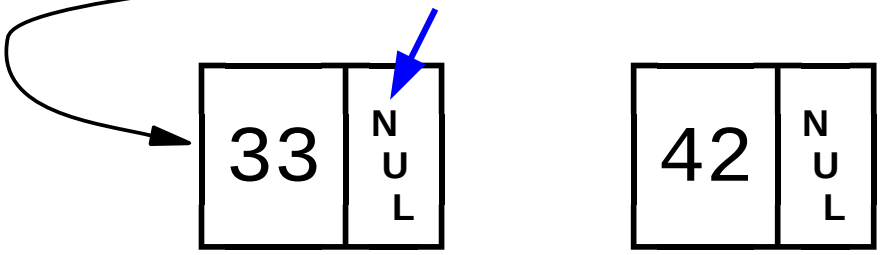
inserir 42 no final

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



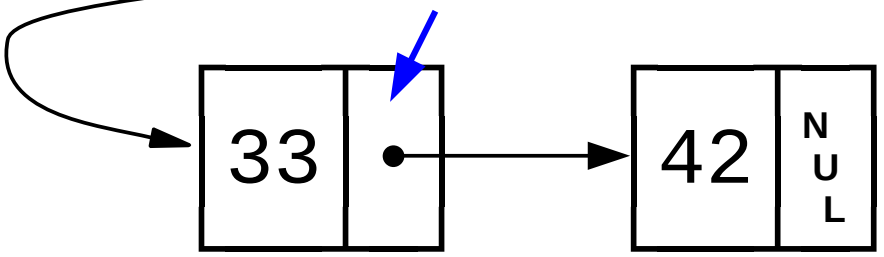
inserir 42 no final

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



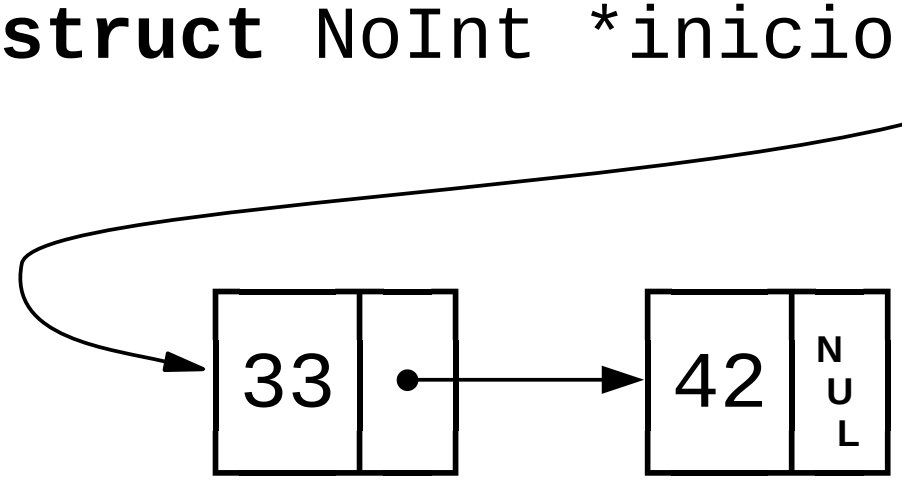
inserir 42 no final

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



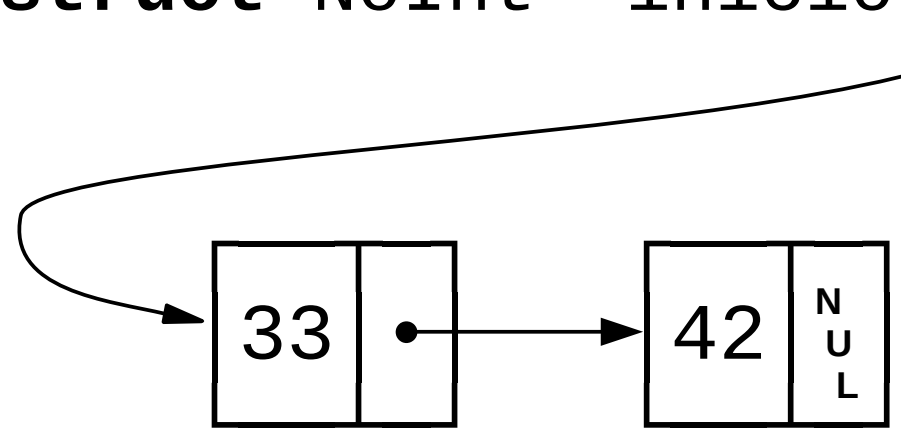
42 inserido no final

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



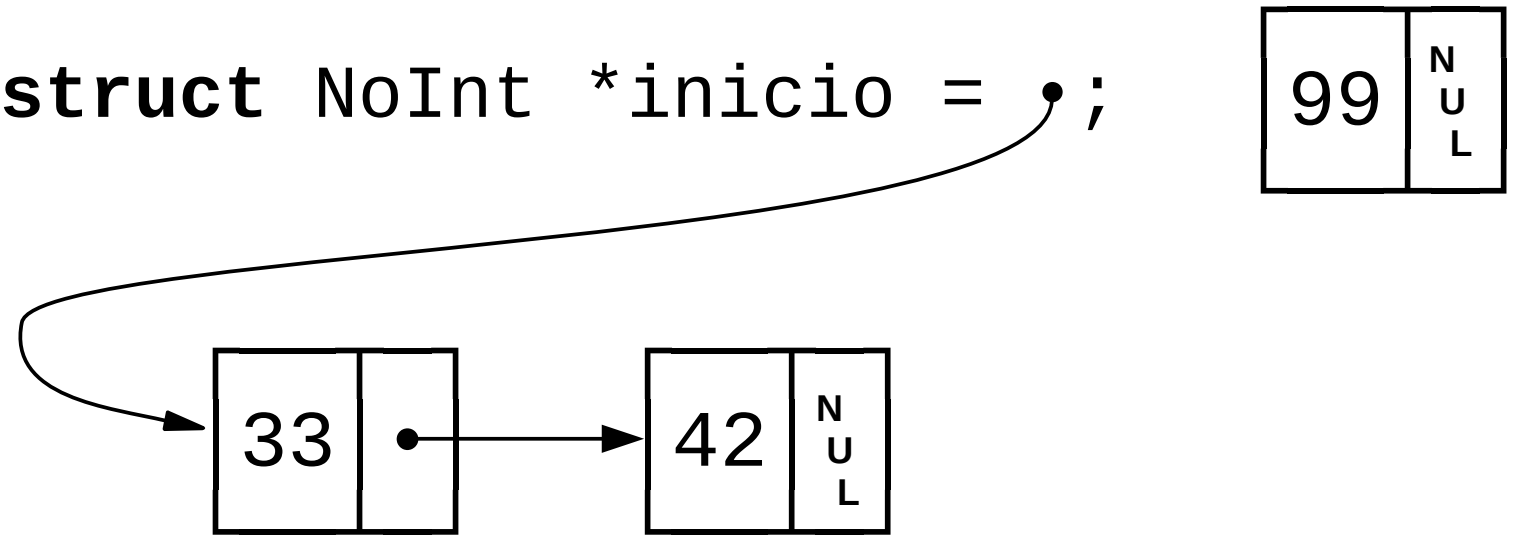
inserir 99 no início

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



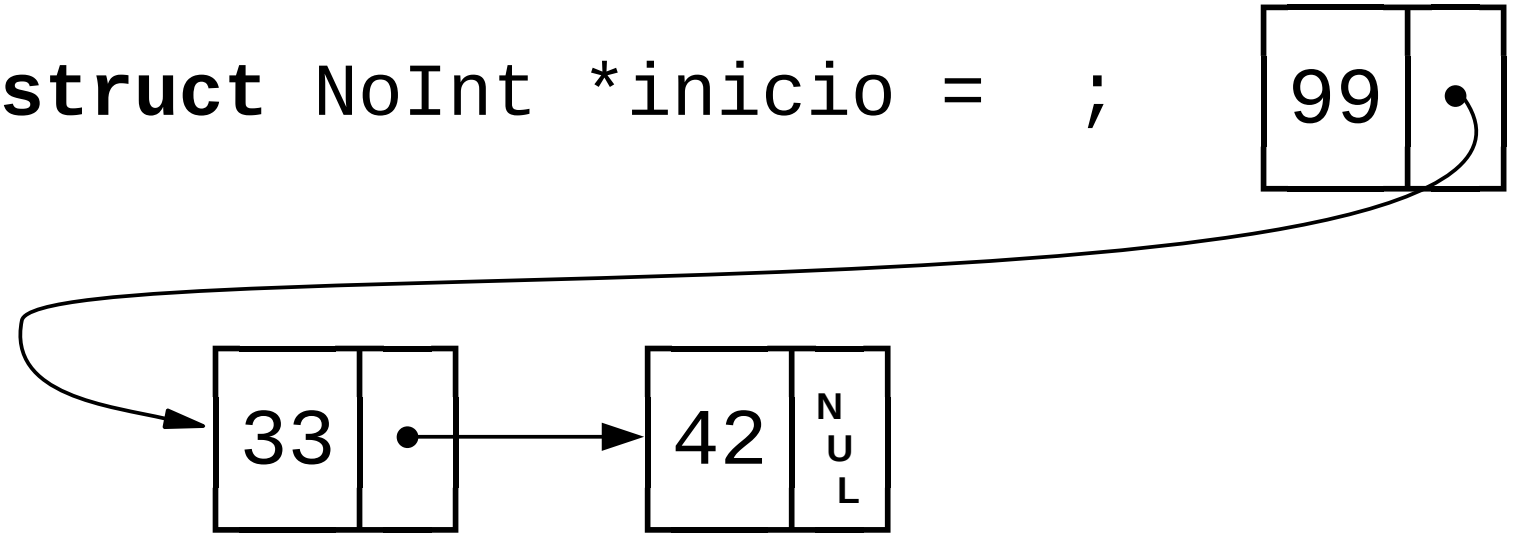
inserir 99 no início

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio =  ;
```



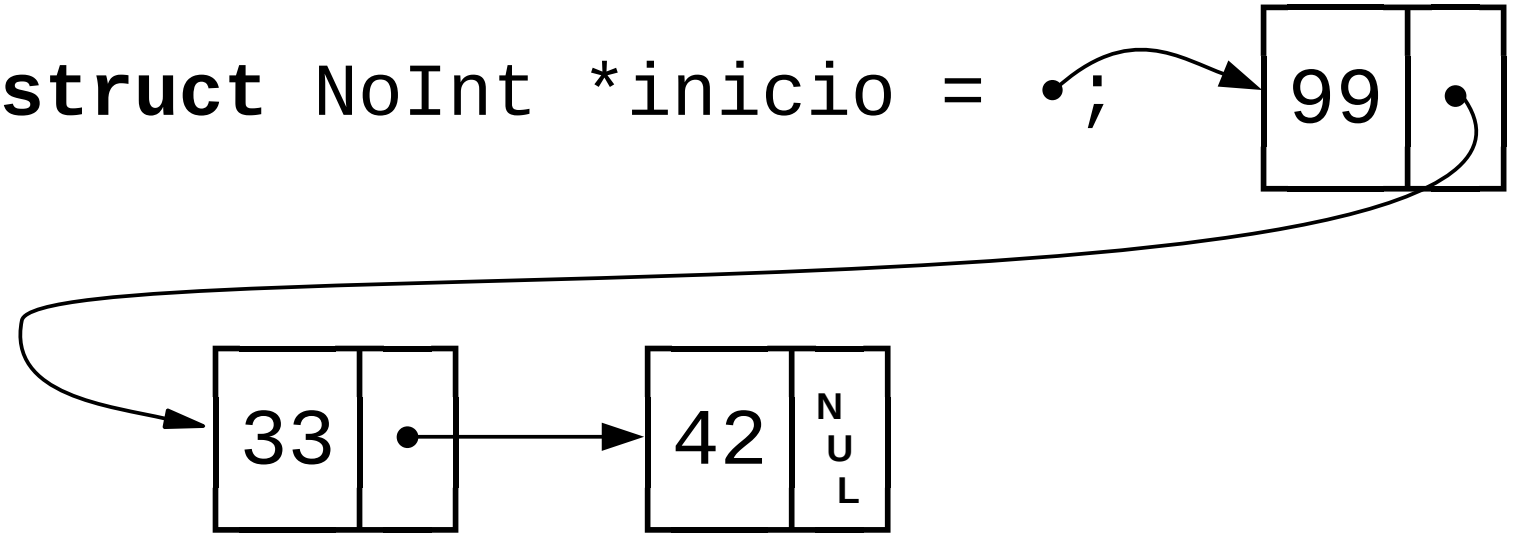
inserir 99 no início

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



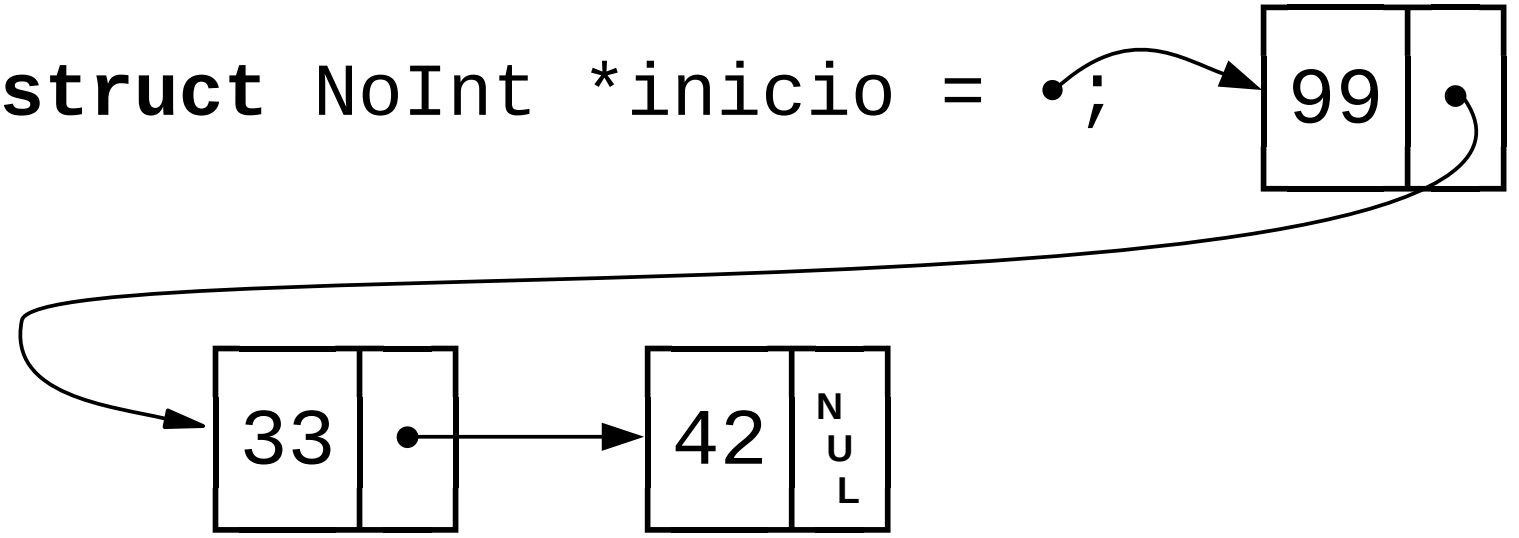
inserir 99 no início

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



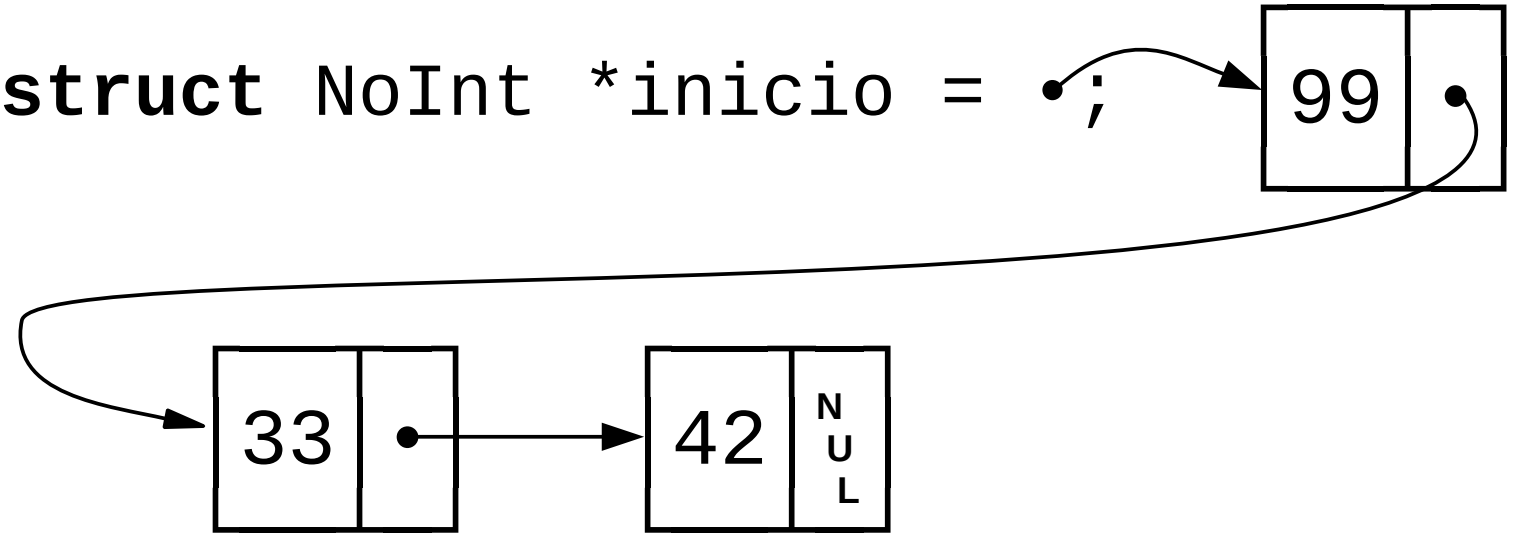
99 inserido no início

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



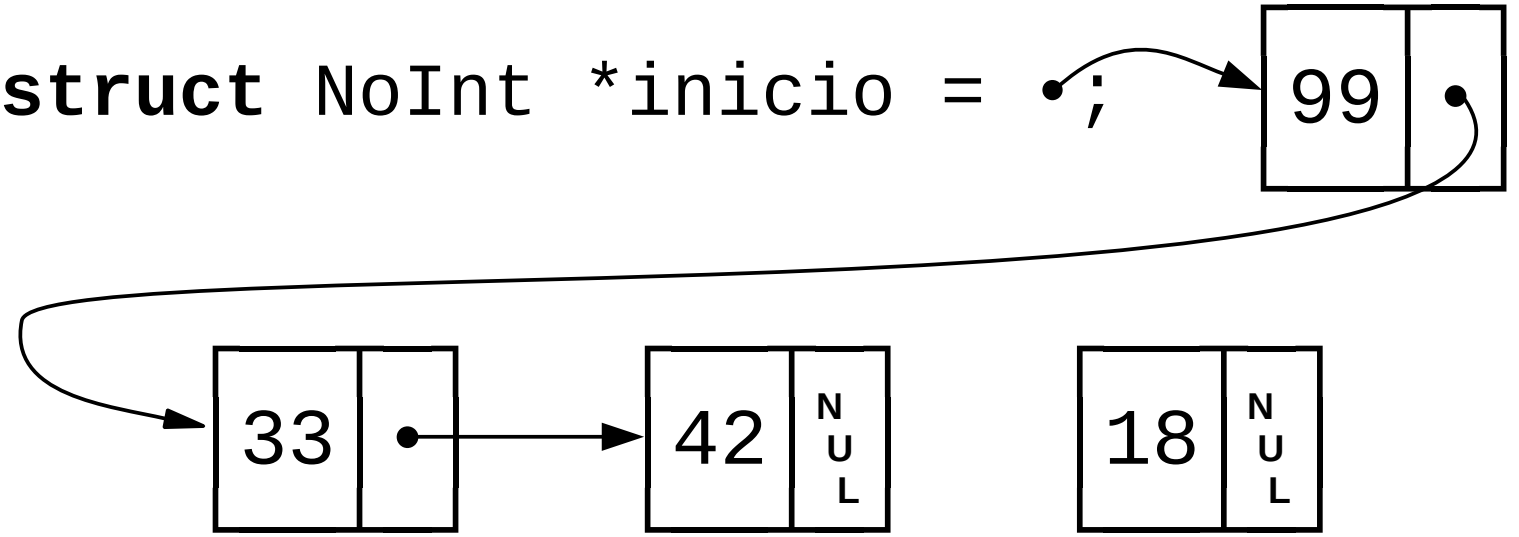
insere 18 no final

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



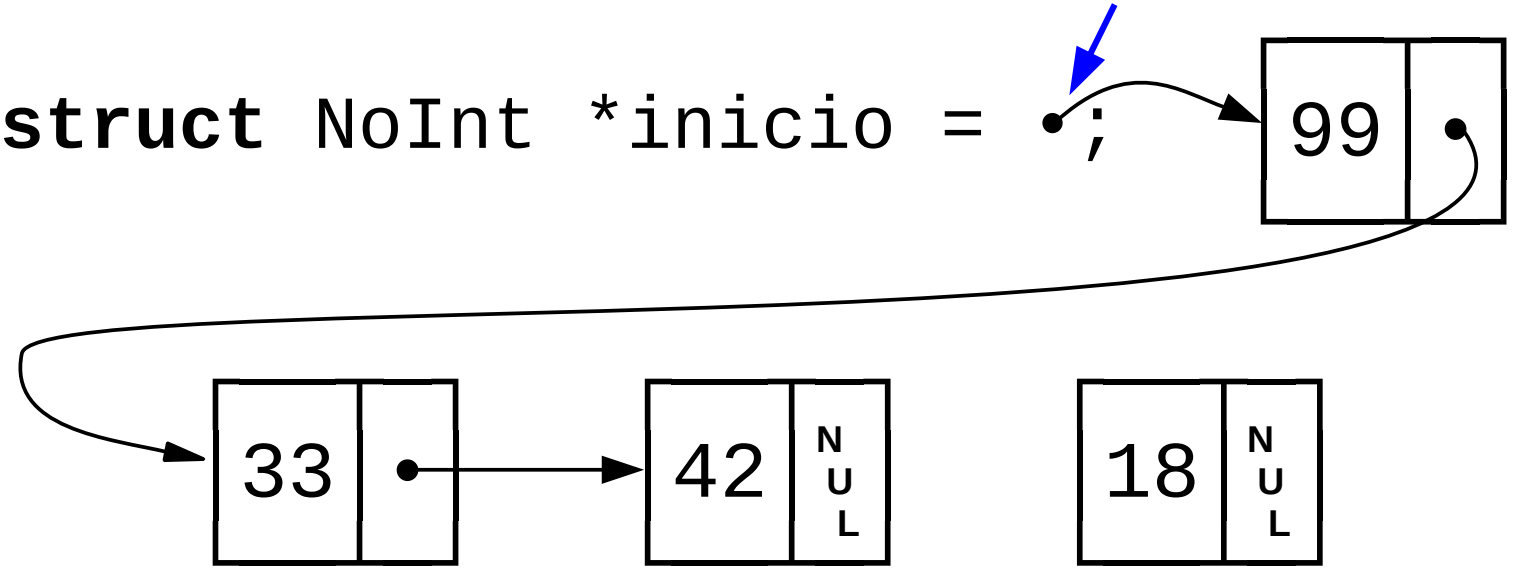
insere 18 no final

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



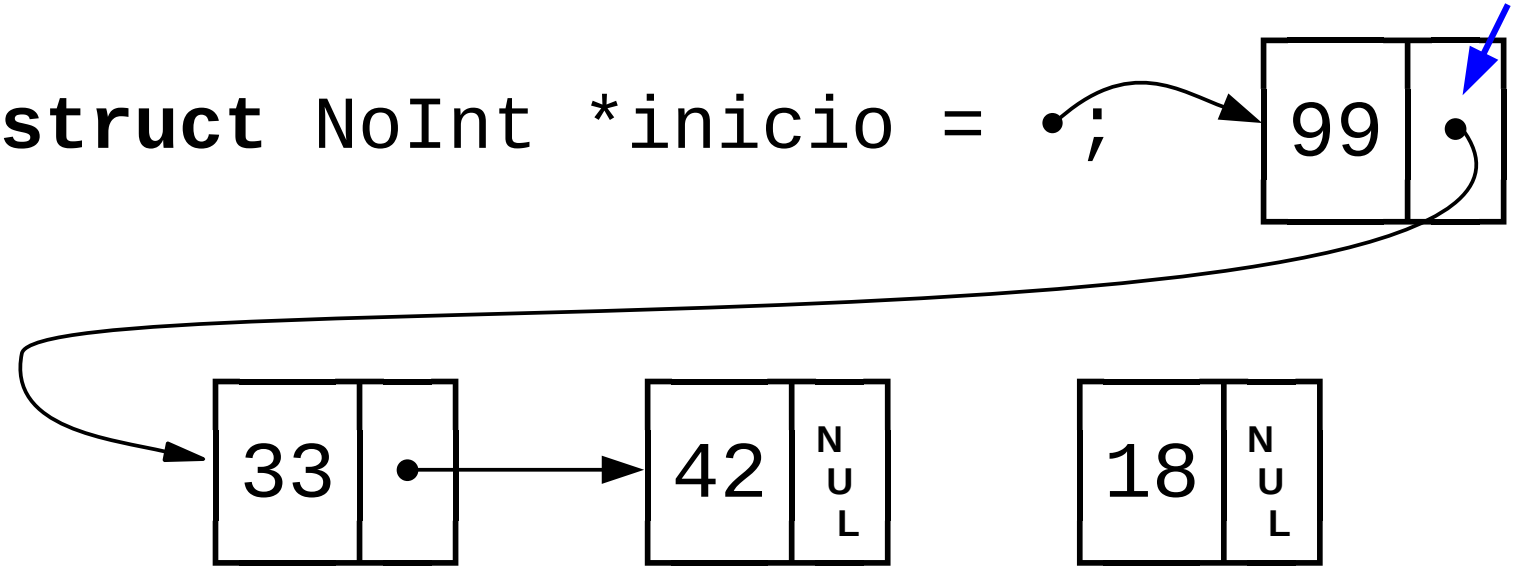
insere 18 no final

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



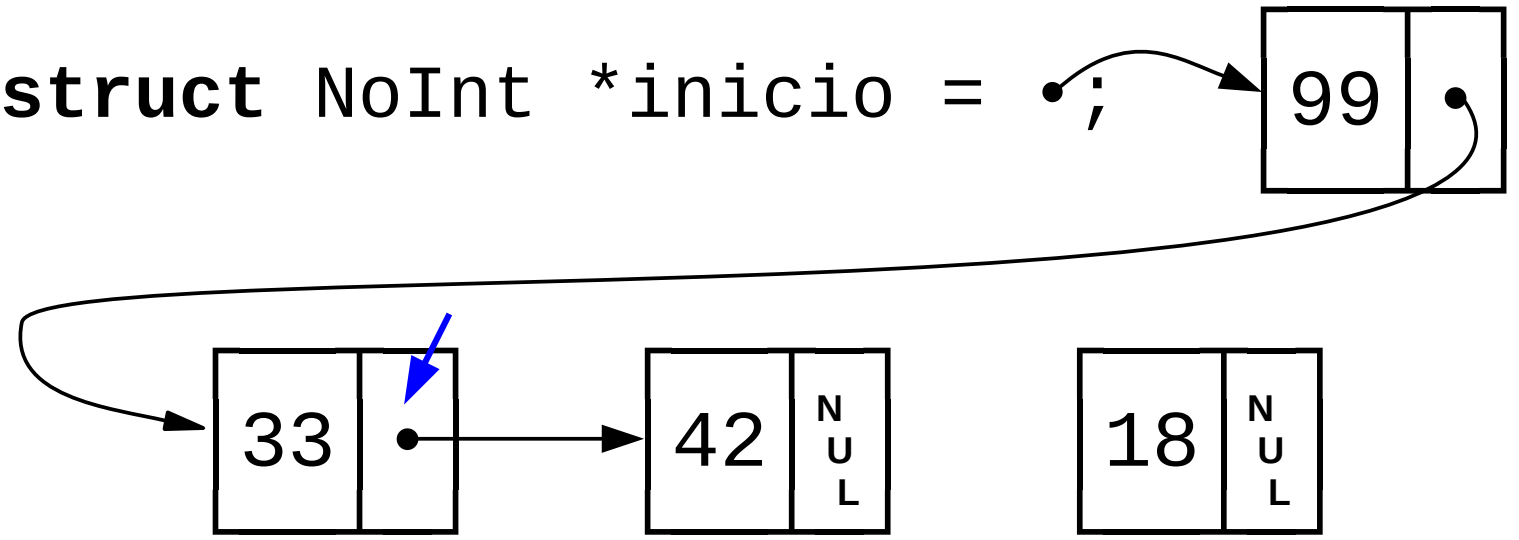
insere 18 no final

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



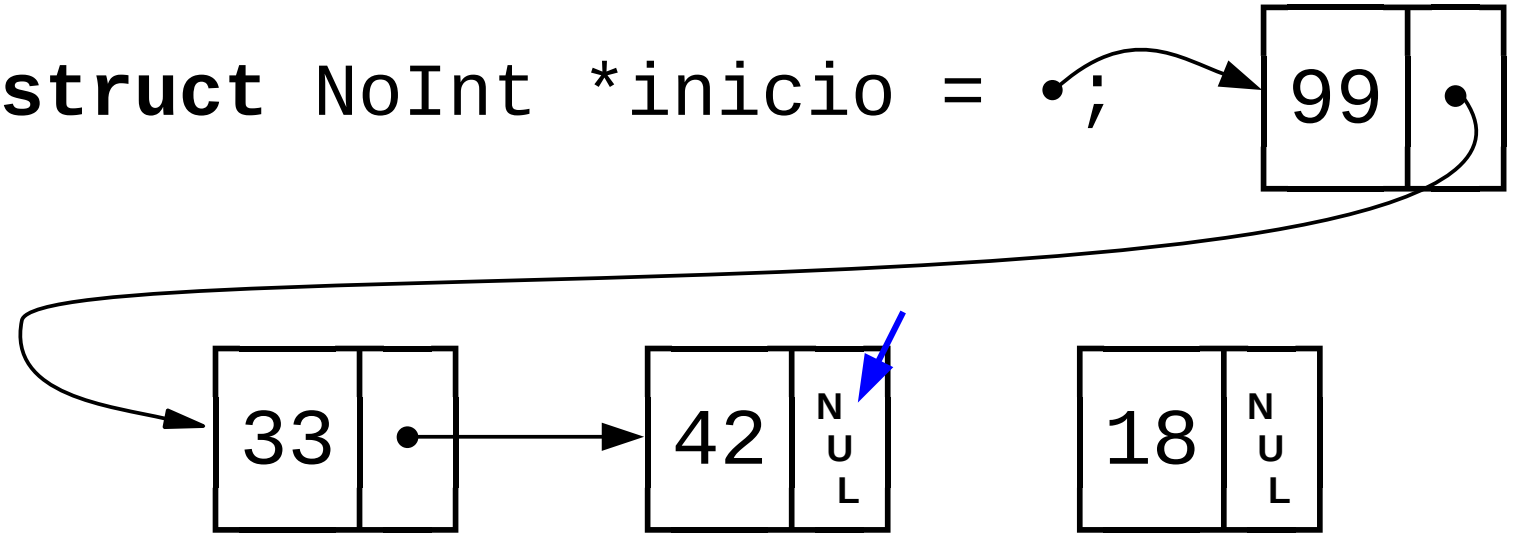
insere 18 no final

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



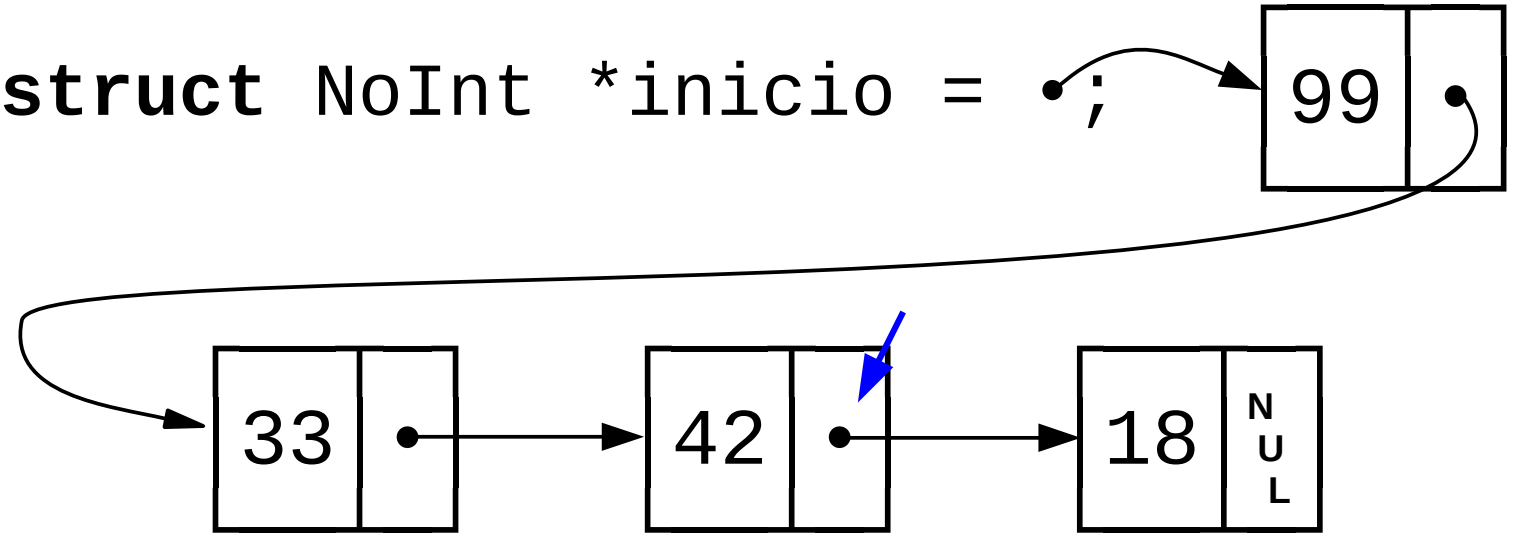
insere 18 no final

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



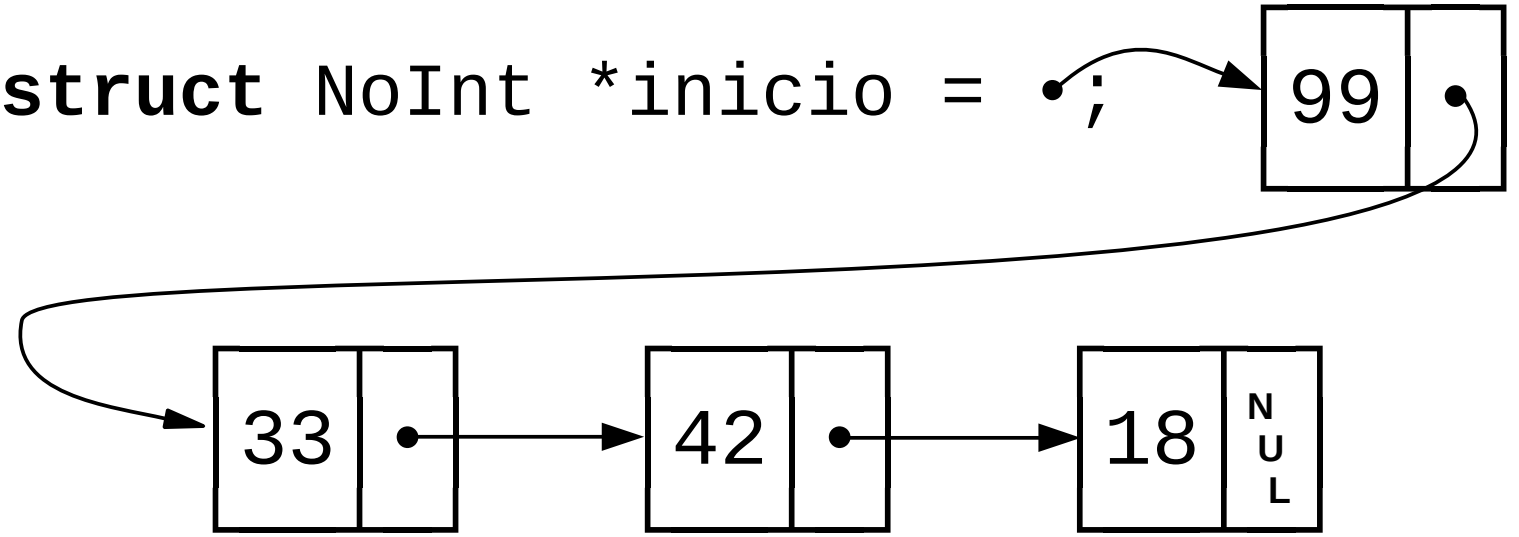
insere 18 no final

LISTA LIGADA (linked list)

Exemplo: lista de inteiros.

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *inicio = ;
```



Lista com elementos 99, 33, 42, 18.

LISTA LIGADA (linked list)

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};
```

O nó inicial de uma lista ligada também é chamado de **nó cabeça** desta lista.

Dado o nó cabeça de uma lista de inteiros, como podemos inserir um inteiro no início da lista?

LISTA LIGADA (linked list)

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *
noint_insere_inicio(struct NoInt *cabeca,
                    int dado) {




}
```

Retorna ponteiro para novo nó cabeça da lista.

LISTA LIGADA (linked list)

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

struct NoInt *
noint_insere_inicio(struct NoInt *cabeca,
                    int dado) {
  struct NoInt *novo = (struct NoInt *)
    malloc(sizeof(struct NoInt));
  if (novo == NULL) return cabeca;



}
```

LISTA LIGADA (linked list)

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

struct NoInt *
noint_insere_inicio(struct NoInt *cabeca,
                    int dado) {
  struct NoInt *novo = (struct NoInt *)
    malloc(sizeof(struct NoInt));
  if (novo == NULL) return cabeca;
  novo->dado = dado;


}
```

LISTA LIGADA (linked list)

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

struct NoInt *
noint_insere_inicio(struct NoInt *cabeca,
                    int dado) {
  struct NoInt *novo = (struct NoInt *)
    malloc(sizeof(struct NoInt));
  if (novo == NULL) return cabeca;
  novo->dado = dado;
  novo->prox =       ;

}
```

LISTA LIGADA (linked list)

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

struct NoInt *
noint_insere_inicio(struct NoInt *cabeca,
                    int dado) {
  struct NoInt *novo = (struct NoInt *)
    malloc(sizeof(struct NoInt));
  if (novo == NULL) return cabeca;
  novo->dado = dado;
  novo->prox = cabeca;

}
```

LISTA LIGADA (linked list)

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

struct NoInt *
noint_insere_inicio(struct NoInt *cabeca,
                    int dado) {
  struct NoInt *novo = (struct NoInt *)
    malloc(sizeof(struct NoInt));
  if (novo == NULL) return cabeca;
  novo->dado = dado;
  novo->prox = cabeca;
  return novo;
}
```

LISTA LIGADA (linked list)

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

#include <stdio.h>

void
noint_imprime_lista(struct NoInt *cabeca) {




}
```

LISTA LIGADA (linked list)

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

#include <stdio.h>

void
noint_imprime_lista(struct NoInt *cabeca) {
  struct NoInt *atual = cabeca;




}
```

LISTA LIGADA (linked list)

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

#include <stdio.h>

void
noint_imprime_lista(struct NoInt *cabeca) {
  struct NoInt *atual = cabeca;
  while (atual != NULL) {


  }

}
```

LISTA LIGADA (linked list)

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

#include <stdio.h>

void
noint_imprime_lista(struct NoInt *cabeca) {
  struct NoInt *atual = cabeca;
  while (atual != NULL) {
    printf("%d ", atual->dado);

  }

}
```

LISTA LIGADA (linked list)

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

#include <stdio.h>

void
noint_imprime_lista(struct NoInt *cabeca) {
  struct NoInt *atual = cabeca;
  while (atual != NULL) {
    printf("%d ", atual->dado);
    atual = atual->prox;
  }

}
```

LISTA LIGADA (linked list)

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

#include <stdio.h>

void
noint_imprime_lista(struct NoInt *cabeca) {
  struct NoInt *atual = cabeca;
  while (atual != NULL) {
    printf("%d ", atual->dado);
    atual = atual->prox;
  }
  printf("\n");
}
```

LISTA LIGADA (linked list)

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

#include <stdio.h>

void
noint_imprime_lista(struct NoInt *cabeca) {
  while (cabeca != NULL) {
    printf("%d ", cabeca->dado);
    cabeca = cabeca->prox;
  }
  printf("\n");
}
```

*Porque isto **não altera** o nó cabeça fora da função?*

TESTE NO CLING

```
[cling]$ .L noint.c
[cling]$ struct NoInt *cabeca = NULL;
[cling]$
```

1. Imprima a lista. O que aparece na tela?
2. Insira o elemento 18 no início e imprima a lista.
3. Insira no início, nesta ordem: 42, 33 e 99
5. Imprima a lista

TESTE NO CLING

```
[cling]$ .L noint.c
[cling]$ struct NoInt *cabeca = NULL;
[cling]$ noint_imprime_lista(cabeca)
```

TESTE NO CLING

```
[cling]$ .L noint.c
[cling]$ struct NoInt *cabeca = NULL;
[cling]$ noint_imprime_lista(cabeca)

[cling]$
```

Lista vazia. Pulou linha sem imprimir mais nada.

TESTE NO CLING

```
[cling]$ .L noint.c
[cling]$ struct NoInt *cabeca = NULL;
[cling]$ noint_imprime_lista(cabeca)

[cling]$ cabeca =
         noint_insere_inicio(cabeca,18)
```

TESTE NO CLING

```
[cling]$ .L noint.c
[cling]$ struct NoInt *cabeca = NULL;
[cling]$ noint_imprime_lista(cabeca)

[cling]$ cabeca =
         noint_insere_inicio(cabeca,18)
[cling]$ noint_imprime_lista(cabeca)
```

TESTE NO CLING

```
[cling]$ .L noint.c
[cling]$ struct NoInt *cabeca = NULL;
[cling]$ noint_imprime_lista(cabeca)

[cling]$ cabeca =
         noint_insere_inicio(cabeca,18)
[cling]$ noint_imprime_lista(cabeca)
18
[cling]$
```

TESTE NO CLING

```
[cling]$ .L noint.c
[cling]$ struct NoInt *cabeca = NULL;
[cling]$ noint_imprime_lista(cabeca)

[cling]$ cabeca =
         noint_insere_inicio(cabeca,18)
[cling]$ noint_imprime_lista(cabeca)
18
[cling]$ cabeca =
         noint_insere_inicio(cabeca,42)
```

TESTE NO CLING

```
[cling]$ .L noint.c
[cling]$ struct NoInt *cabeca = NULL;
[cling]$ noint_imprime_lista(cabeca)

[cling]$ cabeca =
         noint_insere_inicio(cabeca,18)
[cling]$ noint_imprime_lista(cabeca)
18
[cling]$ cabeca =
         noint_insere_inicio(cabeca,42)
[cling]$ cabeca =
         noint_insere_inicio(cabeca,33)
```

Continue no seu computador.

REMOÇÃO DO INÍCIO

```
struct NoInt {
   int dado;
   struct NoInt *prox;
};

struct NoInt *
noint_remove_inicio(struct NoInt *cabeca) {




}
```

REMOÇÃO DO INÍCIO

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

struct NoInt *
noint_remove_inicio(struct NoInt *cabeca) {
  if (cabeca == NULL) return NULL;



}
```

REMOÇÃO DO INÍCIO

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

struct NoInt *
noint_remove_inicio(struct NoInt *cabeca) {
  if (cabeca == NULL) return NULL;
  struct NoInt *nova_cabeca = cabeca->prox;


}
```

REMOÇÃO DO INÍCIO

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

struct NoInt *
noint_remove_inicio(struct NoInt *cabeca) {
  if (cabeca == NULL) return NULL;
  struct NoInt *nova_cabeca = cabeca->prox;
  free(cabeca);

}
```

REMOÇÃO DO INÍCIO

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

struct NoInt *
noint_remove_inicio(struct NoInt *cabeca) {
  if (cabeca == NULL) return NULL;
  struct NoInt *nova_cabeca = cabeca->prox;
  free(cabeca);
  return nova_cabeca;
}
```

REMOÇÃO DO INÍCIO

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

struct NoInt *
noint_remove_inicio(struct NoInt *cabeca) {
  if (cabeca == NULL) return NULL;
  struct NoInt *nova_cabeca = cabeca->prox;
  free(cabeca);
  return nova_cabeca;
}
```

TESTE NO CLING: vá removendo e imprimindo a lista.

ÚLTIMO NÓ DA LISTA

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

// implemente a função abaixo para
// retornar o último nó da lista
struct NoInt *
noint_ultimo_no(struct NoInt *cabeca) {
  ...
}
```

TESTE NO CLING para: a) lista vazia; b) com apenas 1 elemento; c) caso geral

OPERAÇÕES NO FINAL DA LISTA

```
struct NoInt {
  int dado;
  struct NoInt *prox;
};

// implemente as funções abaixo para
// operar no final da lista e retornar
// o nó cabeça atualizado
struct NoInt *
noint_insere_fim(struct NoInt *cabeca, int d);

struct NoInt *
noint_remove_fim(struct NoInt *cabeca);
```

TESTE NO CLING para: a) lista vazia; b) com apenas 1 elemento; c) caso geral

PARA CASA

Lista 2 no site.