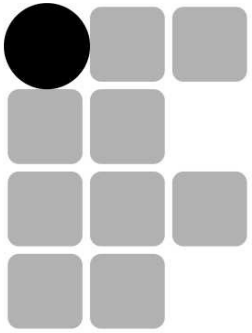


INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO SUL
Câmpus Feliz

Estruturas de Dados

Listas Encadeadas

Prof. Moser Fagundes

Programação II
Técnico em Informática

Sumário

- Recapitulação do conteúdo da aula anterior
- Listas encadeadas simples
- Listas duplamente encadeadas
- ArrayList e LinkedList

O que são estruturas de dados?

Quais são as operações básicas que podemos realizar em uma lista linear?

Qual é a diferença entre uma lista sequencial e uma lista encadeada?

Como é feita uma busca binária em uma lista linear?

Como é feita uma busca binária em uma lista linear?

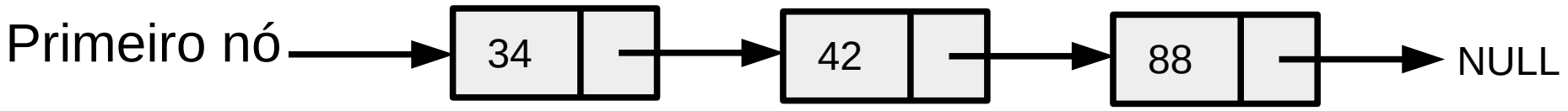
Código em Java no Moodle.

Listas Encadeadas

Uma **lista encadeada** (em inglês *linked list*) é uma representação **dinâmica** e **linear** de uma sequência de objetos na memória.

Cada elemento da sequência é armazenado em um **nó** da lista. Cada nó possui um encadeamento (*link*) para o próximo nó.

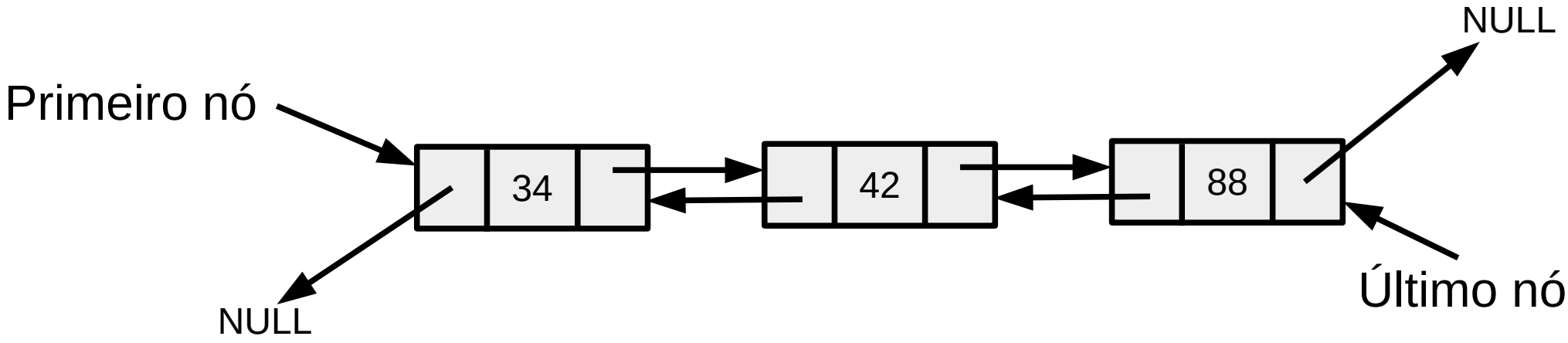
Exemplo: Lista encadeada com 3 elementos (3 nós)



Listas Duplamente Encadeadas

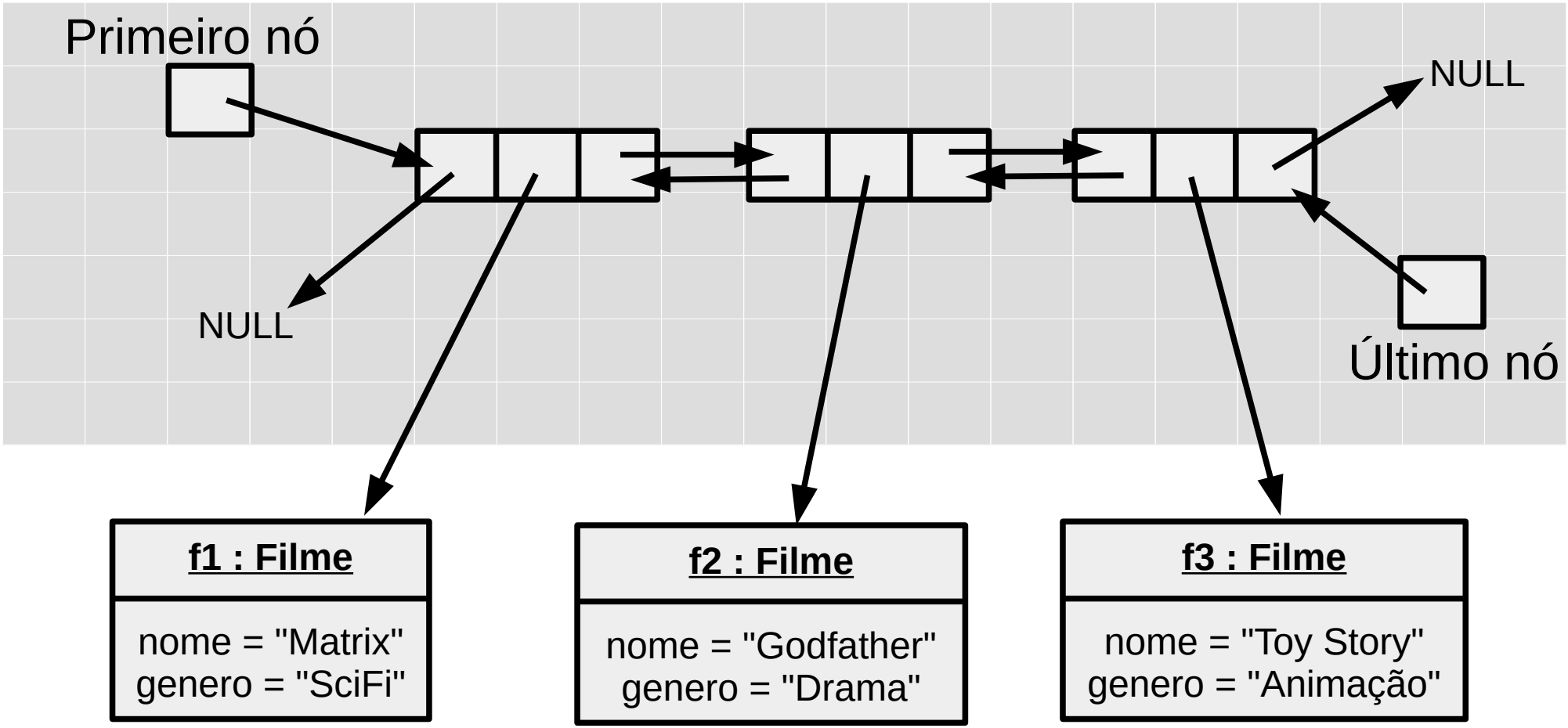
Uma **lista duplamente encadeada** (em inglês *doubly linked list*) se diferencia da lista encadeada simples por possuir encadeamento (*link*) tanto para o próximo elemento quanto para o elemento anterior.

Exemplo: Lista **duplamente** encadeada com 3 elementos (3 nós)



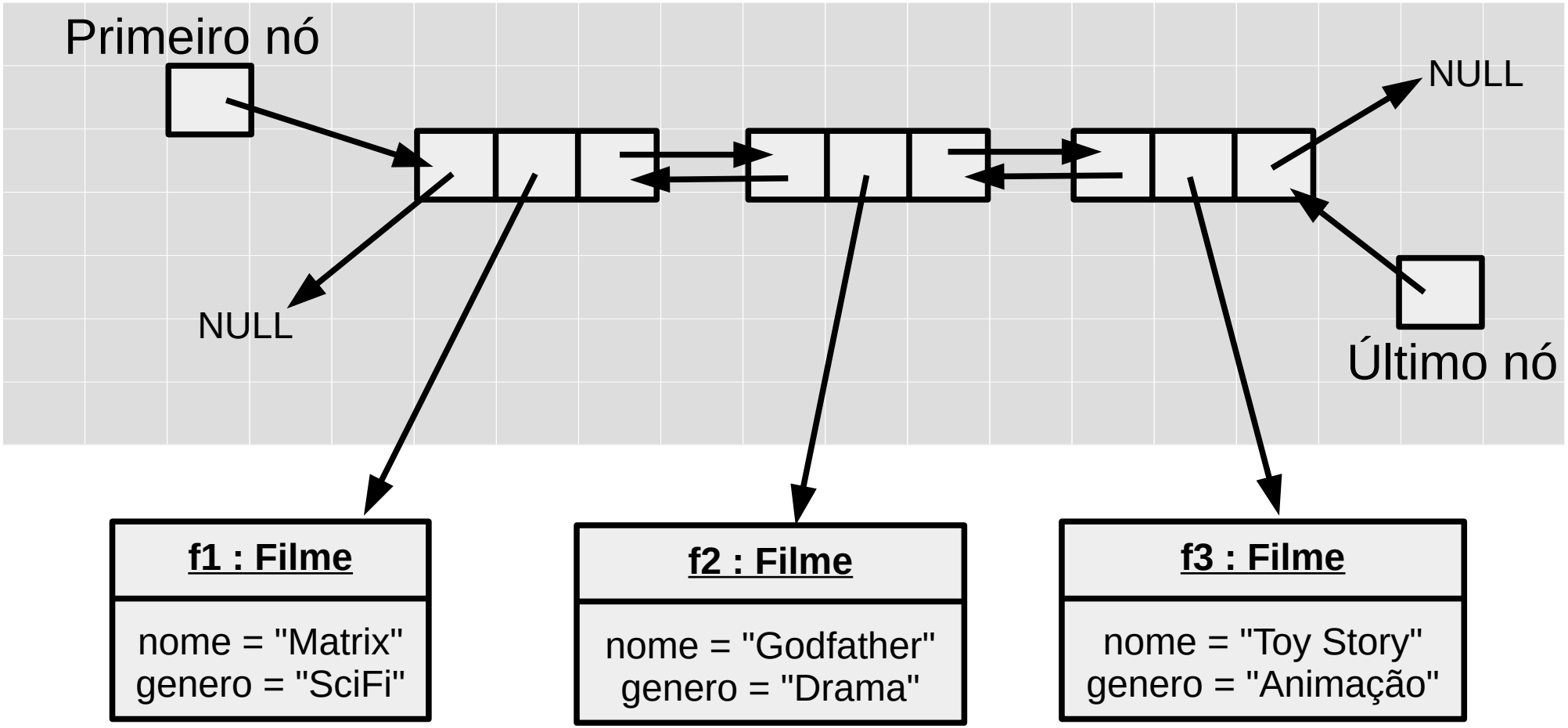
Listas Duplamente Encadeadas

Exemplo: Lista **duplamente** encadeada com 3 objetos Filme.



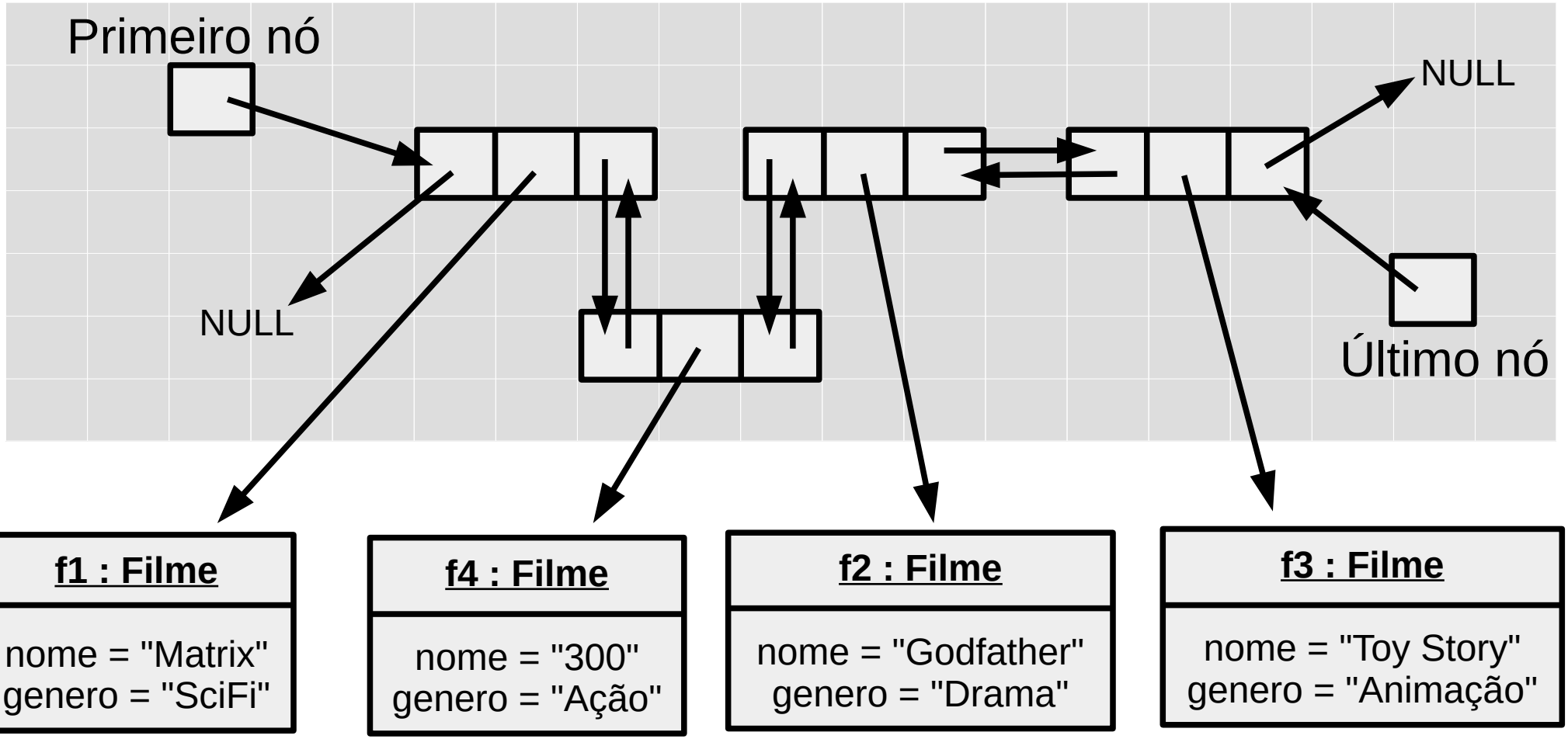
Listas Duplamente Encadeadas

Exemplo: Vamos inserir um novo filme na segunda posição.



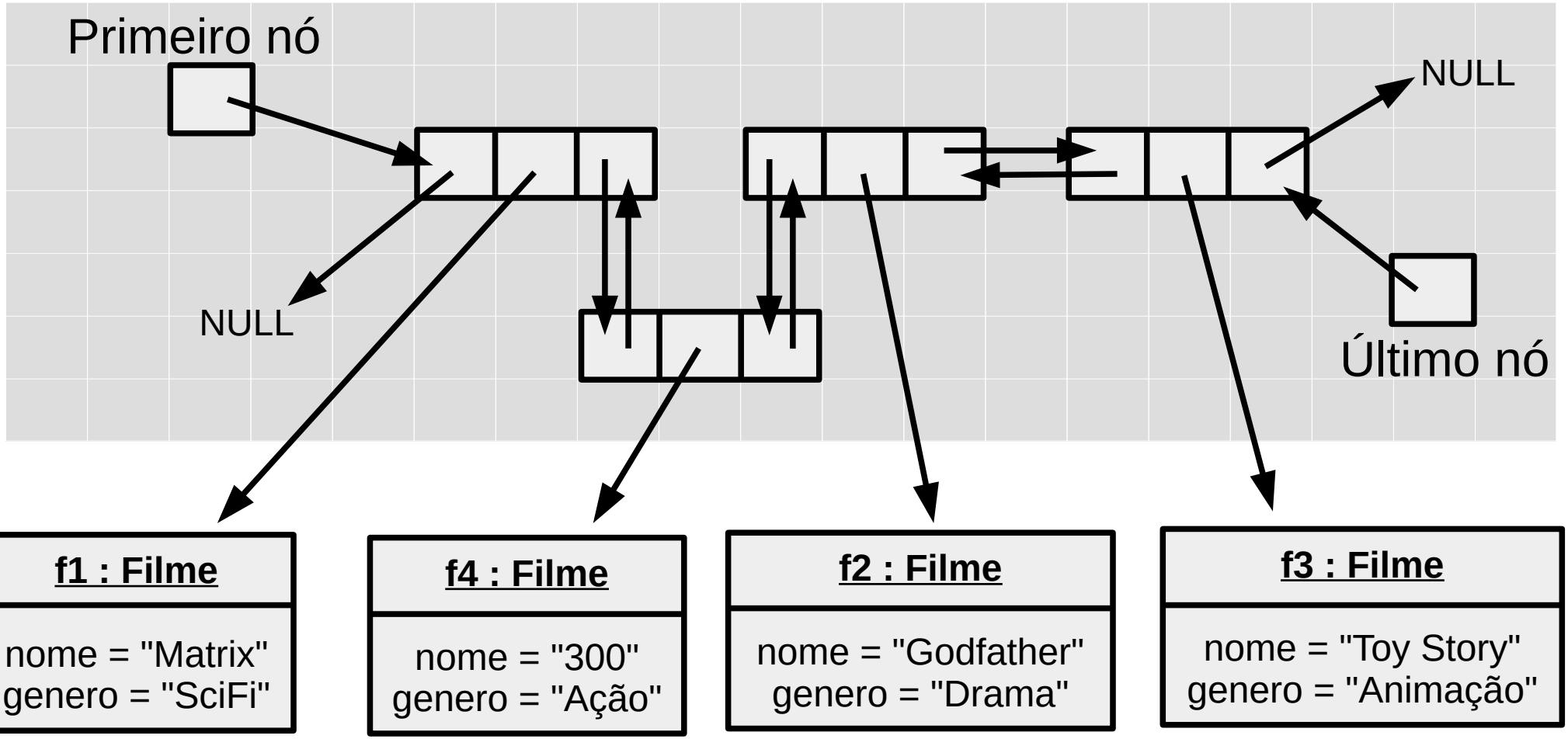
Listas Duplamente Encadeadas

Exemplo: Um novo filme **foi inserido** na segunda posição.



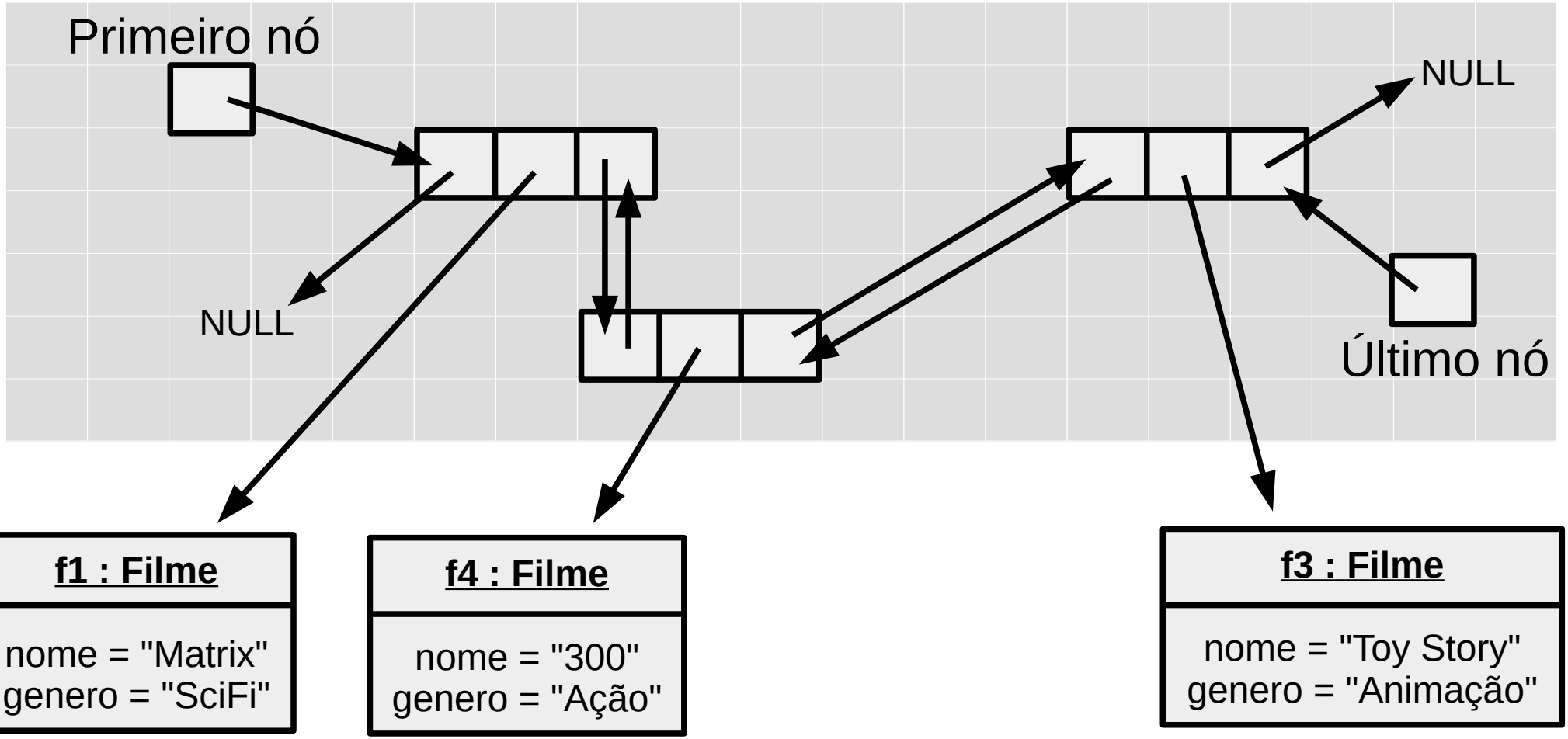
Listas Duplamente Encadeadas

Exemplo: Vamos **remover** o filme f2 da terceira posição.



Listas Duplamente Encadeadas

Exemplo: O filme f2 **foi removido** da terceira posição.



Listas Encadeadas

Vantagens das **lista encadeadas**:

- Inserir ou remover elementos não implica na realocação (movimentação) de dados na memória.
- Não é necessário definir o número de elementos da lista encadeada.

Listas Encadeadas

Desvantagens das **lista encadeadas**:

- Os algoritmos de inserção e remoção devem ser cuidadosamente implementados, caso contrário é possível que elementos da lista sejam perdidos por encadeamentos incorretos.
- Para acessar o n-ésimo elemento, temos que percorrer os n-1 elementos anteriores da lista.

`ArrayList` e `LinkedList`

`ArrayList` e `LinkedList` são muito parecidas pois ambas implementam a interface `List`.

A interface `List` obriga a implementação dos métodos como `get`, `set`, `add` e `remove`.

Ambas admitem elementos repetidos.

`ArrayList`

- O `ArrayList` é implementado através de um **array** que é redimensionado dinamicamente sempre que necessário
- O tamanho do array aumenta em 50% quando necessário. Por exemplo, se o array tem 10 elementos, ao ser preenchido ele adiciona mais 5 posições.
  - *Isso é transparente para o usuário...*
  - *Tem um custo computacional associado ao redimensionamento pois é necessário copiar os elementos do array.*

`LinkedList`

- O `LinkedList` é implementado através de uma **lista duplamente encadeada**.

- Na maioria dos casos (não todos), possui melhor performance que o `ArrayList` nas inserções no início do array (para isso usamos o método `addFirst`).

- Em contrapartida, os métodos `get` têm uma performance pior que o `ArrayList`.

`LinkedList`

- O `LinkedList` também fornece elementos para a implementação de **filas** e **deques**:
  - `poolFirst(...)` obtém e remove o primeiro elemento. Ou seja, funciona como um método get que além de acessar o objeto também o remove.
  - `poolFirst(...)` obtém e remove o último elemento.

Exemplo

Criar a classe `Filme`

```
public class Filme {
    private String nome;
    private String genero;

    public String getNome() {
        return nome;
    }
    public void setNome(String nome) {
        this.nome = nome;
    }
    public String getGenero() {
        return genero;
    }
    public void setGenero(String genero) {
        this.genero = genero;
    }
}
```

Exemplo

```
public class Exemplo1 {
    public static void main(String[] args) {
        LinkedList<Filme> lista = new LinkedList<Filme>();

        Filme f1 = new Filme();
        f1.setNome("Matrix");
        f1.setGenero("SciFi");

        Filme f2 = new Filme();
        f2.setNome("Godfather");
        f2.setGenero("Drama");

        lista.add(f1);
        lista.add(f2);

        // addFirst, addLast, uso do Iterator
        // poolFirst, poolLast
    }
}
```

Qual é a diferença de performance entre um `ArrayList` e uma `LinkedList` em operações básicas?