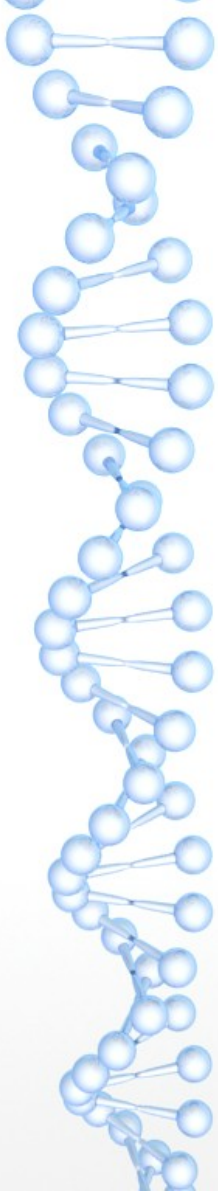
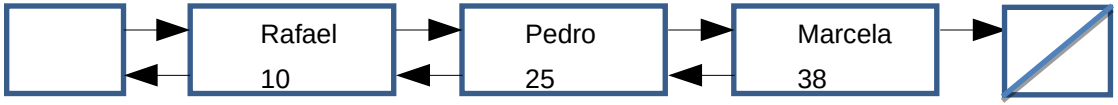


Estrutura de Dados - I

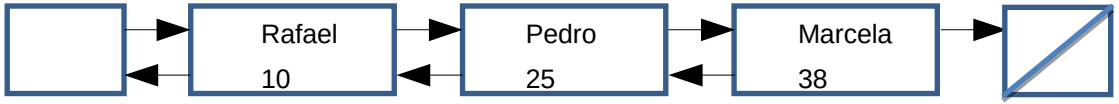
Listas Ligadas

Prof. MSc. Rafael Staiger Bressan
rafael.bressan@unicesumar.edu.br

Lista Duplamente Encadeada

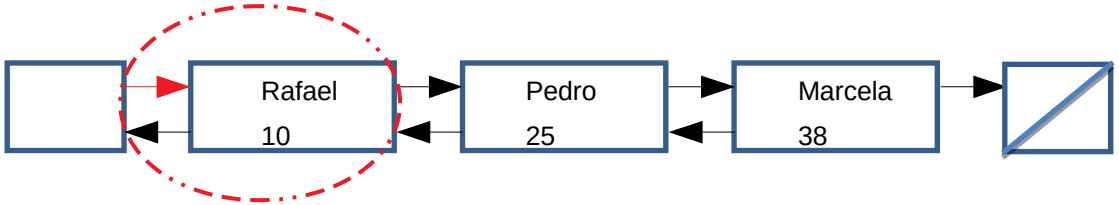


Lista Duplamente Encadeada
Remove no Início



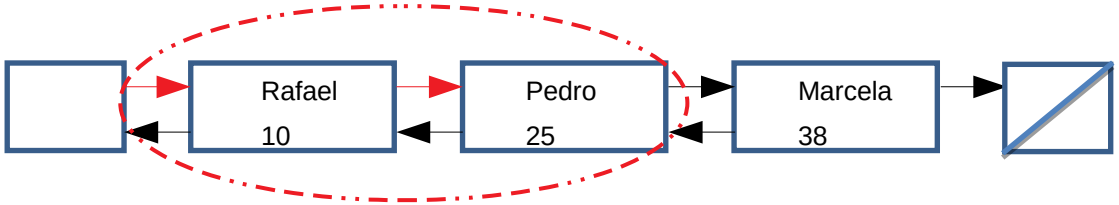

```
40    // Remove elemento no início da lista
41    void removeInicio(celula *ini) {
42        if(ini->prox == NULL){
43            printf("Lista Vazia!\n");
44        }else if( ini->prox->prox != NULL ){
45            ini->prox->prox->ant = ini;
46            ini->prox = ini->prox->prox;
47        }else{
48            ini->prox = NULL;
49        }
50    }
```

Lista Duplamente Encadeada
Remove no Início



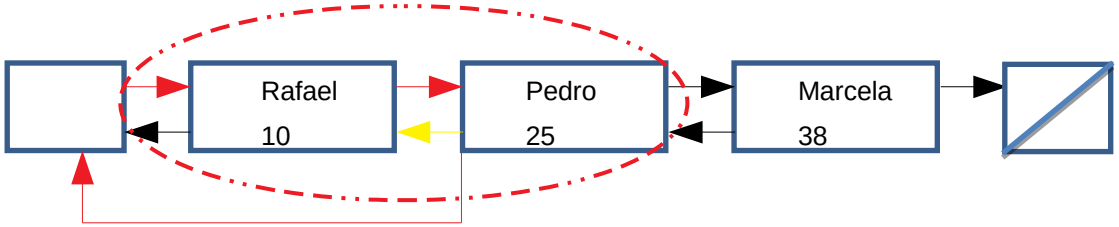

```
40    // Remove elemento no início da lista
41    void removeInicio(celula *ini) {
42        if(ini->prox == NULL){
43            printf("Lista Vazia!\n");
44        }else if( ini->prox->prox != NULL ){
45            ini->prox->prox->ant = ini;
46            ini->prox = ini->prox->prox;
47        }else{
48            ini->prox = NULL;
49        }
50    }
```

Lista Duplamente Encadeada
Remove no Início



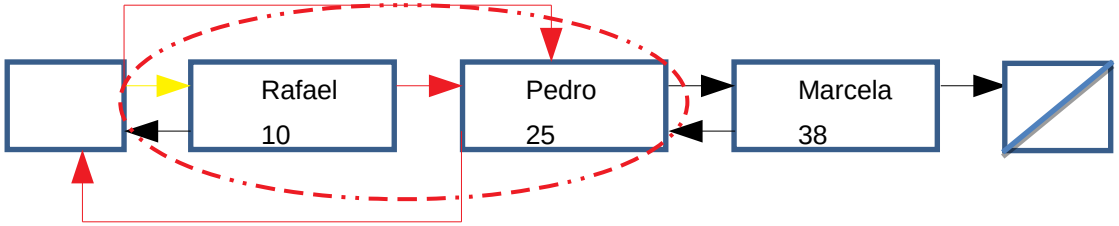

```
40  // Remove elemento no início da lista
41  void removeInicio(celula *ini) {
42      if(ini->prox == NULL){
43          printf("Lista Vazia!\n");
44      }else if( ini->prox->prox != NULL ){
45          ini->prox->prox->ant = ini;
46          ini->prox = ini->prox->prox;
47      }else{
48          ini->prox = NULL;
49      }
50  }
```

Lista Duplamente Encadeada
Remove no Início



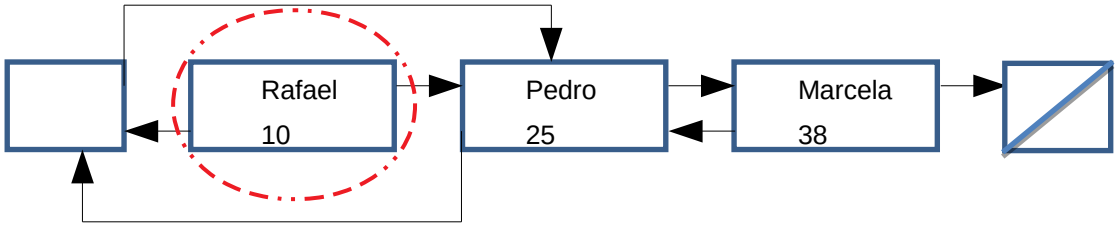

```
40    // Remove elemento no início da lista
41    void removeInicio(celula *ini) {
42        if(ini->prox == NULL){
43            printf("Lista Vazia!\n");
44        }else if( ini->prox->prox != NULL ){
45            ini->prox->prox->ant = ini;
46            ini->prox = ini->prox->prox;
47        }else{
48            ini->prox = NULL;
49        }
50    }
```

Lista Duplamente Encadeada
Remove no Início



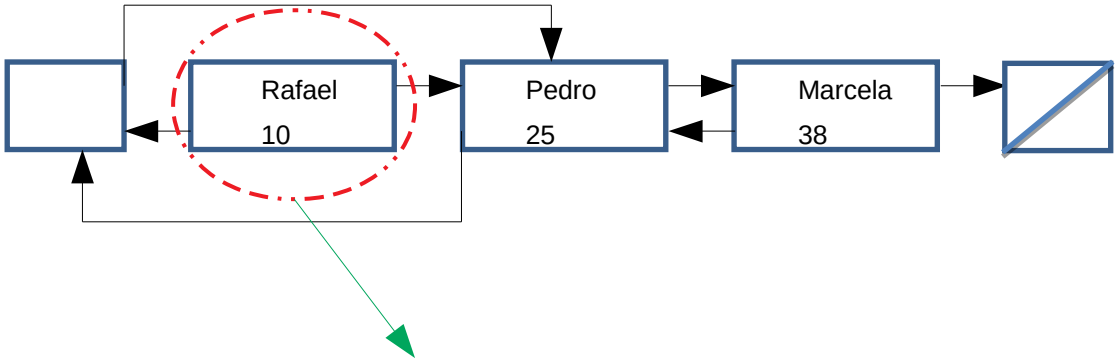

```
40    // Remove elemento no início da lista
41    void removeInicio(celula *ini) {
42        if(ini->prox == NULL){
43            printf("Lista Vazia!\n");
44        }else if( ini->prox->prox != NULL ){
45            ini->prox->prox->ant = ini;
46            ini->prox = ini->prox->prox;
47        }else{
48            ini->prox = NULL;
49        }
50    }
```

Lista Duplamente Encadeada
Remove no Início



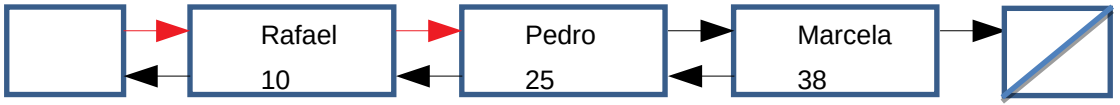

```
40    // Remove elemento no início da lista
41    void removeInicio(celula *ini) {
42        if(ini->prox == NULL){
43            printf("Lista Vazia!\n");
44        }else if( ini->prox->prox != NULL ){
45            ini->prox->prox->ant = ini;
46            ini->prox = ini->prox->prox;
47        }else{
48            ini->prox = NULL;
49        }
50    }
```

Lista Duplamente Encadeada
Remove no Início



Como “desalocar” a célula?

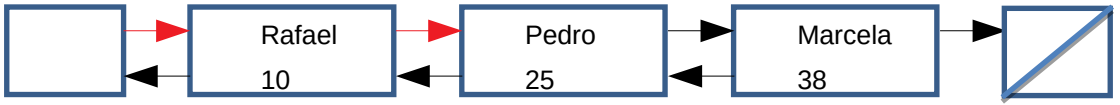
Lista Duplamente Encadeada
Remove no Início



LIXO

```
40  // Remove elemento no início da lista
41  void removeInicio(celula *ini) {
42      if(ini->prox == NULL){
43          printf("Lista Vazia!\n");
44      }else if( ini->prox->prox != NULL ){
45          celula *lixo;
46          lixo = ini->prox;
47          ini->prox->prox->ant = ini;
48          ini->prox = ini->prox->prox;
49          free(lixo);
50      }else{
51          ini->prox = NULL;
52      }
53  }
```

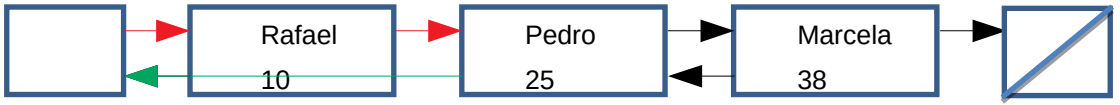
Lista Duplamente Encadeada
Remove no Início



LIXO

```
40  // Remove elemento no início da lista
41  void removeInicio(celula *ini) {
42      if(ini->prox == NULL){
43          printf("Lista Vazia!\n");
44      }else if( ini->prox->prox != NULL ){
45          celula *lixo;
46          lixo = ini->prox;
47          ini->prox->prox->ant = ini;
48          ini->prox = ini->prox->prox;
49          free(lixo);
50      }else{
51          ini->prox = NULL;
52      }
53  }
```

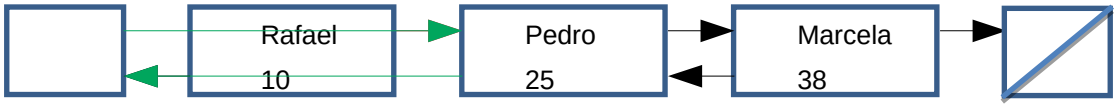
Lista Duplamente Encadeada
Remove no Início



LIXO

```
40  // Remove elemento no início da lista
41  void removeInicio(celula *ini) {
42      if(ini->prox == NULL){
43          printf("Lista Vazia!\n");
44      }else if( ini->prox->prox != NULL ){
45          celula *lixo;
46          lixo = ini->prox;
47          ini->prox->prox->ant = ini;
48          ini->prox = ini->prox->prox;
49          free(lixo);
50      }else{
51          ini->prox = NULL;
52      }
53  }
```

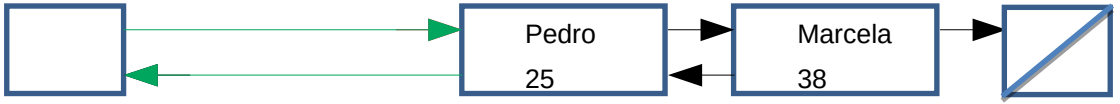
Lista Duplamente Encadeada
Remove no Início



LIXO

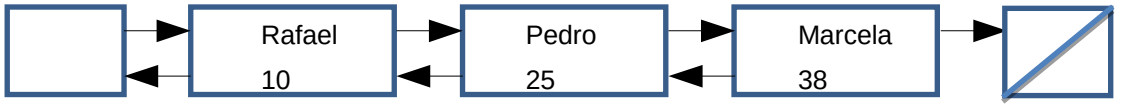
```
40  // Remove elemento no início da lista
41  void removeInicio(celula *ini) {
42      if(ini->prox == NULL){
43          printf("Lista Vazia!\n");
44      }else if( ini->prox->prox != NULL ){
45          celula *lixo;
46          lixo = ini->prox;
47          ini->prox->prox->ant = ini;
48          ini->prox = ini->prox->prox;
49          free(lixo);
50      }else{
51          ini->prox = NULL;
52      }
53  }
```

Lista Duplamente Encadeada
Remove no Início



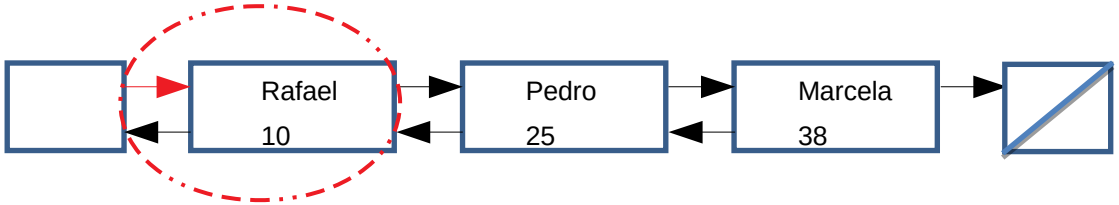

```
40  // Remove elemento no início da lista
41  void removeInicio(celula *ini) {
42      if(ini->prox == NULL){
43          printf("Lista Vazia!\n");
44      }else if( ini->prox->prox != NULL ){
45          celula *lixo;
46          lixo = ini->prox;
47          ini->prox->prox->ant = ini;
48          ini->prox = ini->prox->prox;
49          free(lixo);
50      }else{
51          ini->prox = NULL;
52      }
53  }
```

Lista Duplamente Encadeada
Remove no Fim



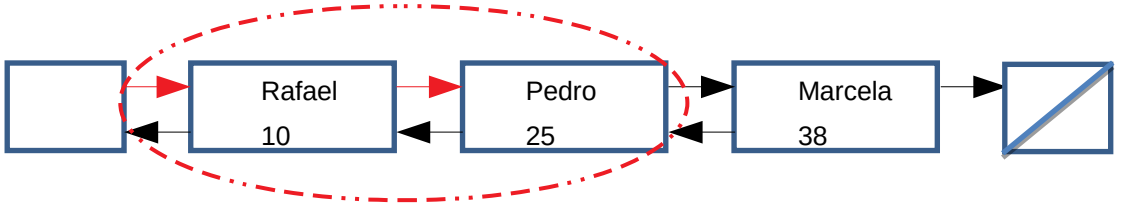

```
54    // Remove elemento no fim da lista
55    void removeFim(celula *ini) {
56        if(ini->prox == NULL){
57            printf("Lista Vazia!\n");
58        }else if(ini->prox->prox == NULL){
59            celula *lixo;
60            lixo = ini->prox;
61            ini->prox = NULL;
62            free(lixo);
63        }else{
64            removeFim(ini->prox);
65        }
66    }
67
```

Lista Duplamente Encadeada
Remove no Fim



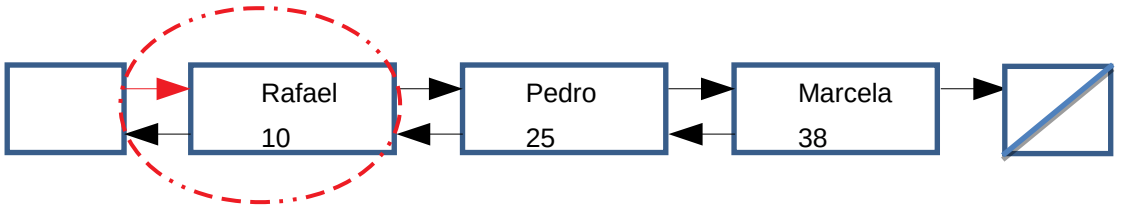

```
54  // Remove elemento no fim da lista
55  void removeFim(celula *ini) {
56      if(ini->prox == NULL){
57          printf("Lista Vazia!\n");
58      }else if(ini->prox->prox == NULL){
59          celula *lixo;
60          lixo = ini->prox;
61          ini->prox = NULL;
62          free(lixo);
63      }else{
64          removeFim(ini->prox);
65      }
66  }
67
```

Lista Duplamente Encadeada
Remove no Fim



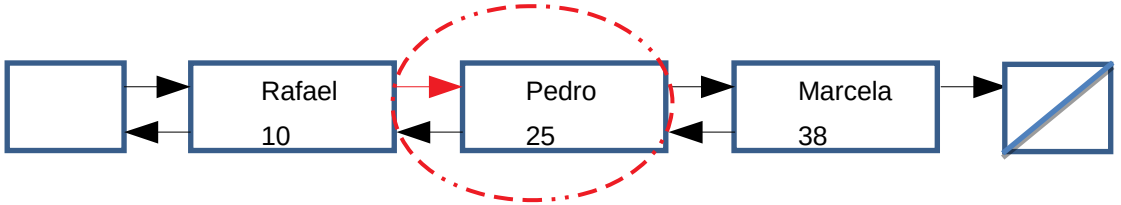

```
54  // Remove elemento no fim da lista
55  void removeFim(celula *ini) {
56      if(ini->prox == NULL){
57          printf("Lista Vazia!\n");
58      }else if(ini->prox->prox == NULL){
59          celula *lixo;
60          lixo = ini->prox;
61          ini->prox = NULL;
62          free(lixo);
63      }else{
64          removeFim(ini->prox);
65      }
66  }
67
```

Lista Duplamente Encadeada
Remove no Fim



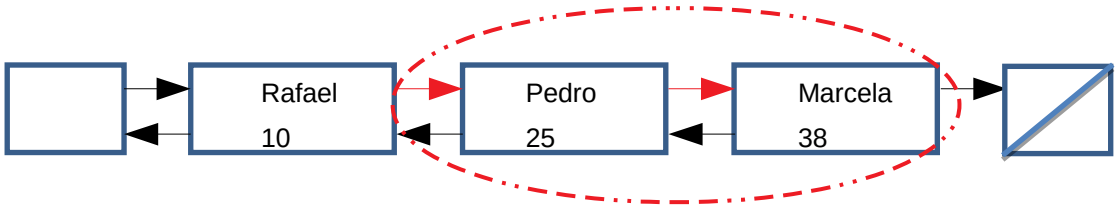

```
54  // Remove elemento no fim da lista
55  void removeFim(celula *ini) {
56      if(ini->prox == NULL){
57          printf("Lista Vazia!\n");
58      }else if(ini->prox->prox == NULL){
59          celula *lixo;
60          lixo = ini->prox;
61          ini->prox = NULL;
62          free(lixo);
63      }else{
64          removeFim(ini->prox);
65      }
66  }
67
```

Lista Duplamente Encadeada
Remove no Fim



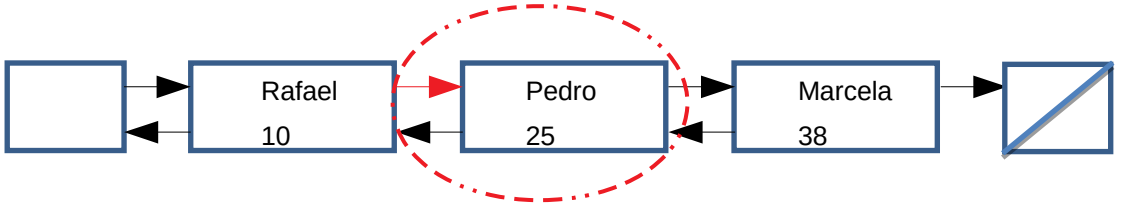

```
54    // Remove elemento no fim da lista
55    void removeFim(celula *ini) {
56        if(ini->prox == NULL){
57            printf("Lista Vazia!\n");
58        }else if(ini->prox->prox == NULL){
59            celula *lixo;
60            lixo = ini->prox;
61            ini->prox = NULL;
62            free(lixo);
63        }else{
64            removeFim(ini->prox);
65        }
66    }
67
```

Lista Duplamente Encadeada
Remove no Fim



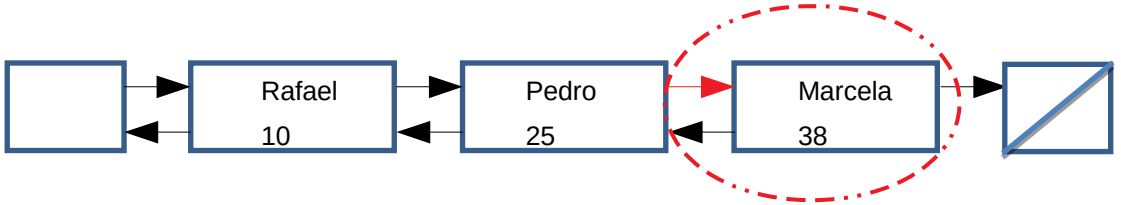

```
54  // Remove elemento no fim da lista
55  void removeFim(celula *ini) {
56      if(ini->prox == NULL){
57          printf("Lista Vazia!\n");
58      }else if(ini->prox->prox == NULL){
59          celula *lixo;
60          lixo = ini->prox;
61          ini->prox = NULL;
62          free(lixo);
63      }else{
64          removeFim(ini->prox);
65      }
66  }
67
```

Lista Duplamente Encadeada
Remove no Fim



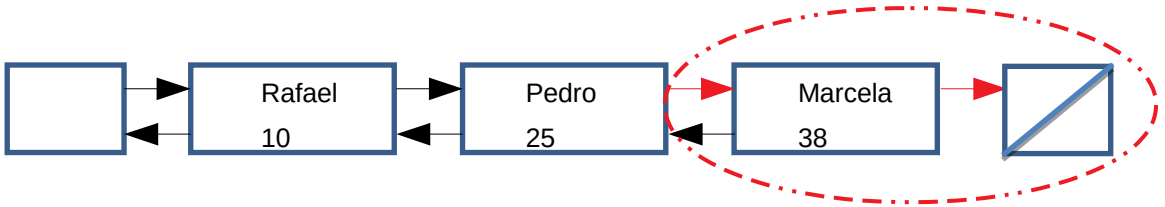

```
54    // Remove elemento no fim da lista
55    void removeFim(celula *ini) {
56        if(ini->prox == NULL){
57            printf("Lista Vazia!\n");
58        }else if(ini->prox->prox == NULL){
59            celula *lixo;
60            lixo = ini->prox;
61            ini->prox = NULL;
62            free(lixo);
63        }else{
64            removeFim(ini->prox);
65        }
66    }
67
```

Lista Duplamente Encadeada
Remove no Fim



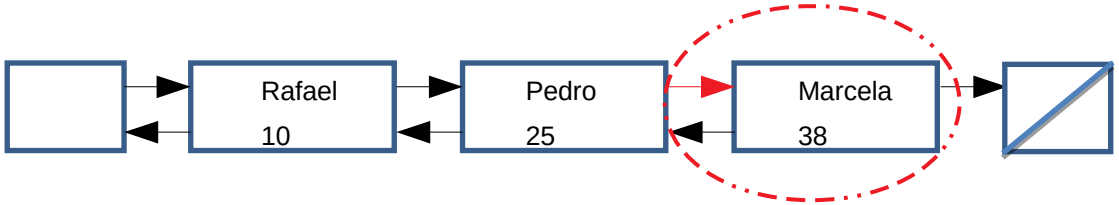

```
54    // Remove elemento no fim da lista
55    void removeFim(celula *ini) {
56        if(ini->prox == NULL){
57            printf("Lista Vazia!\n");
58        }else if(ini->prox->prox == NULL){
59            celula *lixo;
60            lixo = ini->prox;
61            ini->prox = NULL;
62            free(lixo);
63        }else{
64            removeFim(ini->prox);
65        }
66    }
67
```

Lista Duplamente Encadeada
Remove no Fim




```
54  // Remove elemento no fim da lista
55  void removeFim(celula *ini) {
56      if(ini->prox == NULL){
57          printf("Lista Vazia!\n");
58      }else if(ini->prox->prox == NULL){
59          celula *lixo;
60          lixo = ini->prox;
61          ini->prox = NULL;
62          free(lixo);
63      }else{
64          removeFim(ini->prox);
65      }
66  }
67
```

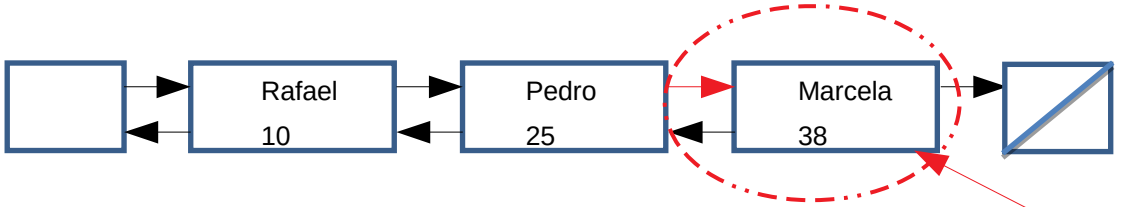
Lista Duplamente Encadeada
Remove no Fim




```
54   // Remove elemento no fim da lista
55   void removeFim(celula *ini) {
56       if(ini->prox == NULL){
57           printf("Lista Vazia!\n");
58       }else if(ini->prox->prox == NULL){
59           celula *lixo;
60           lixo = ini->prox;
61           ini->prox = NULL;
62           free(lixo);
63       }else{
64           removeFim(ini->prox);
65       }
66   }
67
```

LIXO

Lista Duplamente Encadeada
Remove no Fim

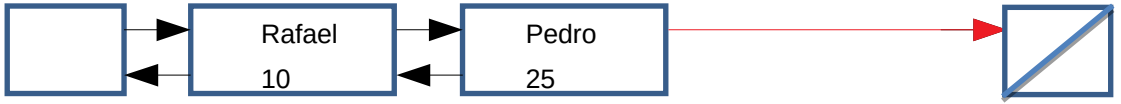



```
54  // Remove elemento no fim da lista
55  void removeFim(celula *ini) {
56      if(ini->prox == NULL){
57          printf("Lista Vazia!\n");
58      }else if(ini->prox->prox == NULL){
59          celula *lixo;
60          lixo = ini->prox;
61          ini->prox = NULL;
62          free(lixo);
63      }else{
64          removeFim(ini->prox);
65      }
66  }
67
```

Lista Duplamente Encadeada
Remove no Fim

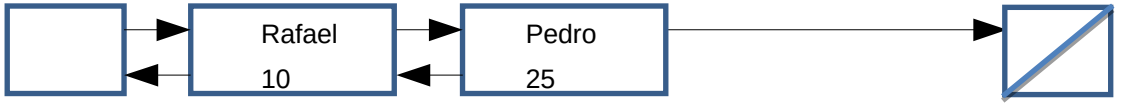

```
54    // Remove elemento no fim da lista
55    void removeFim(celula *ini) {
56        if(ini->prox == NULL){
57            printf("Lista Vazia!\n");
58        }else if(ini->prox->prox == NULL){
59            celula *lixo;
60            lixo = ini->prox;
61            ini->prox = NULL;
62            free(lixo);
63        }else{
64            removeFim(ini->prox);
65        }
66    }
67
```

Lista Duplamente Encadeada
Remove no Fim



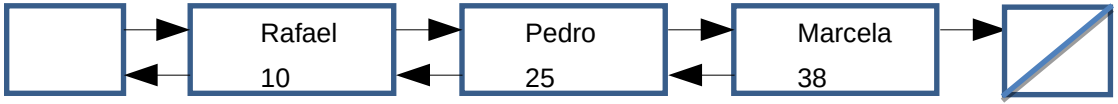

```
54    // Remove elemento no fim da lista
55    void removeFim(celula *ini) {
56        if(ini->prox == NULL){
57            printf("Lista Vazia!\n");
58        }else if(ini->prox->prox == NULL){
59            celula *lixo;
60            lixo = ini->prox;
61            ini->prox = NULL;
62            free(lixo);
63        }else{
64            removeFim(ini->prox);
65        }
66    }
67
```

Lista Duplamente Encadeada
Remove no Fim

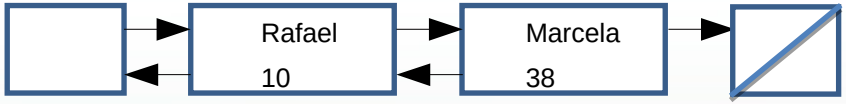



```
54  // Remove elemento no fim da lista
55  void removeFim(celula *ini) {
56      if(ini->prox == NULL){
57          printf("Lista Vazia!\n");
58      }else if(ini->prox->prox == NULL){
59          celula *lixo;
60          lixo = ini->prox;
61          ini->prox = NULL;
62          free(lixo);
63      }else{
64          removeFim(ini->prox);
65      }
66  }
67
```

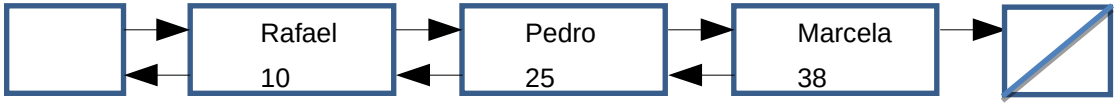
Lista Duplamente Encadeada
Atividade - Busca e Remove



- Desenvolva uma função para remover a primeira célula que contenha a idade passada por parâmetro.
- Exemplo.
  - removeIdade(25, ini);



Lista Duplamente Encadeada
Atividade - Busca e Remove



- Desenvolva uma função para remover a primeira célula que contenha o mome passado por parâmetro.
- Exemplo.
  - removeNome(“Rafael”, ini);