INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO PIAUÍ

CAMPUS TERESINA-CENTRAL

DIRETORIA DE ENSINO

Estrutura de Dados
Aula 3 – Listas
Usando Lista Dinâmica

Professora: Elanne Cristina O. dos Santos

elannecristina.santos@gmail.com
elannecristina.santos@ifpi.edu.br

LISTA DINÂMICA – A estrutura de cada elemento da lista

- EXEMPLO:

```
typedef struct elemento{
      int mat;
      char nome[20];
      elemento *prox;
}Elemento;
```

Alocação dinâmica para cada elemento

- Depois de declarar o elemento, é necessário alocar espaço de memória para só depois poder utilizá-lo.

```
main(){
    Elemento *novo;
    //Aloca memoria para o novo elemento
    novo = (Elemento *)malloc(sizeof(Elemento));
    novo->mat=1;
    strcpy(novo->nome,"Joao");
    novo->prox=NULL;
}
```

novo

1	João	Null

Para inserir outro elemento, como encadeá-los em uma lista?

- Guarde a posição do primeiro elemento da lista:

```
Elemento *inicio;        inicio= NULL;
```

- Guarde a posição do último elemento da lista:

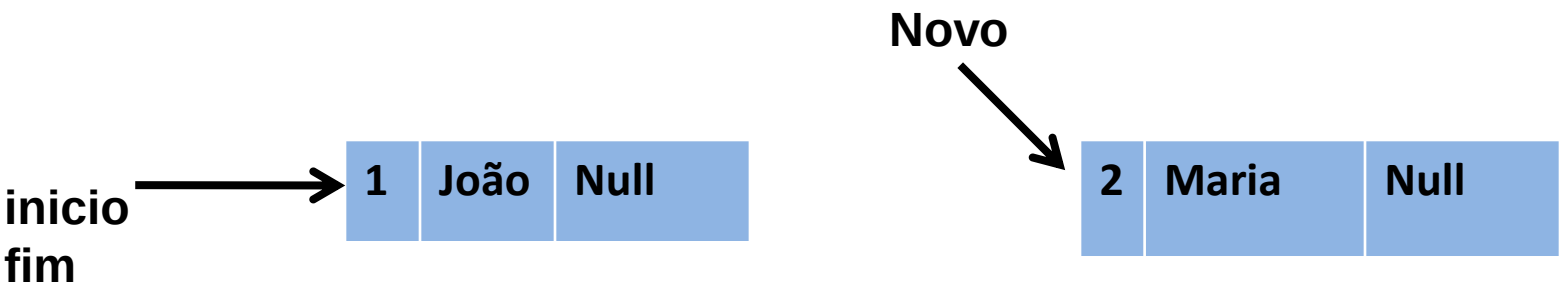
```
Elemento *fim;           fim=NULL;
```

- Quando só temos um elemento:

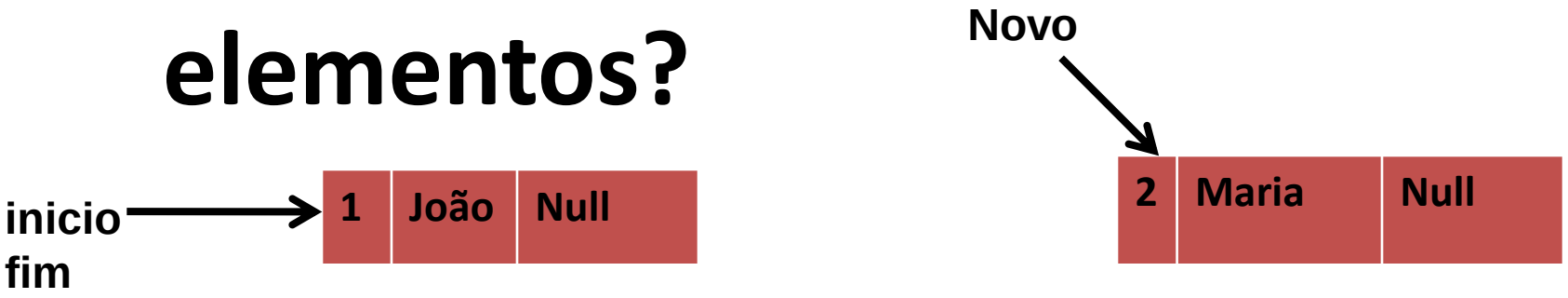
```
If (inicio == NULL) {
     inicio  = novo;
      fim = novo;
}else.....
```

Crie um novo elemento

```
novo = (Elemento *)malloc(sizeof(Elemento));
novo->mat=2;
strcpy(novo->nome,"Maria");
novo->prox=NULL;
```

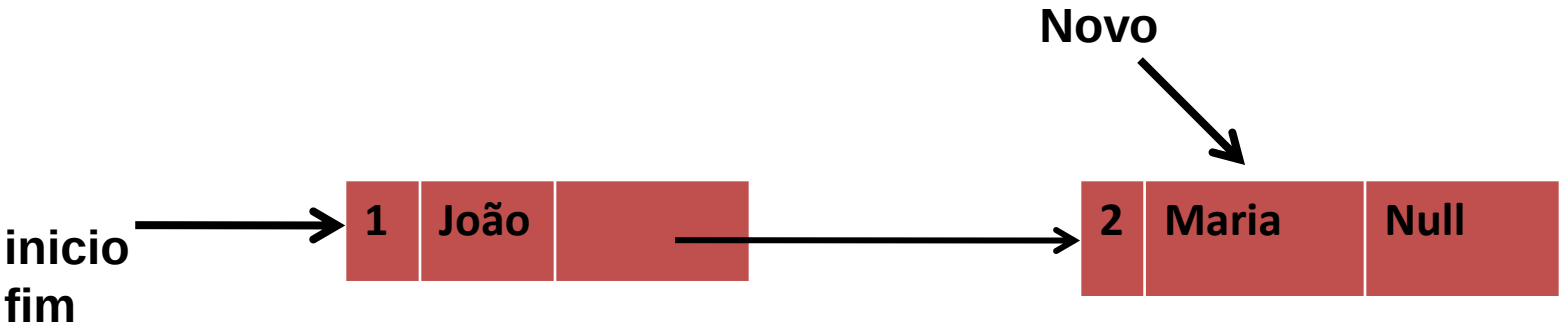


Como encadear os elementos?



- Para encadear os elementos:

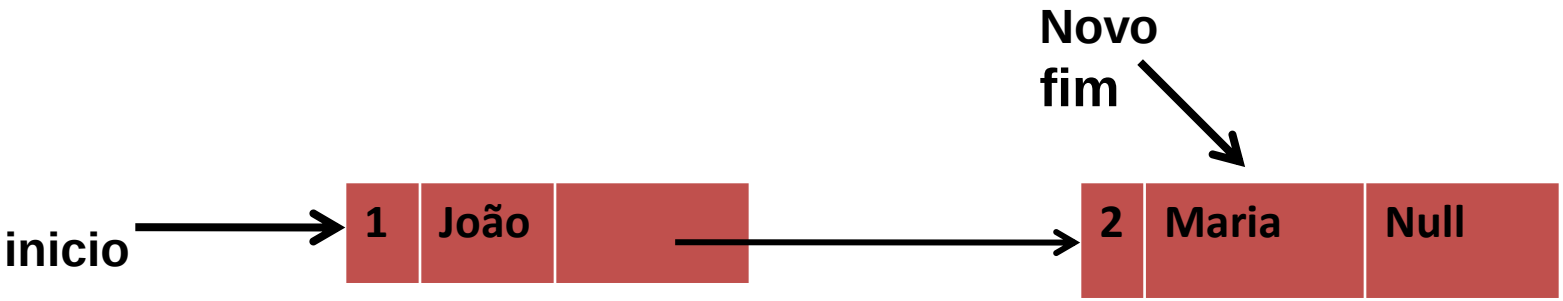
fim->prox=novo;



O que ainda falta fazer para atualizar a lista?

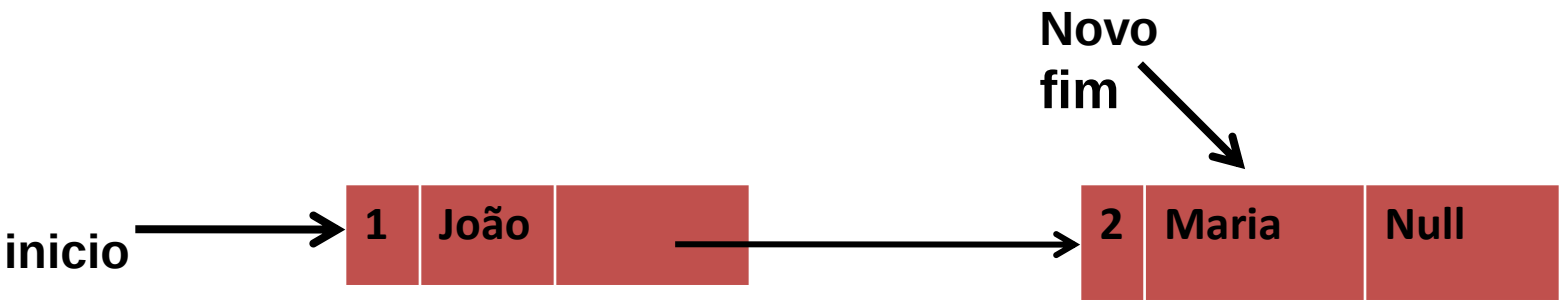
Atualize o último elemento da lista

fim = novo; Ou fim = fim->prox;



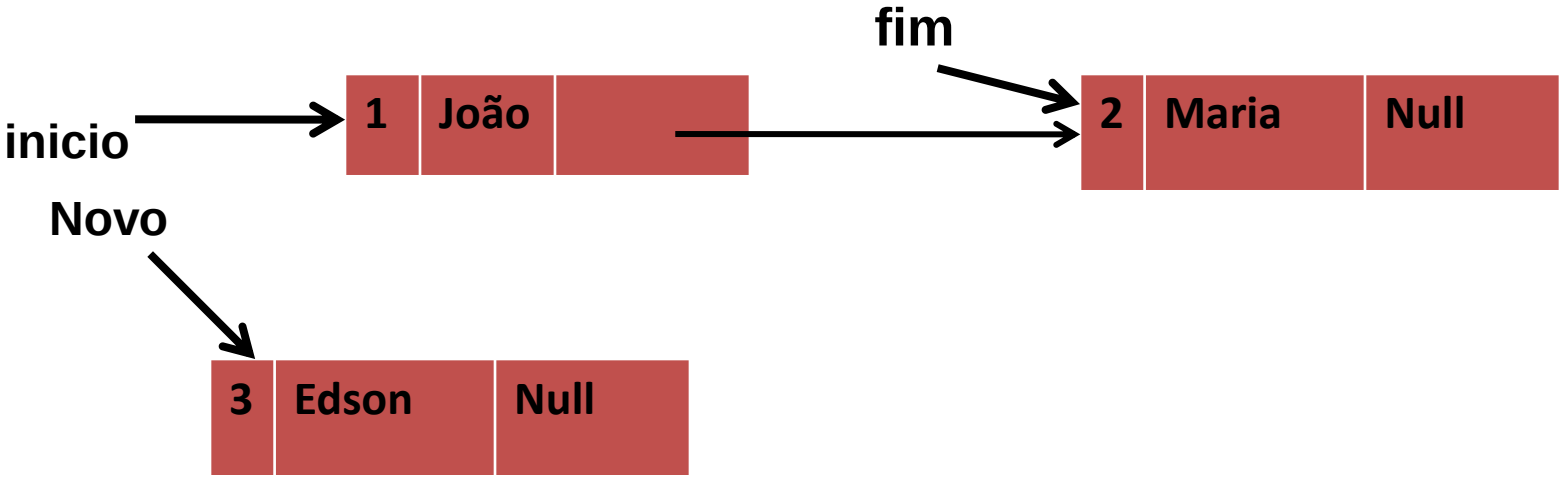

```
If (inicio == NULL) {
     inicio  = novo;
      fim = novo;
}else{
       fim->prox = novo;
       fim = novo;  }
```

Como fazer um programa pra percorrer a lista do inicio até o fim?

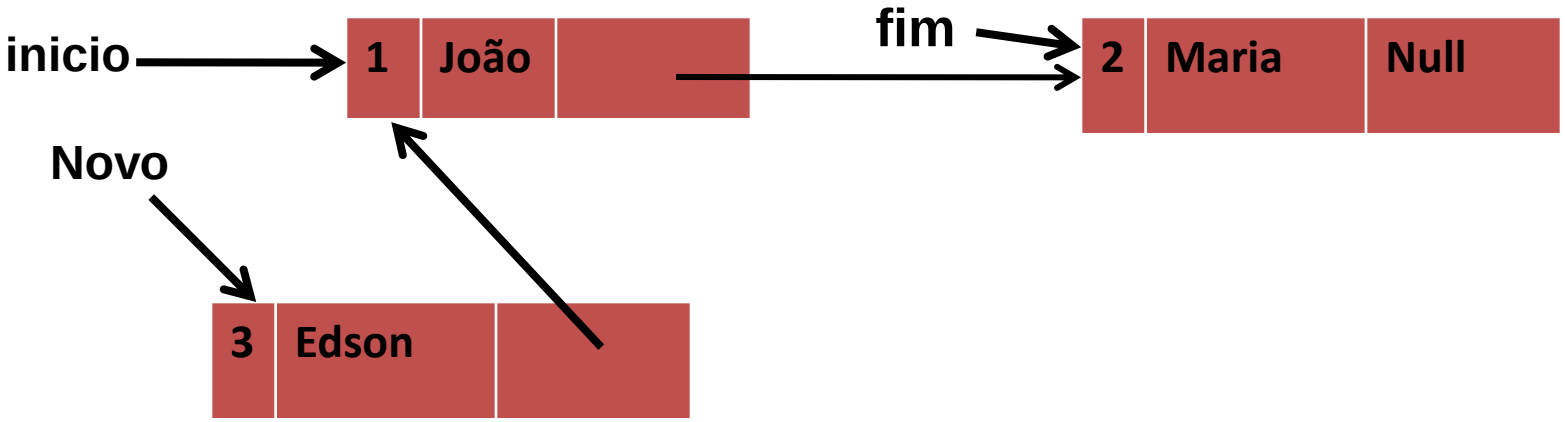



```
atual = inicio;
while (atual != NULL) {
        printf("%d",atual->mat);
        printf("%s",atual->nome);
        atual=atual->prox;
}
```

Como incluir o elemento sempre no início da lista?



- Faça com que ***novo->prox*** aponte para o ***inicio:***



- Atualize o valor de ***inicio*** ➔ ***inicio = novo;***

Exemplo – Struct aluno

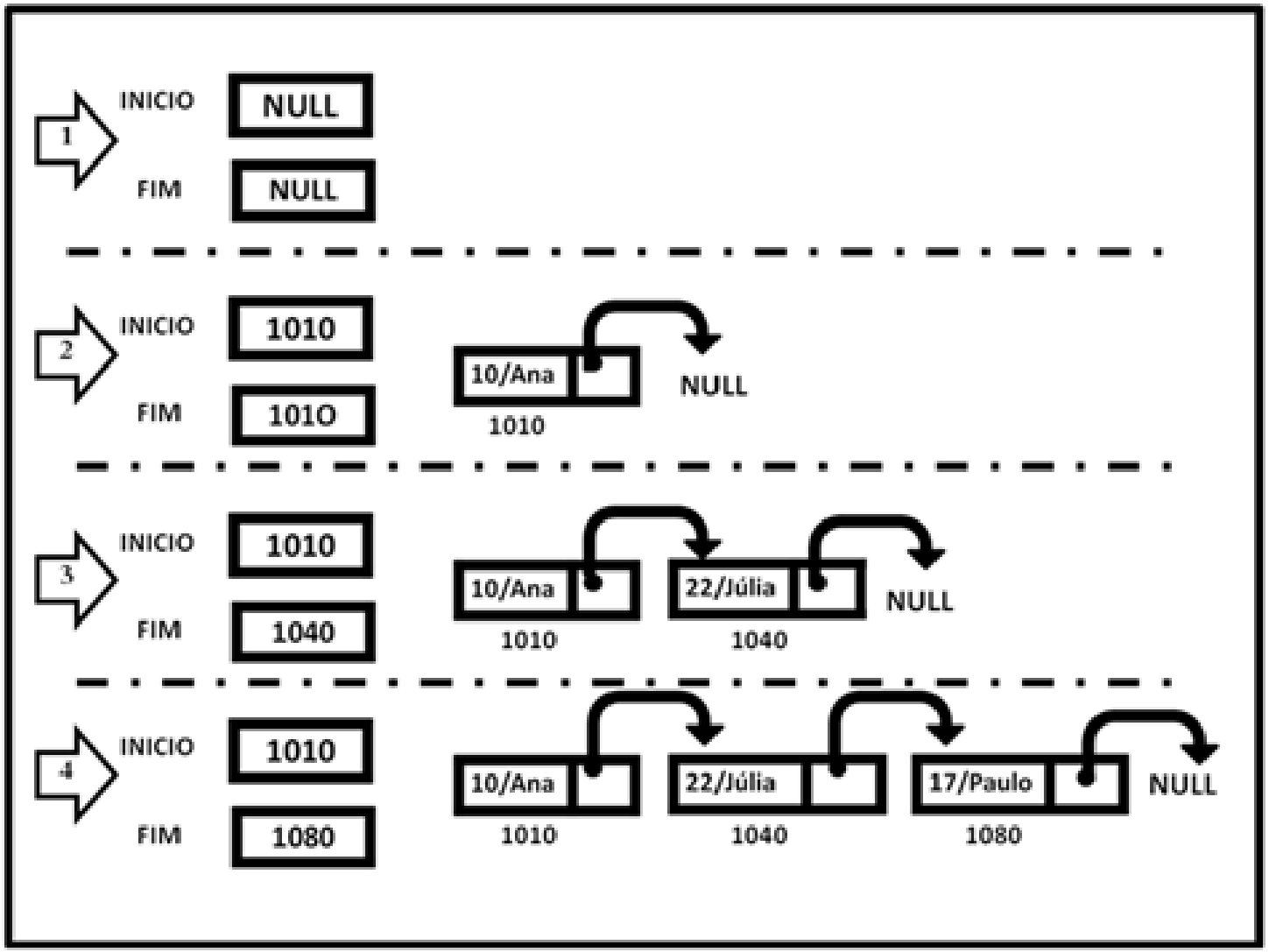
```
struct aluno {
    int mat;
    char nome[20];
    float nota;
    aluno *prox;
      };

typedef struct aluno Taluno;

Taluno *inicio;
Taluno *fim;
Taluno *novo;
```

LISTA DINÂMICA DESORDENADA

- Esta lista é implementada usando ponteiros. A memória para armazenar os dados é alocada em tempo de execução.

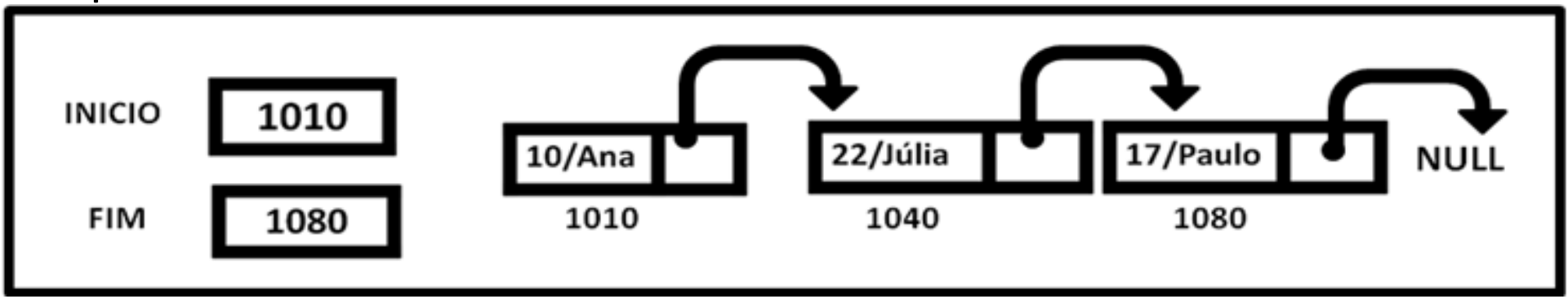


```
// aloca espaco de memoria para novo
    novo = (Taluno *)malloc(sizeof(Taluno));
    novo->mat= matricula;
    strcpy(novo->nome,nome);
    novo->nota = nota;
    novo->prox = NULL;
    if (inicio==NULL) {
       inicio = novo;
       fim  = novo;
     }
     else {
       fim->prox = novo;
       fim = novo;
      }
```

LISTA DINÂMICA – Inclusão desordenada

LISTA DINÂMICA - Busca

- Para fazer uma consulta em uma lista dinâmica é necessário saber qual elemento deseja consultar.Ex.: consulta por matrícula.
- Um ponteiro auxiliar deve ser usado para percorrer a lista, visitando cada nó a procura do elemento.
- Caso quiséssemos consultar o elemento de matrícula 25, iríamos percorrer a lista até chegar no último nó, cujo endereço do vizinho é NULL (nó de endereço 1080) e ficaríamos sabendo que este elemento não se encontra na lista.
- Quando um elemento é encontrado, seus dados são apresentados
- Quando ele não está na lista, uma mensagem é apresentada dizendo que o elemento não existe na lista.



LISTA DINÂMICA - Busca

```
do{
      printf("\nConsulta aluno pelo numero de matricula\n\n");
     printf("\nMatricula: ");
     scanf("%d",&matc);
     noatual = inicio;          achou = 0;
     while (noatual != NULL) {
       if (noatual->mat == matc) {
          achou = 1;
          printf("\n\nMatricula Nome\n");
          printf("-----------------------------------\n");   printf("%9d %-
20s\n",noatual->mat, noatual->nome);
          printf("-----------------------------------\n");   break;
       }   else
            noatual = noatual->prox;  }
```

Lista Dinâmica - Busca

```
if (achou == 0)
            printf("\n\nAluno nao encontrado!!\n");
 printf("\nContinuar consultando (1-sim/2-
  nao)? ");
  scanf("%d",&continuar);
    }while (continuar == 1);
```

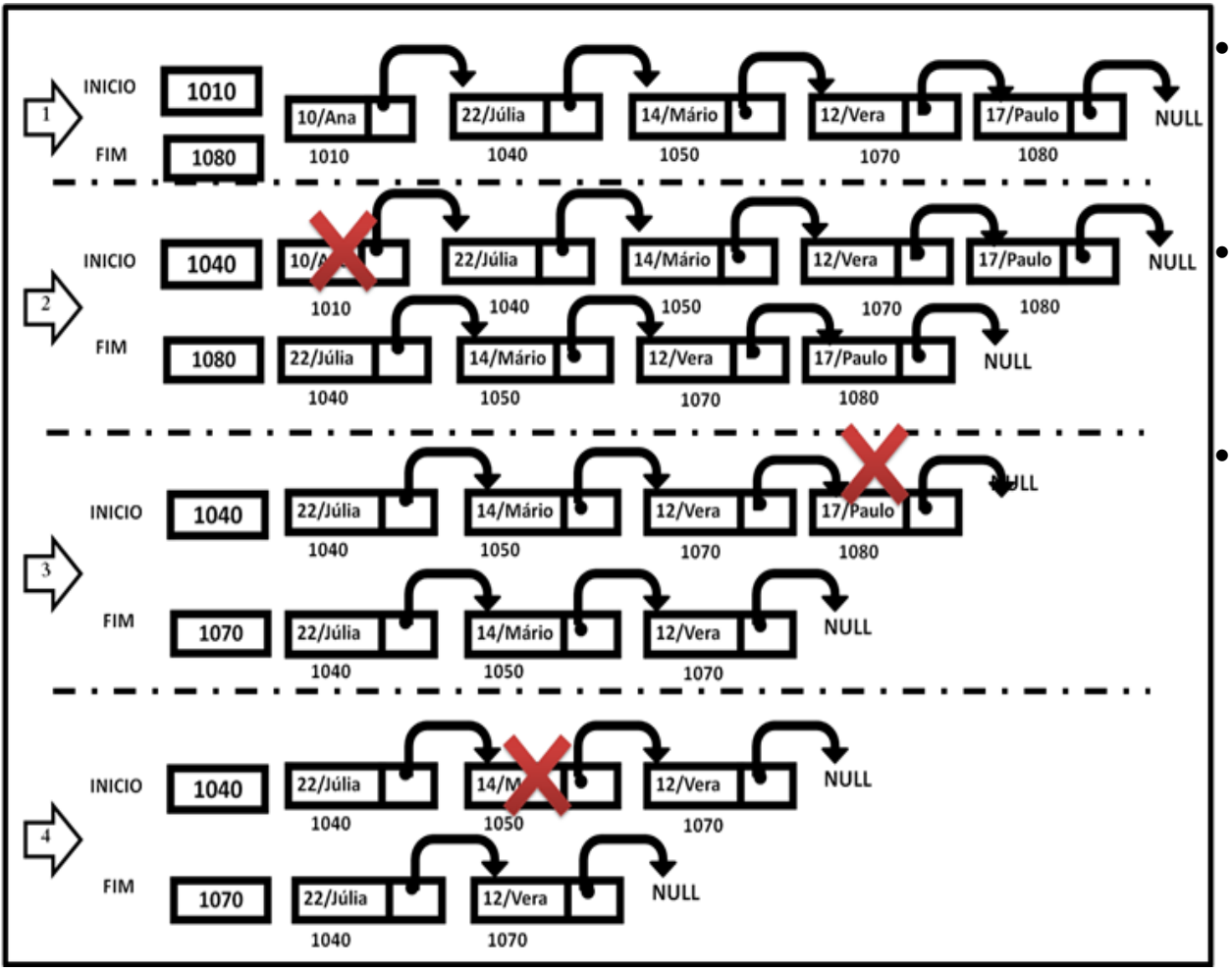
LISTA DINÂMICA – listar todos os elementos

```
void listar(){
    TAluno *noatual = inicio;
    printf("\nListagem de Alunos\n\n");
    if (qa != 0) {
      printf("\n\nMatricula Nome\n");
      printf("-----------------------------------\n");
       while( noatual != NULL) {
            printf("%d %s\n",noatual->mat, noatual->nome);
            noatual = noatual->prox;
       }
      printf("-----------------------------------\n");
      printf("\n\nQuantidade de Alunos = %d\n",qa);
    }
    else
        printf("\n\n Nao tem nenhum aluno cadastrado");
```

LISTA DINÂMICA – Remover um elemento

- Identificar qual elemento deseja remover;
- É feita uma varredura em todos os nós da lista;
- Se elemento é encontrado➔ remove elemento da lista;
  - 3 situações:
  - O elemento encontrado está no INICIO da lista
  - O elemento encontrado está no MEIO da lista
  - O elemento encontrado está no FINAL da lista
- Senão ➔uma mensagem deve ser informada ao usuário informando que o elemento não existe.

LISTA DINÂMICA – Remover um elemento



- **Primeiro elemento da lista:**
  - Inicio = inicio->prox
- **Último elemento da lista:**
  - fim = anterior;
  - Fim->prox=NULL;
- **Elemento no meio da lista:**
  - anterior->prox=noatual->prox;

- **Para percorrer a lista crie os ponteiros:**

  Elemento *anterior, *atual;

  anterior = NULL;

  atual=inicio;

- **Podem acontecer três casos diferentes:**

  ➢remover primeiro da lista:

  inicio = inicio->prox;

  ➢ último da lista:

  fim = anterior;

  fim->prox=NULL;

  ➢elemento no meio da lista:

  Anterior->prox=noatual->prox;

```
int remove(int mat){
        No *atual=inicio;
        No *anterior=NULL;
        while (atual!=NULL){
                if (atual->mat==mat){
                   if (atual==inicio)
                       inicio = inicio->prox;
                   else
                      if (atual==fim){
                        fim=anterior;
                        fim->prox=NULL;
                      }
                      else
                         anterior->prox=atual->prox;
                    free(atual);
                    return 1;
                }
            anterior=atual;
            atual=atual->prox;
         }
         return 0;
      }
```

Atividade

- Considerando a seguinte estrutura:

OPÇÃO 1: Usando Struct

```
struct aluno {
    int mat;
    char nome[20];
    aluno *prox;
};

typedef struct aluno TAluno;
```

OPÇÃO 2: Usando Classes

```
#include <string>
#include <iostream>
using namespace std;
class No{
    public:
        int mat;
        string nome;
        No *prox;
        No(int m,string n){
                mat=m;
                nome = n;
                prox=NULL;
        }
};
```

OPÇÃO 2: Usando Classes

```
class Lista{
    public:
        No *inicio;
        No *fim;
       Lista(){
        inicio = NULL;
        fim = NULL;
        }
}
```

Atividade

Faça usando o modelo da opção 2 (usando classes):

1. Escreve o método para inserir no inicio da lista.
2. Escreva o método consultar, aonde será pesquisado um aluno através do número de matrícula. Se o aluno for encontrado uma mensagem com matrícula e nome deve ser impressa na tela do computador, se o aluno não for encontrado, a mensagem deve informar que o aluno não existe.
3. Escreva o método remover, aonde será retirado um elemento da lista.
4. Escreva o método listar, aonde serão listados todos os elementos da lista.
5. Criar uma nova lista que seja o inverso da primeira já criada.
6. Inverter a própria lista.

Obrigada pela atenção!!!
Boa semana de estudos pra vcs!!