Unidade IV:
Estruturas de Dados Básicas com Alocação Sequencial

Prof. Max do Val Machado

Agenda

- Lista
- Pilha
- Fila

Agenda

- **Lista**
- Pilha
- Fila

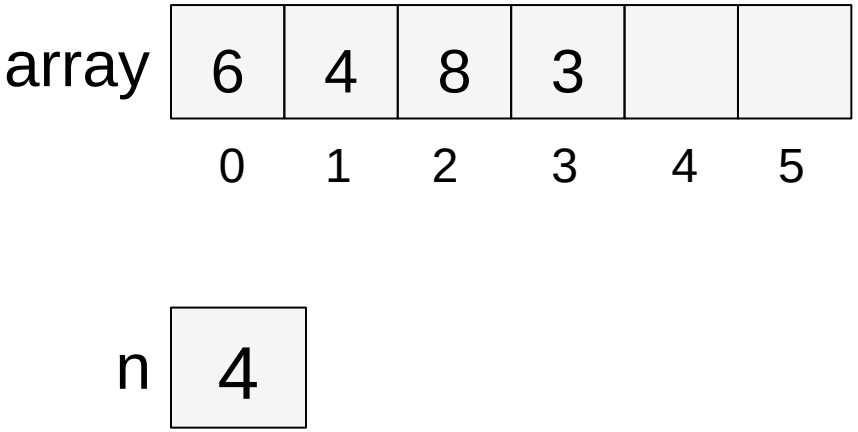
Introdução

- As listas são um Tipo Abstrato de Dados (TAD) no qual podemos inserir e remover elementos em qualquer posição

- Exemplos:
  - Lista de valores (*array* de números inteiros)
  - Lista de nomes (*array* de strings)
  - Lista de notas (*array* de números reais)
  - Lista de carros (*array* de objetos do tipo carro)

Variáveis da Lista

- array (de elementos) e n (contador)



Métodos da Lista

- Construtor
- Inserção de elemento
  - void inserirInicio(elemento)
  - void inserirFim(elemento)
  - void inserir(elemento, posição)
- Remoção de elementos
  - elemento removerInicio()
  - elemento removerFim()
  - elemento remover(posição)
- Mostrar, pesquisar, ordenar, ...

Exemplo

- Supondo a existência da TAD, vamos executar o exemplo ilustrado a seguir

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

Lista

Tela

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

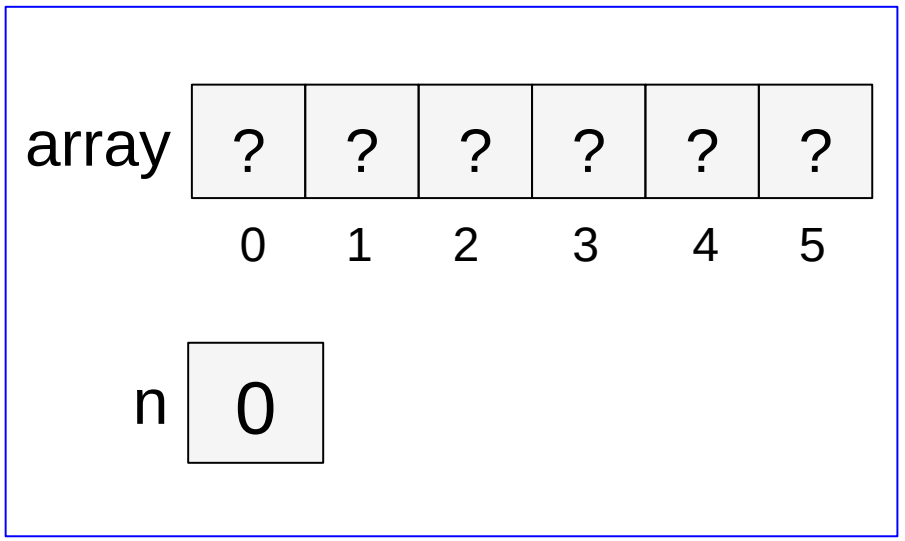
Lista

Tela

```
==== LISTA ESTATICA ====
```

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```



Tela

```
==== LISTA ESTATICA ====
```

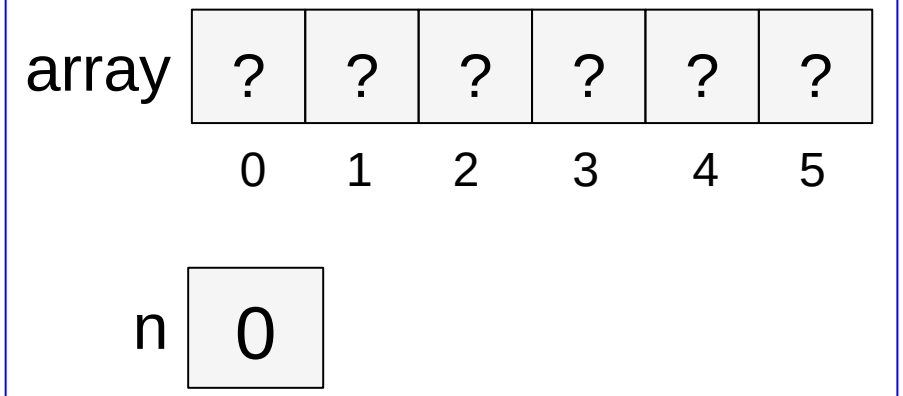
Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1 [ ]

x2 [ ]

x3 [ ]



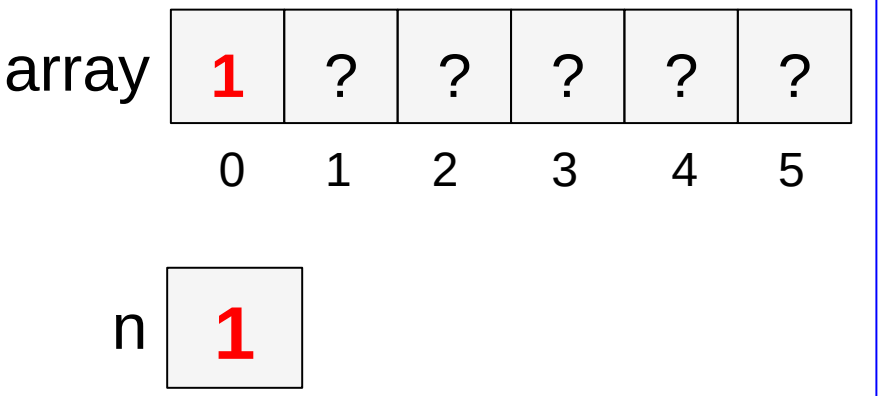
Tela

```
==== LISTA ESTATICA ====
```

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1 x2 x3



Tela

```
==== LISTA ESTATICA ====
```

Exemplo

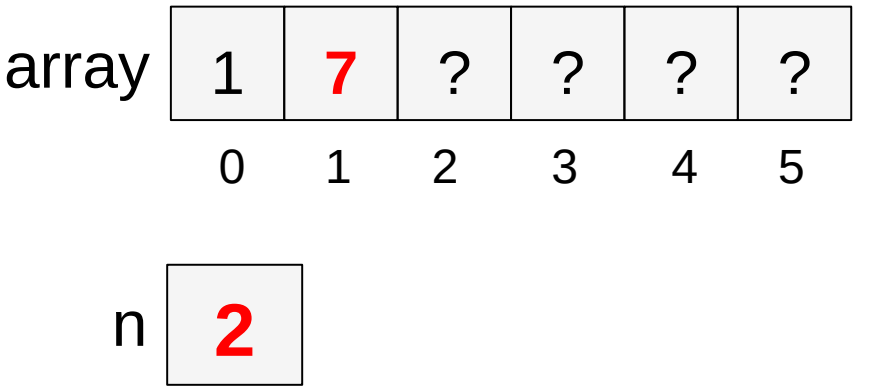
```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

Fim da não do *array*)

x1 ☐

x2 ☐

x3 ☐



Tela

==== LISTA ESTATICA ====

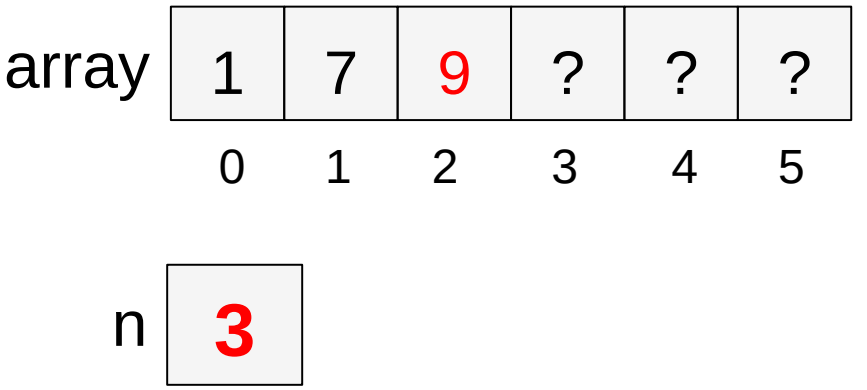
Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1

x2

x3



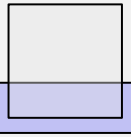
Tela

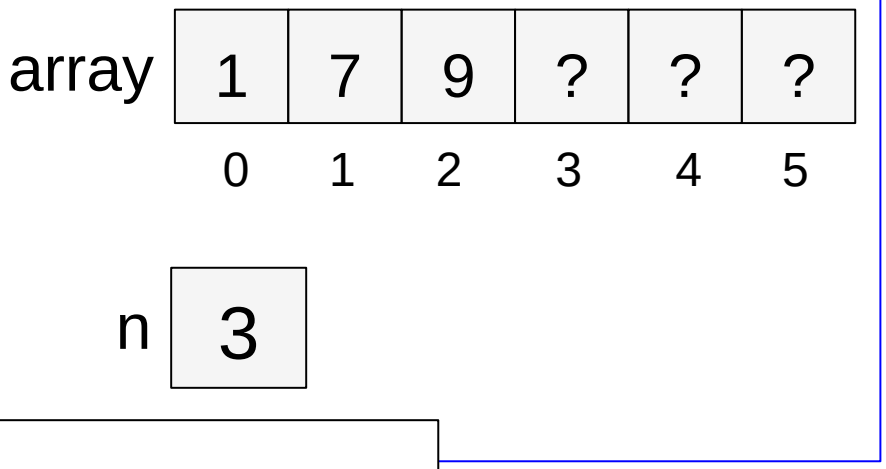
```
==== LISTA ESTATICA ====
```

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1



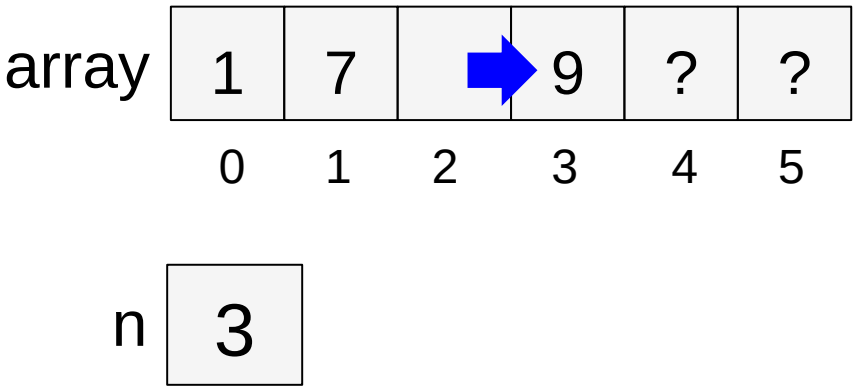


Antes dessa inserção, liberaremos a posição 0, deslocando os elementos para a direita

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1

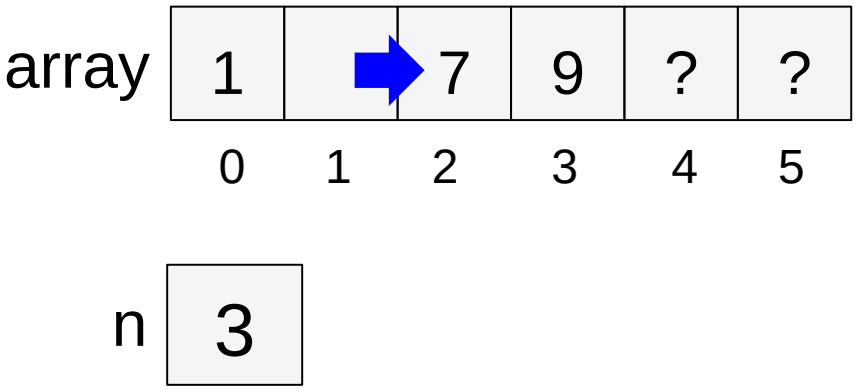


Antes dessa inserção, liberaremos a posição 0, deslocando os elementos para a direita

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1

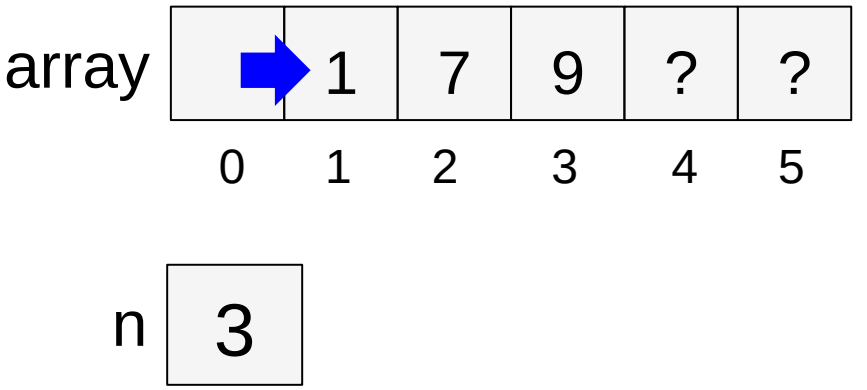


Antes dessa inserção, liberaremos a posição 0, deslocando os elementos para a direita

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1

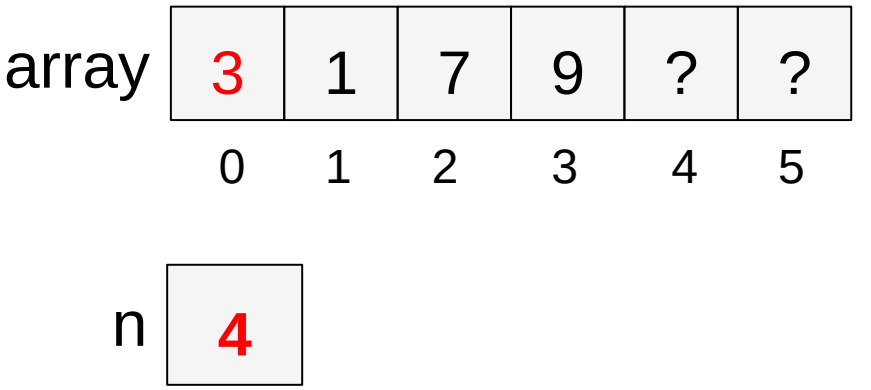


Antes dessa inserção, liberaremos a posição 0, deslocando os elementos para a direita

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1 x2 x3



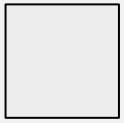
Tela

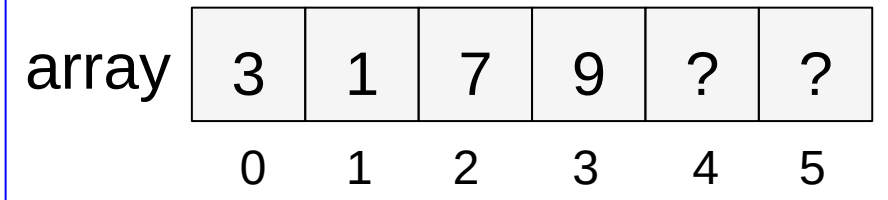
```
==== LISTA ESTATICA ====
```

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1





n 4

Precisamos liberar a posição 3

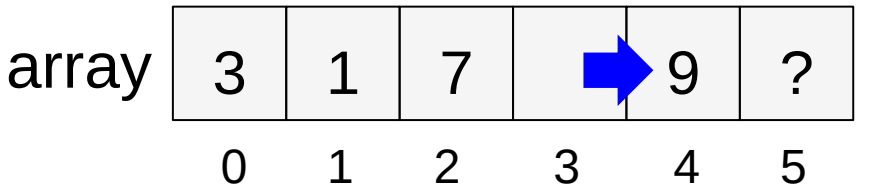
Tela

==== LISTA ESTATICA ====

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1



n 4

Precisamos liberar a posição 3

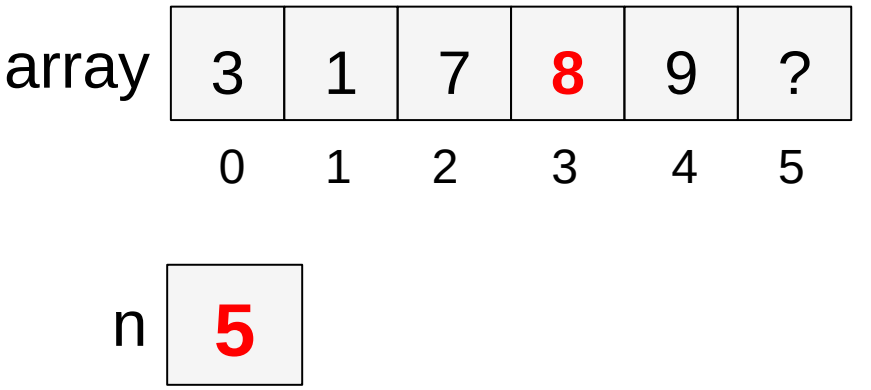
...ela

==== LISTA ESTATICA ====

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1 x2 x3



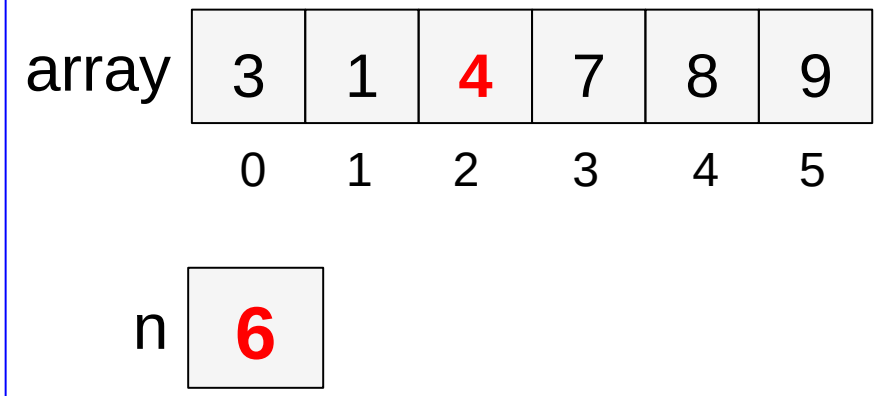
Tela

==== LISTA ESTATICA ====

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1 x2 x3



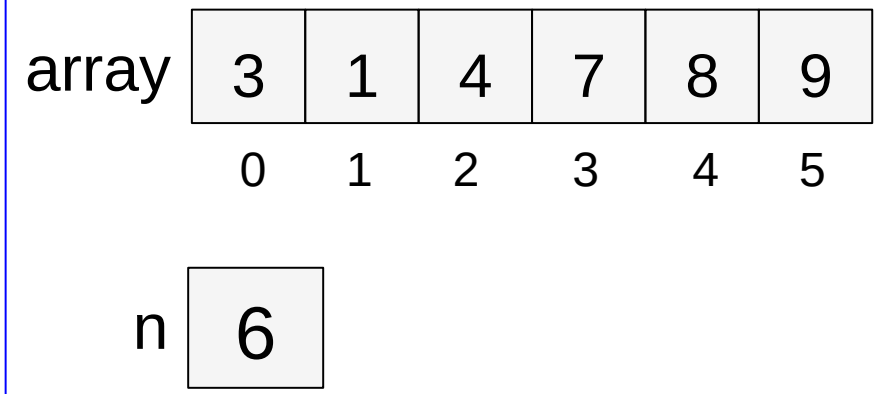
Tela

```
==== LISTA ESTATICA ====
```

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1 x2 x3



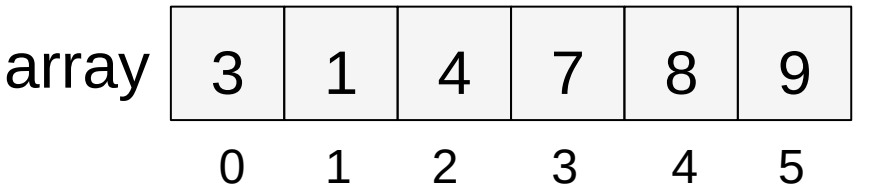
Tela

```
==== LISTA ESTATICA ====
[ 3 1 4 7 8 9 ]
```

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1 [ ]



n 6

Após a remoção do 3, os demais elementos devem ser deslocados para a esquerda

la

CA ====

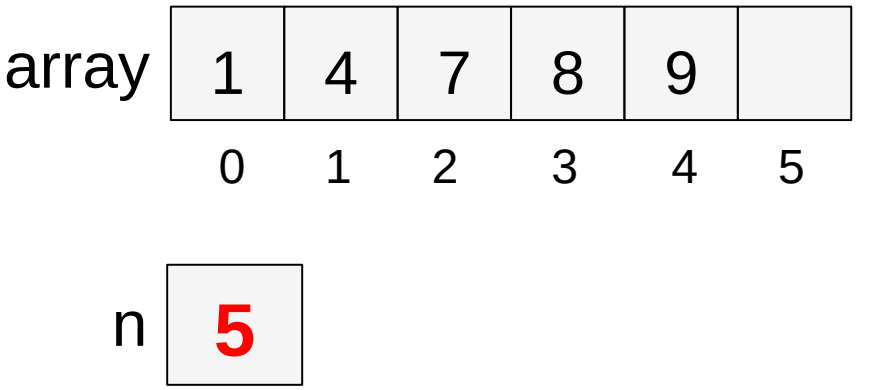
Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1 **3**

x2

x3



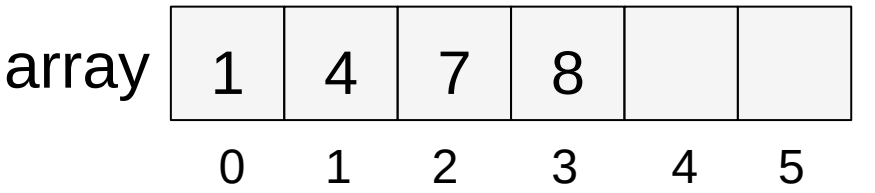
Tela

```
==== LISTA ESTATICA ====
[ 3  1  4  7  8  9 ]
```

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1	3
x2	9
x3	



n = **4**

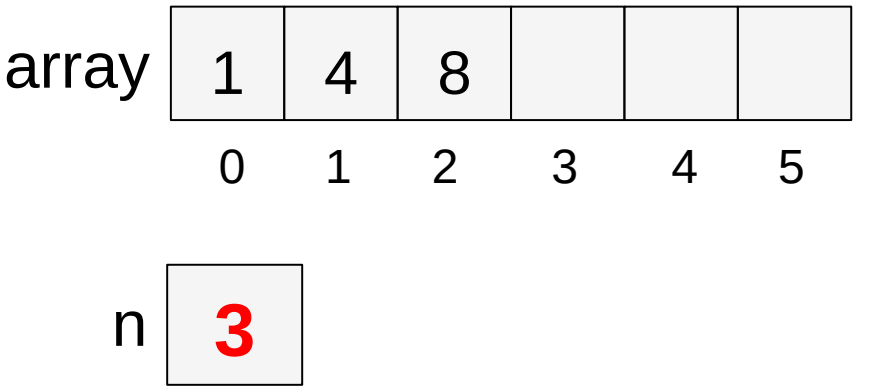
Tela

```
==== LISTA ESTATICA ====
[ 3  1  4  7  8  9 ]
```

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1	3
x2	9
x3	7



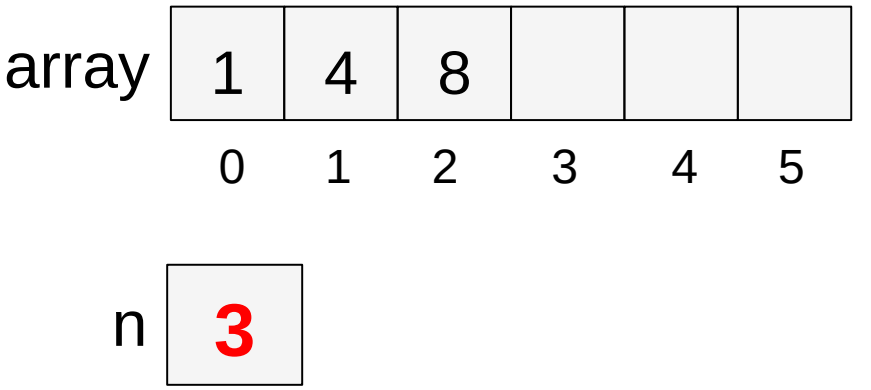
Tela

```
==== LISTA ESTATICA ====
[ 3  1  4  7  8  9 ]
```

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1	3
x2	9
x3	7



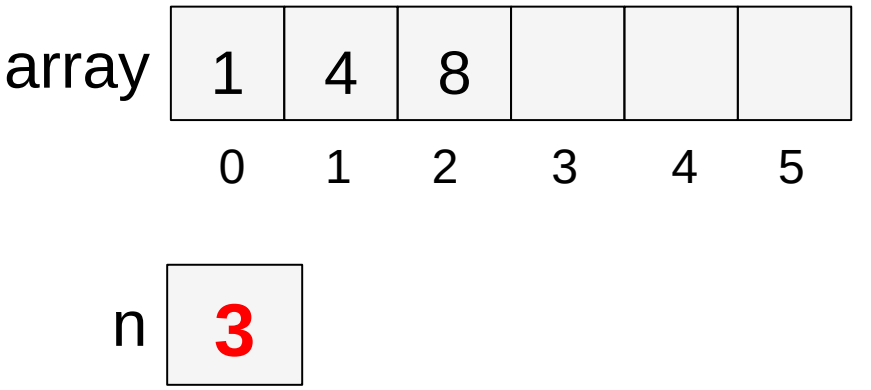
Tela

```
==== LISTA ESTATICA ====
[ 3  1  4  7  8  9 ]
3, 9, 7
```

Exemplo

```
printf("==== LISTA ESTATICA ====");
start();
int x1, x2, x3;
inserirInicio(1);
inserirFim(7);
inserirFim(9);
inserirInicio(3);
inserir(8, 3);
inserir(4, 2);

mostrar();

x1 = removerInicio();
x2 = removerFim();
x3 = remover(2);

printf("%i, %i, %i", x1, x2, x3);
mostrar();
```

x1	3
x2	9
x3	7



Tela

```
==== LISTA ESTATICA ====
[ 3 1 4 7 8 9 ]
3, 9, 7
[ 1 4 8 ]
```

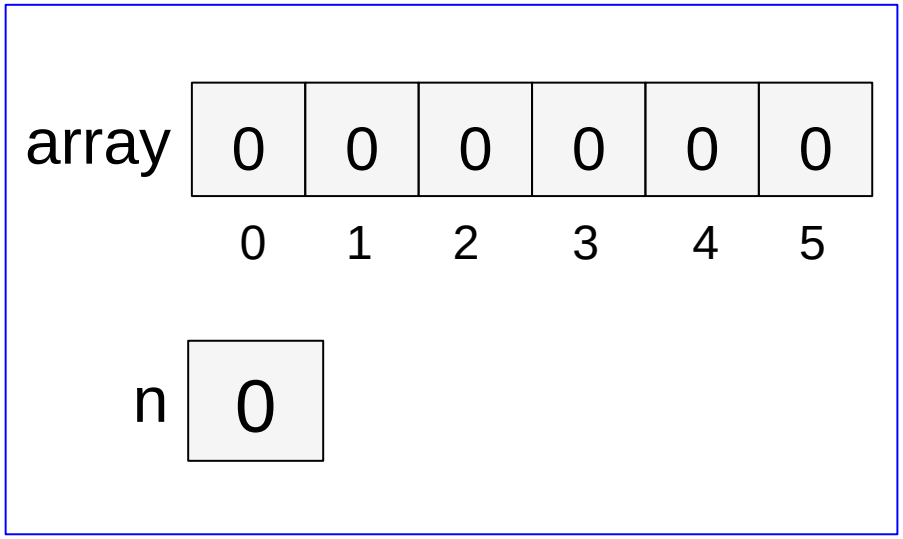
Algoritmo em C

```
int array[MAXTAM];
int n;


void start(){
    n = 0;
}

void inserirInicio(int x)   {  ...  }
void inserirFim(int x)   {  ...  }
void inserir(int x, int pos)   {  ...  }
int removerInicio()   {  ...  }
int removerFim()   {  ...  }
int remover(int pos)   {  ...  }
void mostrar () {  ...  }
```

Algoritmo em C

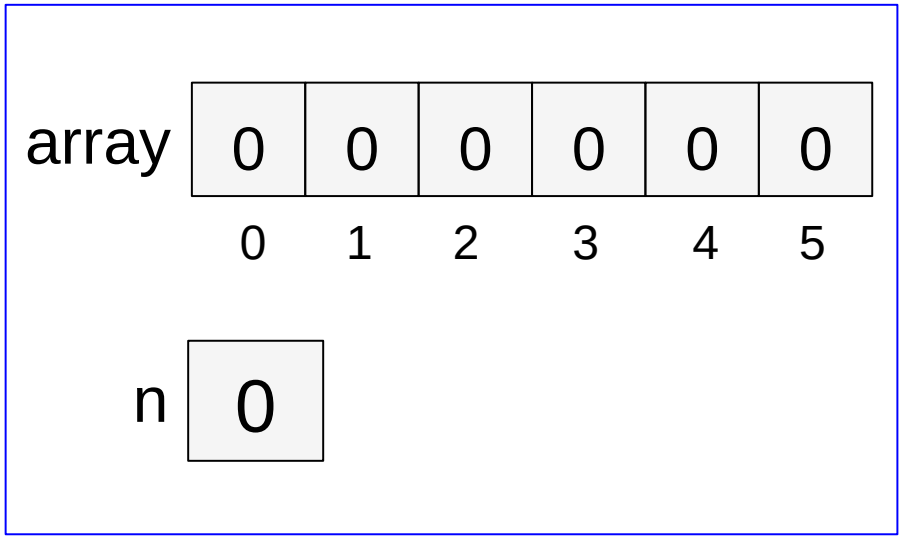
```
int array[MAXTAM];
int n;


void start(){
    n = 0;
}
```

```
void inserirInicio(int x)   {  ...  }
void inserirFim(int x)   {  ...  }
void inserir(int x, int pos)   {  ...  }
int removerInicio()   {  ...  }
int removerFim()   {  ...  }
int remover(int pos)   {  ...  }
void mostrar () {  ...  }
```



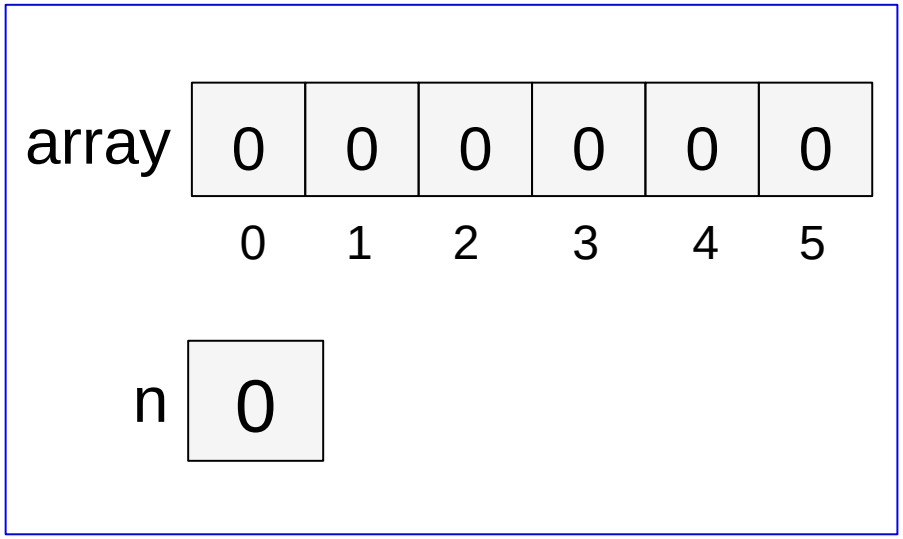
Algoritmo em C

```
int array[MAXTAM];
int n;


void start(){
    n = 0;
}

void inserirInicio(int x)    {  ...  }
void inserirFim(int x)    {  ...  }
void inserir(int x, int pos)    {  ...  }
int removerInicio()    {  ...  }
int removerFim()    {  ...  }
int remover(int pos)    {  ...  }
void mostrar () {  ...  }
```



Algoritmo em C

```
// Exemplo: inserirInicio(1)                    x  1
void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

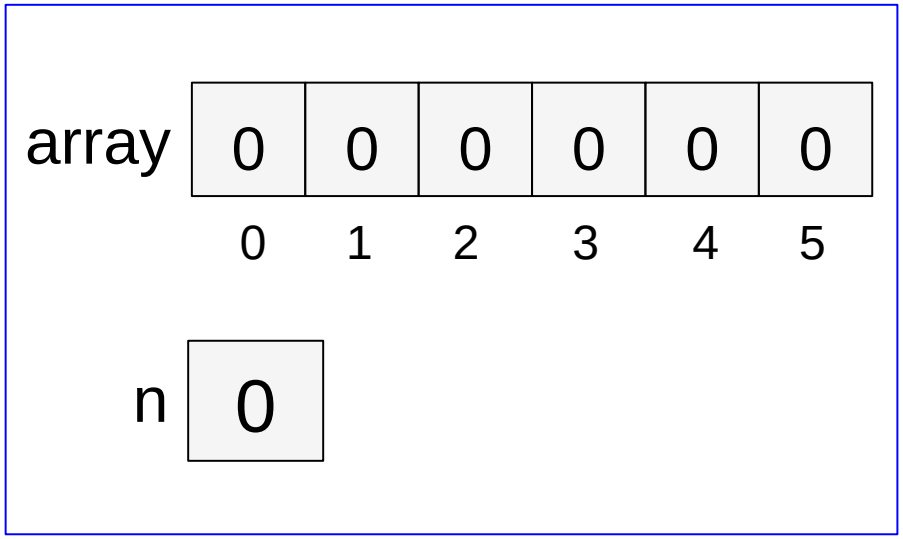


Algoritmo em C

```
// Exemplo: inserirInicio(1)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x 1

false: 0 >= 6

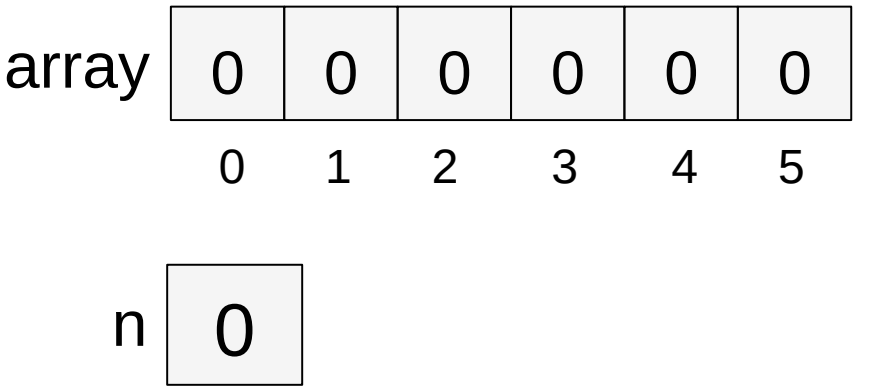


Algoritmo em C

```
// Exemplo: inserirInicio(1)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x 1

i 0



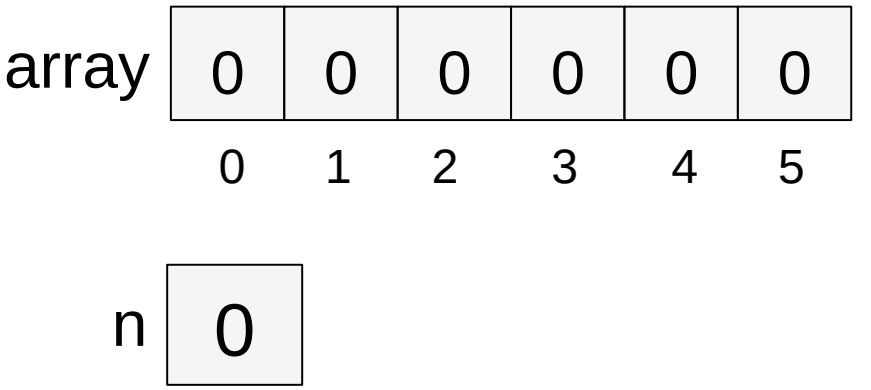
Algoritmo em C

```
// Exemplo: inserirInicio(1)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x 1

i 0

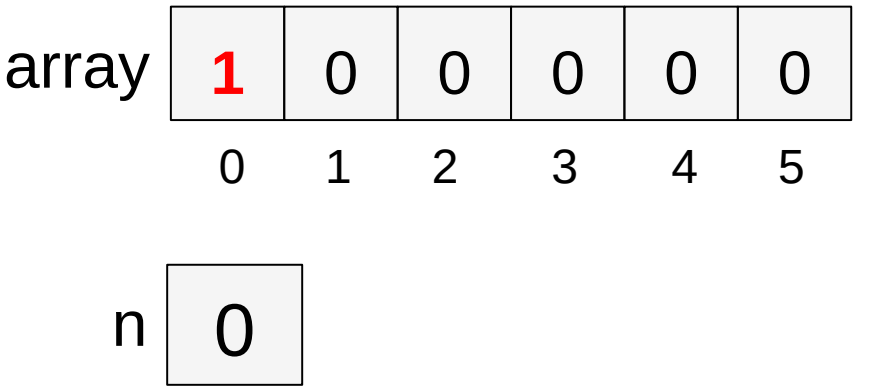
false: 0 > 0



Algoritmo em C

```
// Exemplo: inserirInicio(1)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

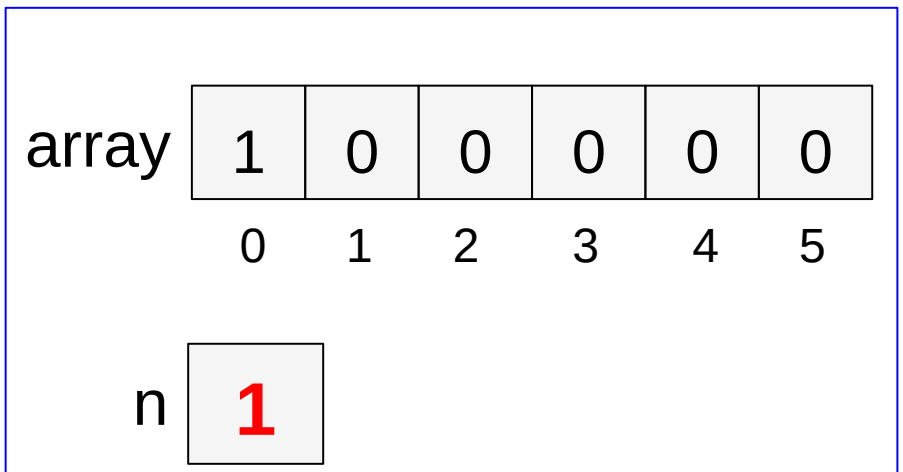
x 1



Algoritmo em C

```
// Exemplo: inserirInicio(1)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

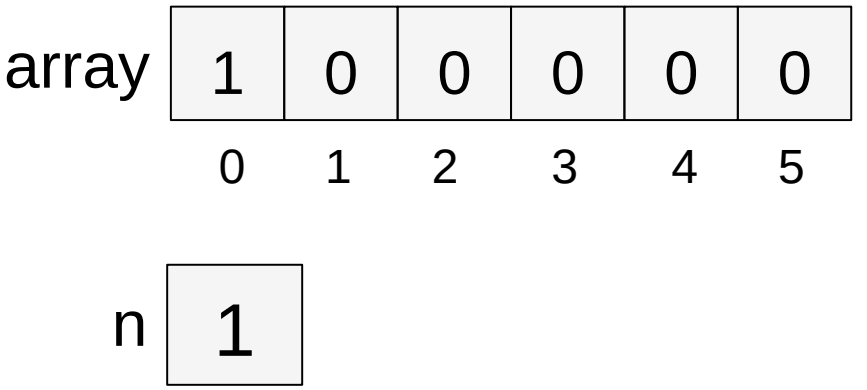
x 1



Algoritmo em C

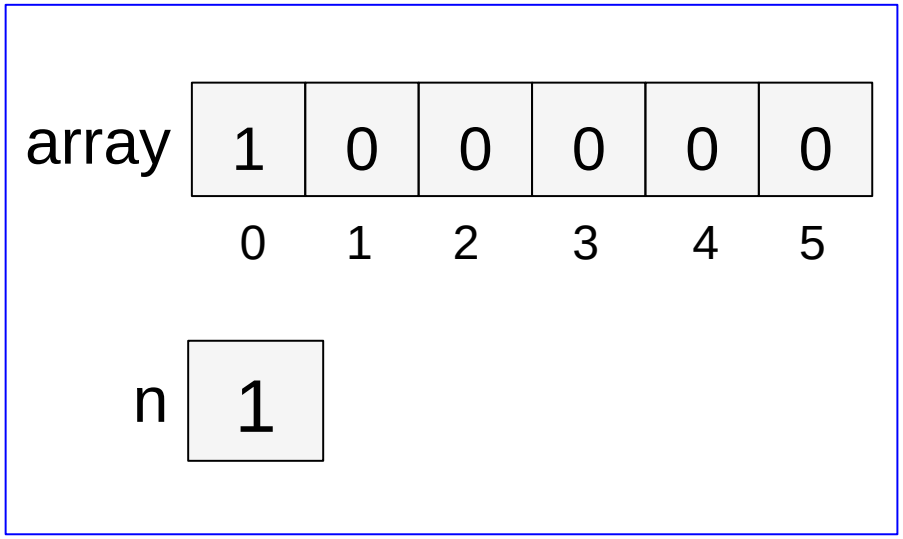
```
// Exemplo: inserirInicio(1)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x | 1



Algoritmo em C

```
// Exemplo: inserirInicio(3)                    x  3
void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

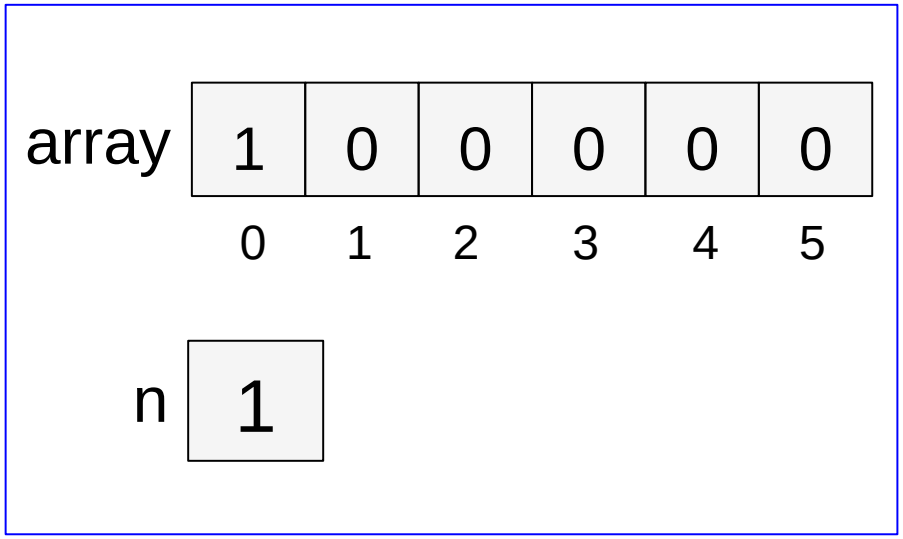


Algoritmo em C

```
// Exemplo: inserirInicio(3)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x 3

false: 1 >= 6

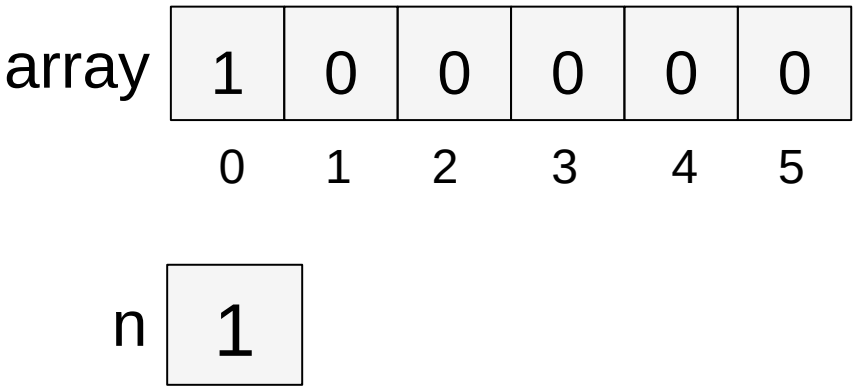


Algoritmo em C

```
// Exemplo: inserirInicio(3)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x 3

i 1



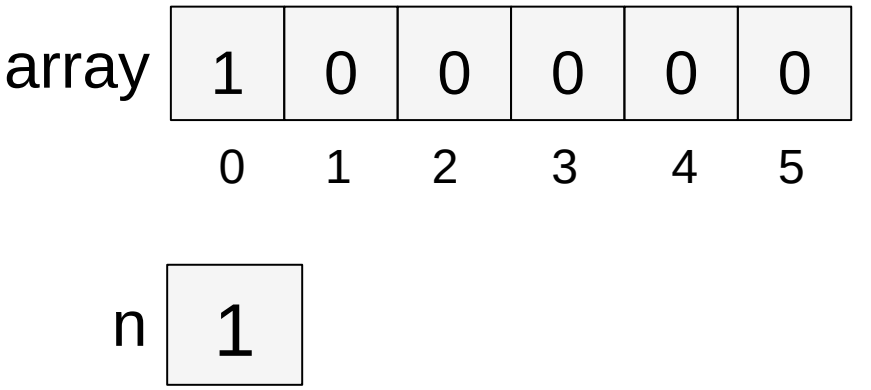
Algoritmo em C

```
// Exemplo: inserirInicio(3)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x 3

i 1

true: 1 > 0



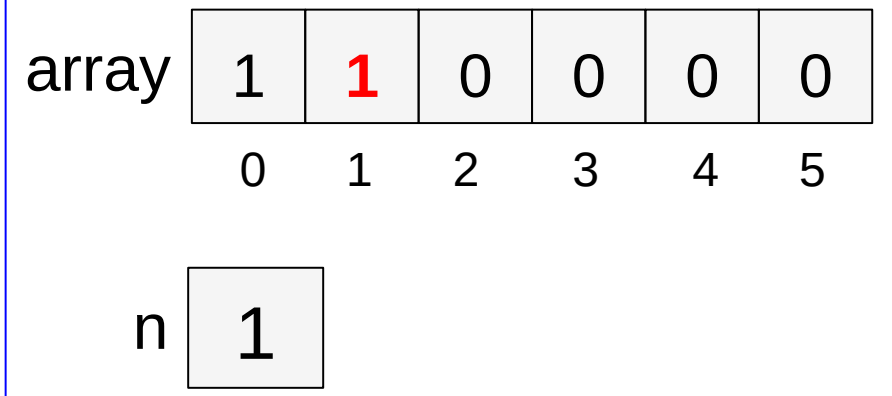
Algoritmo em C

```
// Exemplo: inserirInicio(3)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x 3

i 1

array[1] ← array[0]

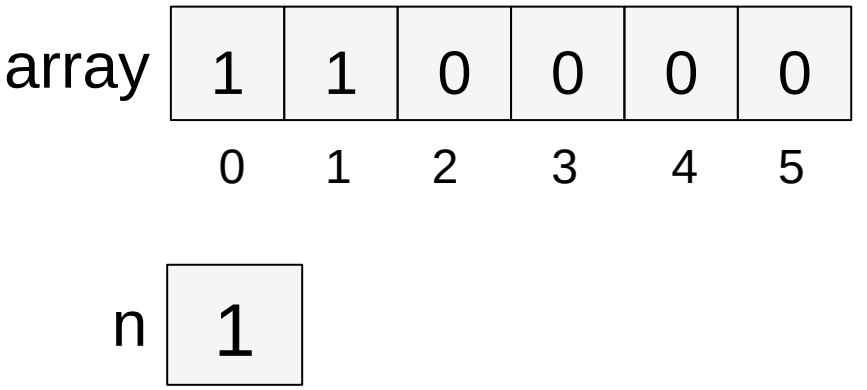


Algoritmo em C

```
// Exemplo: inserirInicio(3)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x 3

i 0



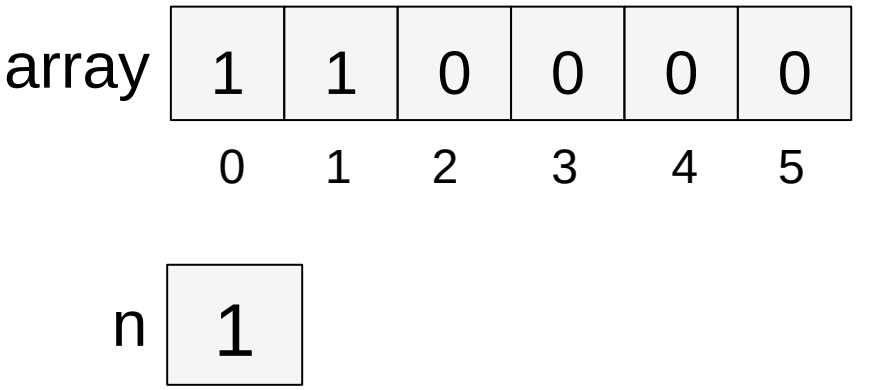
Algoritmo em C

```
// Exemplo: inserirInicio(3)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x 3

i 0

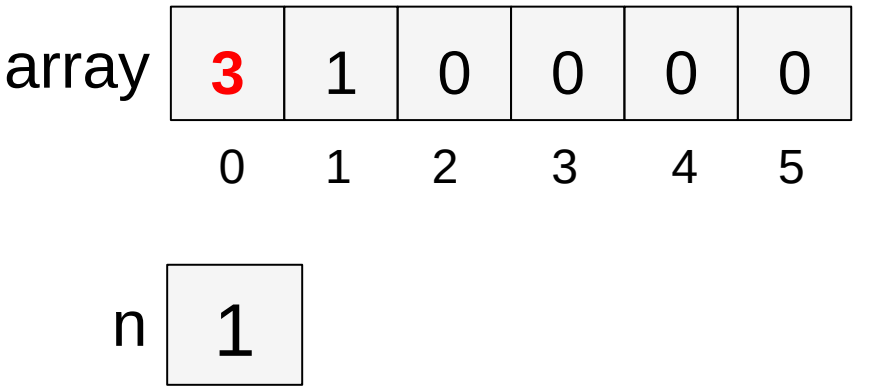
false: 0 > 0



Algoritmo em C

```
// Exemplo: inserirInicio(3)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

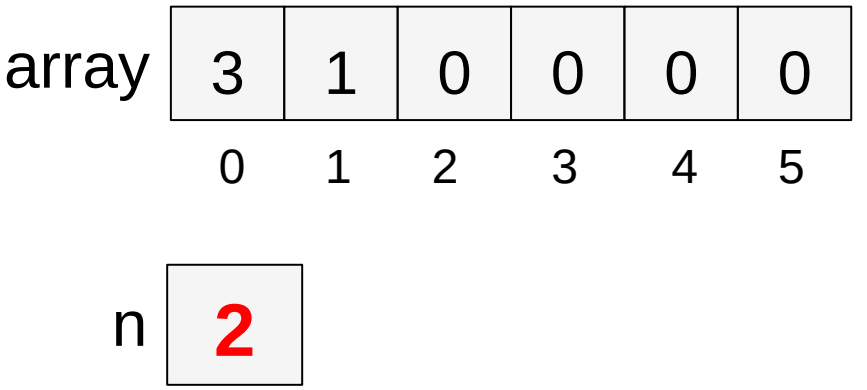
x 3



Algoritmo em C

```
// Exemplo: inserirInicio(3)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

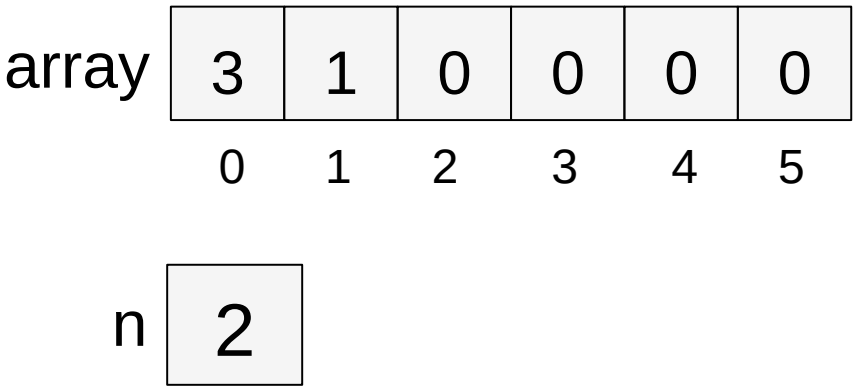
x | 3



Algoritmo em C

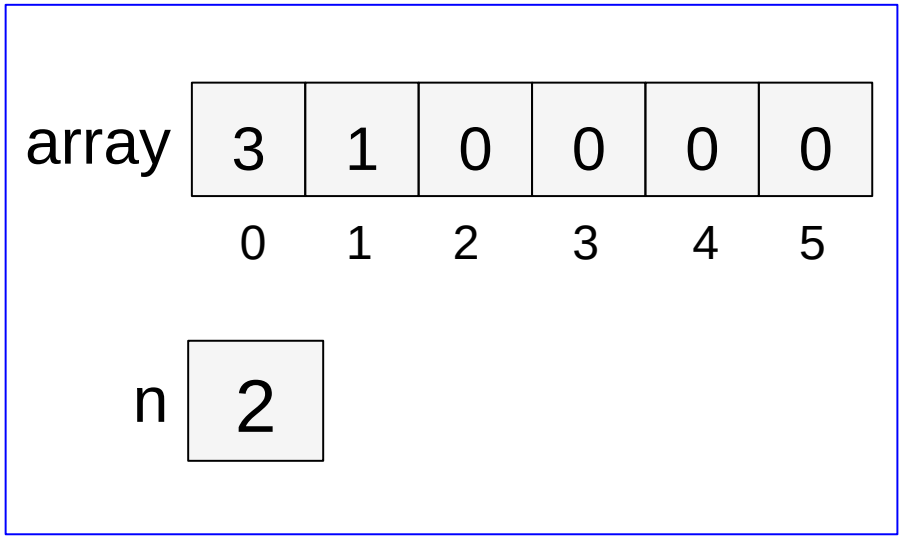
```
// Exemplo: inserirInicio(3)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x | 3



Algoritmo em C

```
// Exemplo: inserirInicio(5)                    x  5
void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

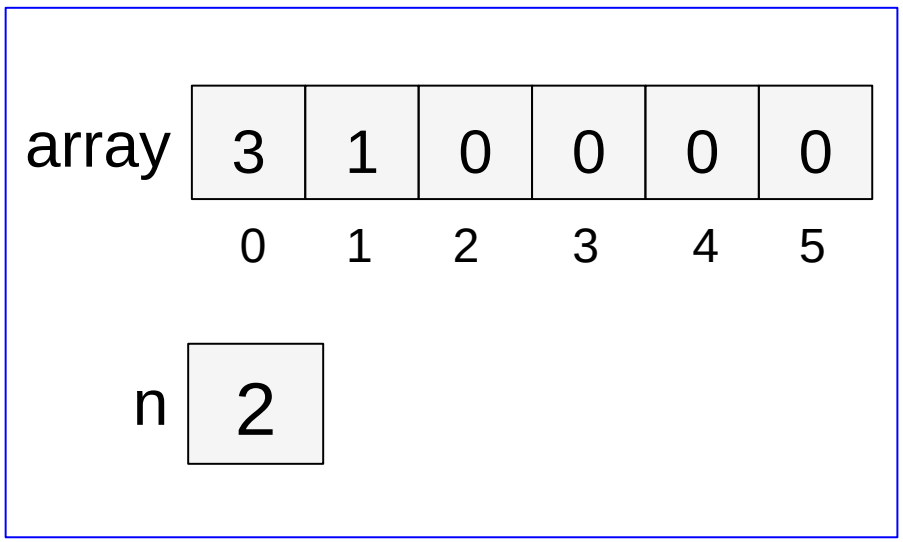


Algoritmo em C

```
// Exemplo: inserirInicio(5)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x 5

false: 2 >= 6

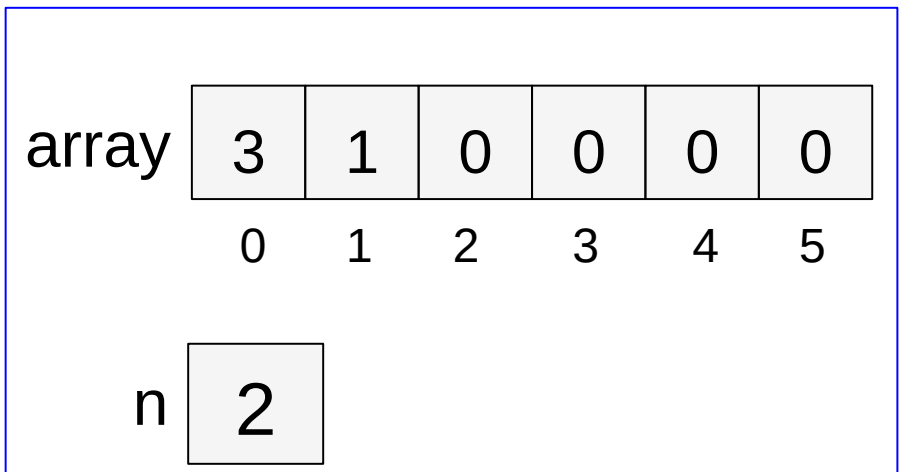


Algoritmo em C

```
// Exemplo: inserirInicio(5)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x 5

i 2



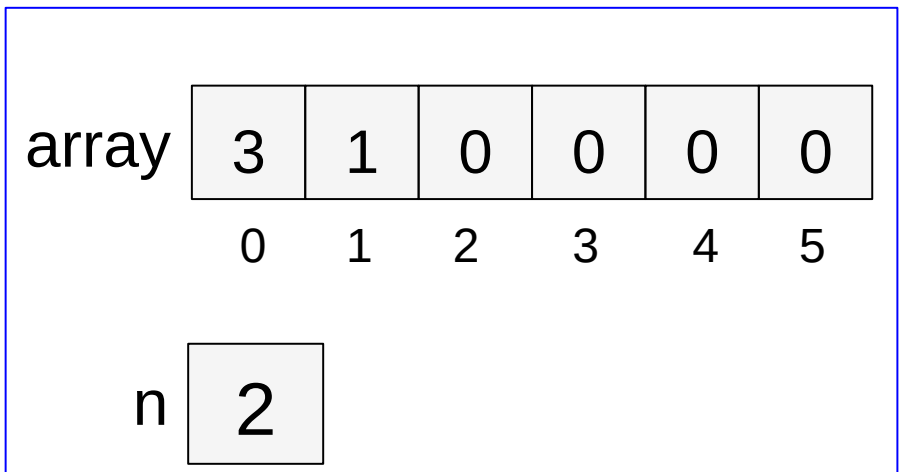
Algoritmo em C

```
// Exemplo: inserirInicio(5)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x 5

i 2

true: 2 > 0



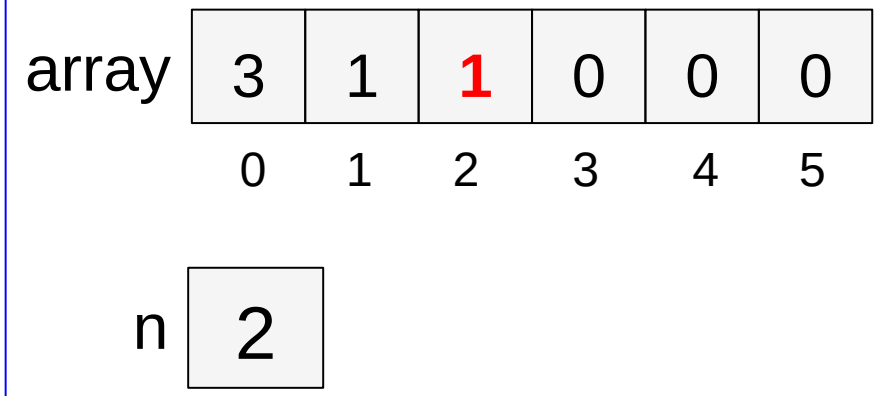
Algoritmo em C

```
// Exemplo: inserirInicio(5)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x 5

i 2

array[2] ← array[1]

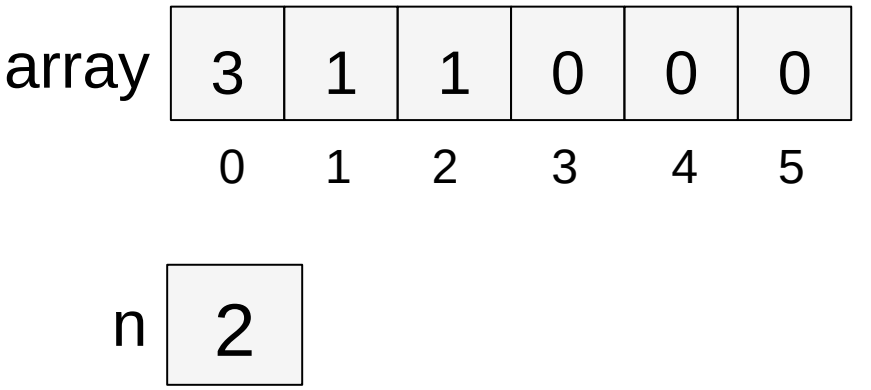


Algoritmo em C

```
// Exemplo: inserirInicio(5)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x 5

i 1



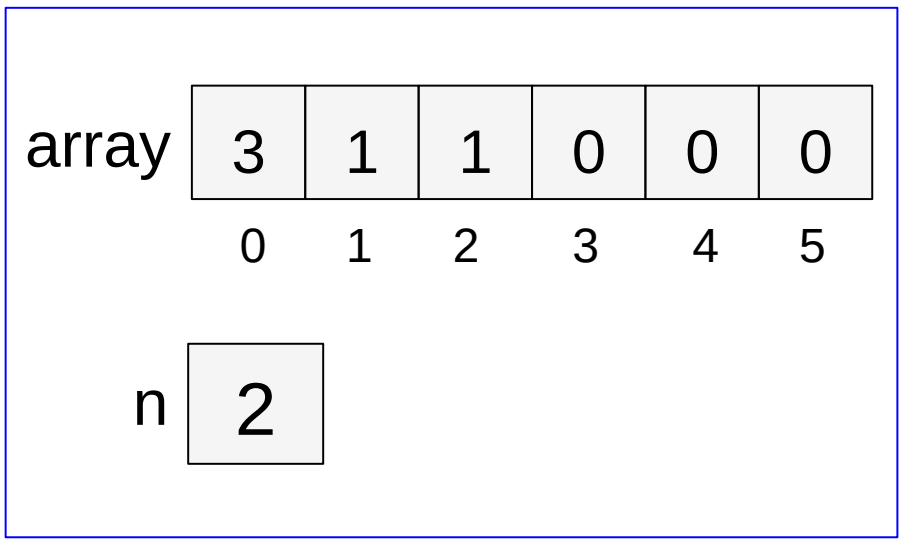
Algoritmo em C

```
// Exemplo: inserirInicio(5)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x 5

i 1

true: 1 > 0



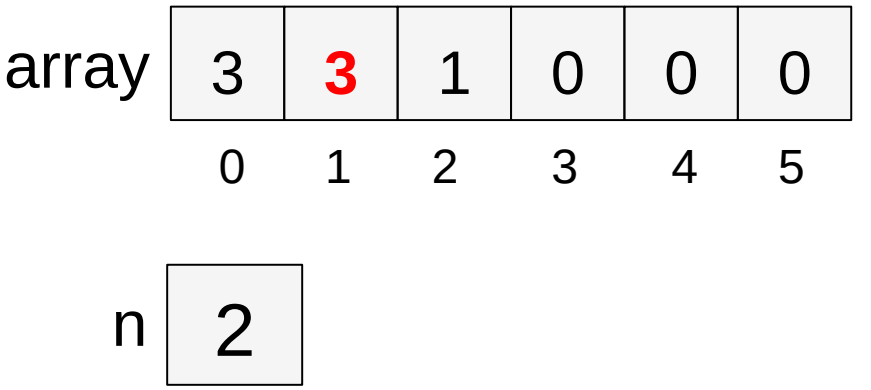
Algoritmo em C

```
// Exemplo: inserirInicio(5)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x 5

i 1

array[1] ← array[0]

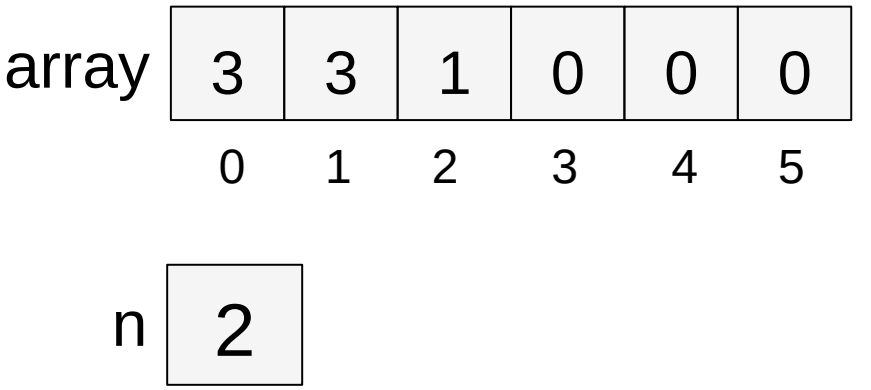


Algoritmo em C

```
// Exemplo: inserirInicio(5)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x 5

i 0



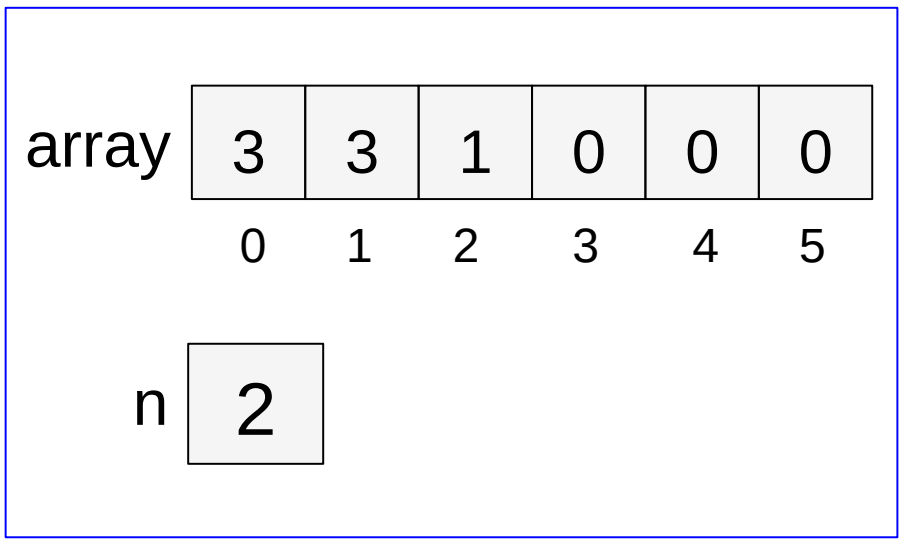
Algoritmo em C

```
// Exemplo: inserirInicio(5)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x 5

i 0

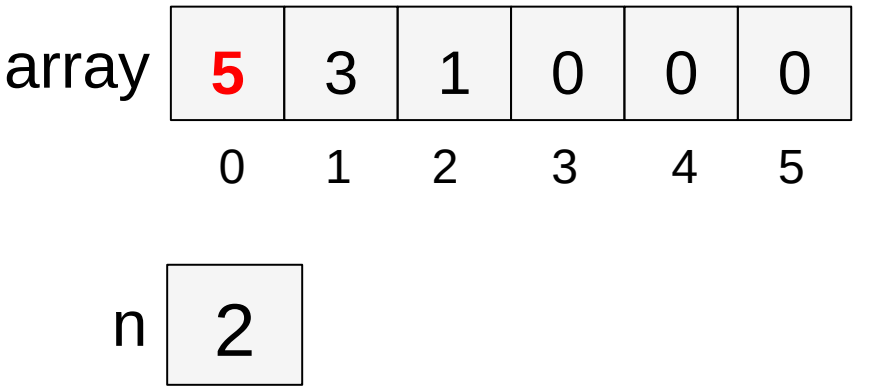
false: 0 > 0



Algoritmo em C

```
// Exemplo: inserirInicio(5)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

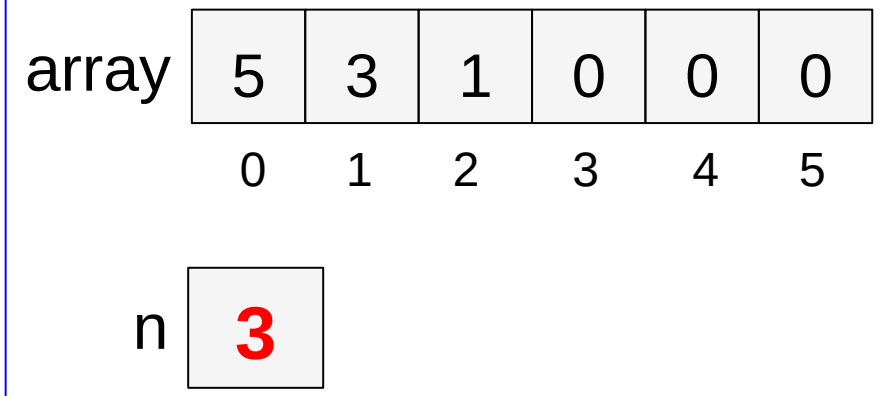
x 5



Algoritmo em C

```
// Exemplo: inserirInicio(5)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

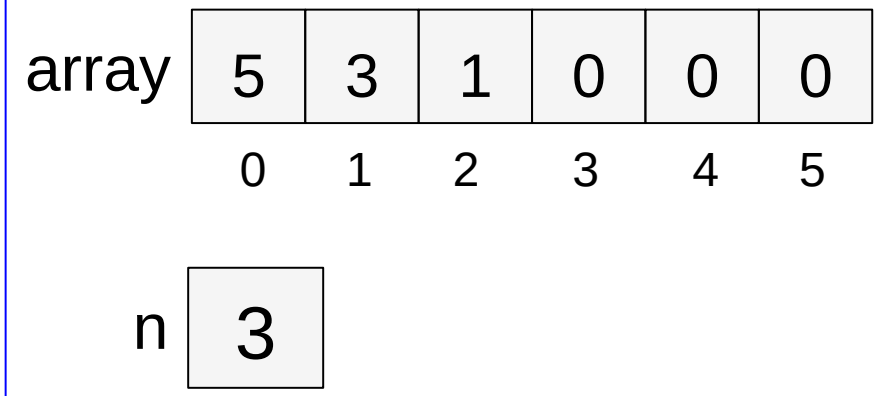
x | 5



Algoritmo em C

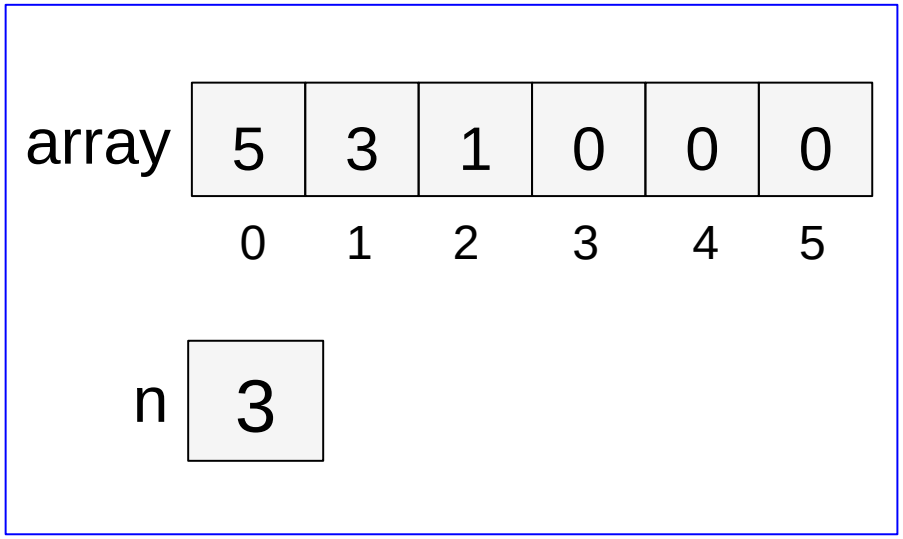
```
// Exemplo: inserirInicio(5)

void inserirInicio(int x) {

   if (n >= MAXTAM)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > 0; i--){
      array[i] = array[i-1];
   }

   array[0] = x;
   n++;
}
```

x | 5



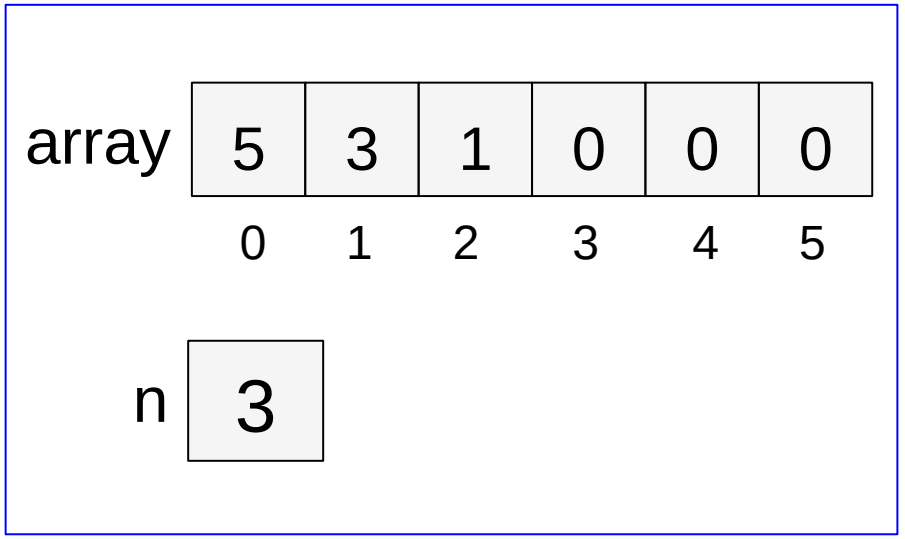
Algoritmo em C

```
int array[MAXTAM];
int n;


void start(){
    n = 0;
}

void inserirInicio(int x)   {  ...  }
void inserirFim(int x)   {  ...  }
void inserir(int x, int pos)   {  ...  }
int removerInicio()   {  ...  }
int removerFim()   {  ...  }
int remover(int pos)   {  ...  }
void mostrar () {  ...  }
```



Algoritmo em C

```
// Exemplo: inserirFim(9)                    x  9
void inserirFim(int x) {

   if (n >= MAXTAM)
      exit(1);

   array[n] = x;
   n++;
}
```

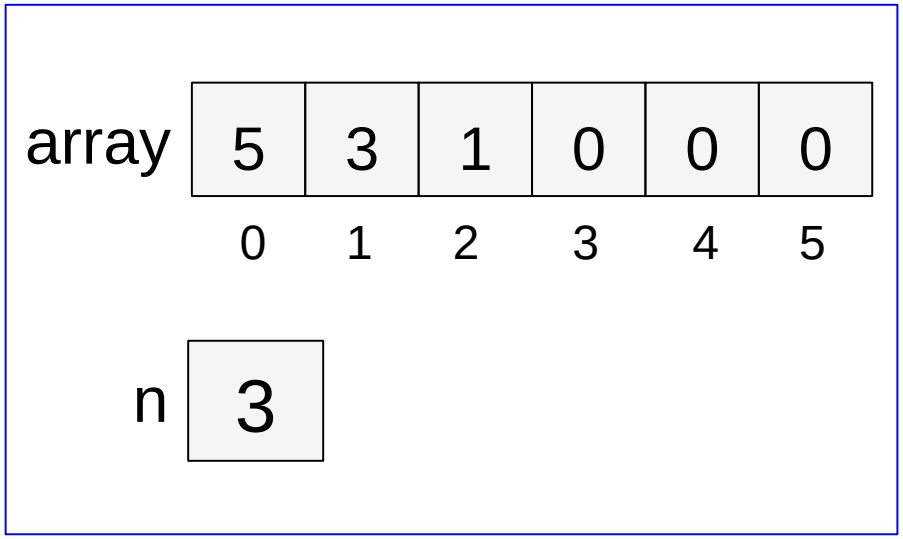


Algoritmo em C

```
// Exemplo: inserirFim(9)

void inserirFim(int x) {

   if (n >= MAXTAM)
      exit(1);

   array[n] = x;
   n++;
}
```

x 9

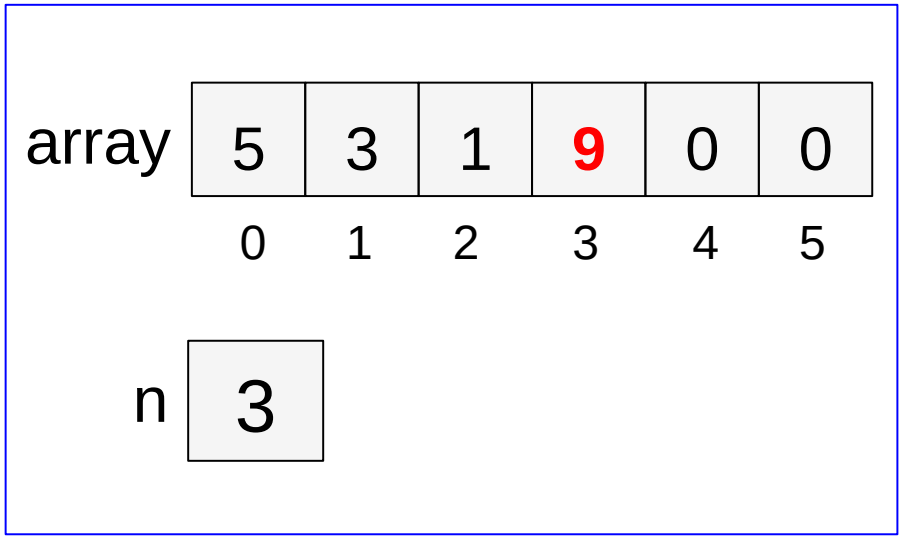
false: 3 >= 6



Algoritmo em C

```
// Exemplo: inserirFim(9)

void inserirFim(int x) {

   if (n >= MAXTAM)
      exit(1);

   array[n] = x;
   n++;
}
```

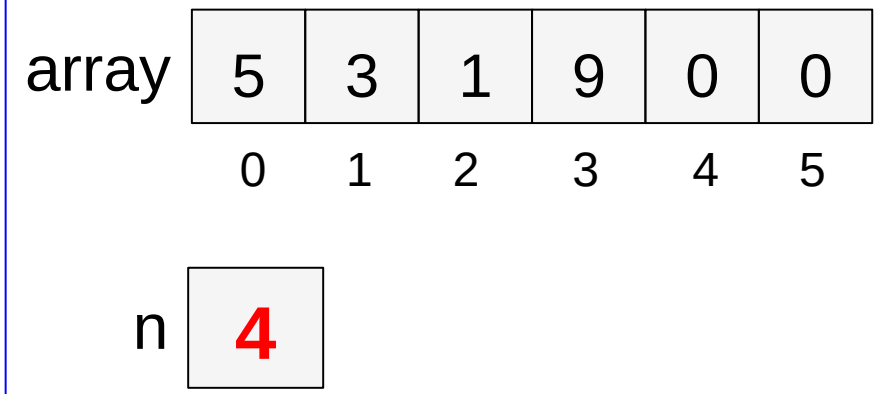
x 9



Algoritmo em C

```
// Exemplo: inserirFim(9)

void inserirFim(int x) {

   if (n >= MAXTAM)
      exit(1);

   array[n] = x;
   n++;
}
```

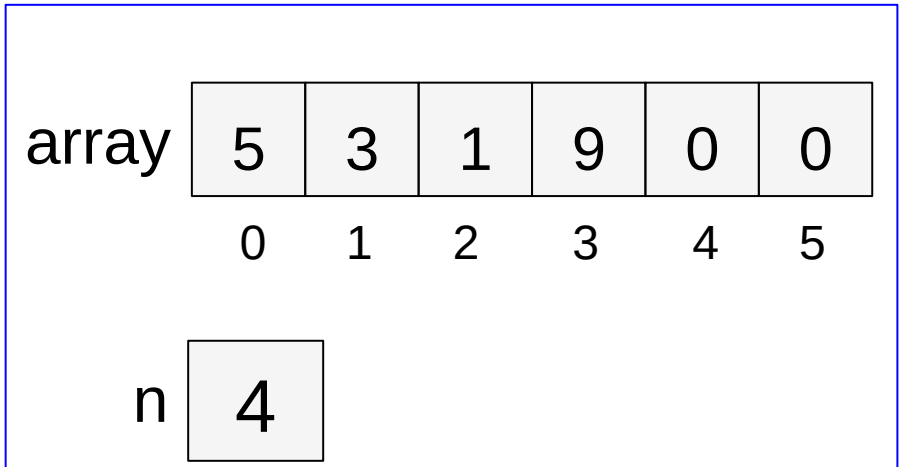
x | 9



Algoritmo em C

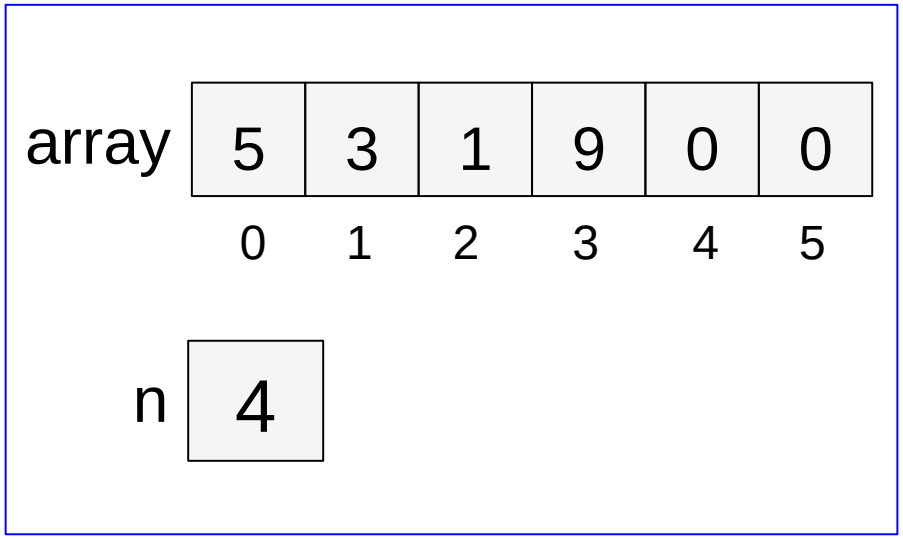
```
// Exemplo: inserirFim(9)

void inserirFim(int x) {

   if (n >= MAXTAM)
      exit(1);

   array[n] = x;
   n++;
}
```

x 9



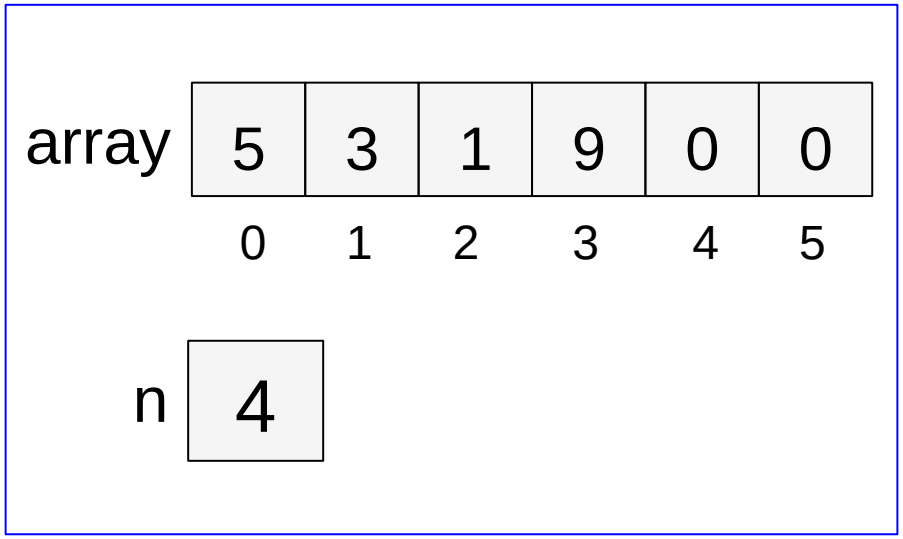
Algoritmo em C

```
int array[MAXTAM];
int n;


void start(){
    n = 0;
}

void inserirInicio(int x)   {  ...  }
void inserirFim(int x)   {  ...  }
void inserir(int x, int pos)   {  ...  }
int removerInicio()   {  ...  }
int removerFim()   {  ...  }
int remover(int pos)   {  ...  }
void mostrar () {  ...  }
```



Algoritmo em C

```
// Exemplo: inserir(4,2)                    pos 2   x 4
void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

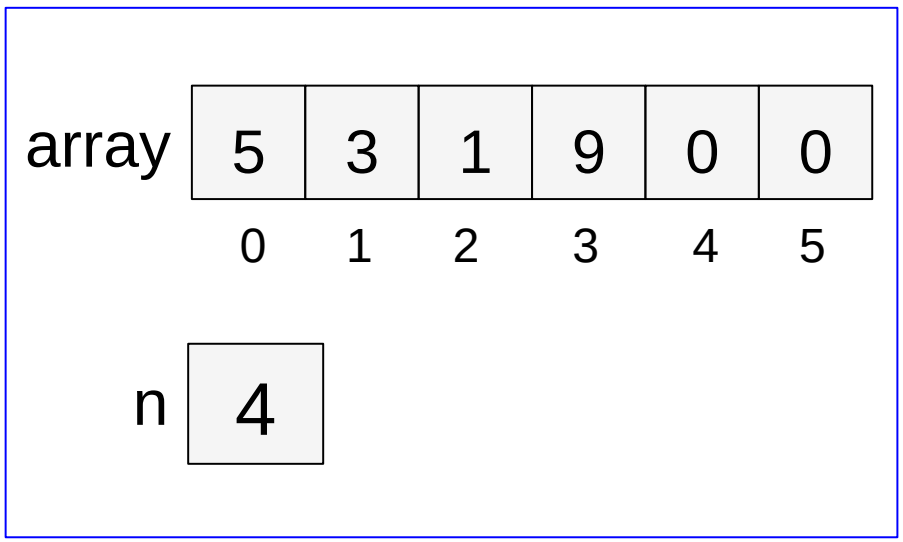


Algoritmo em C

```
// Exemplo: inserir(4,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
      array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 4

false: 4 >= 6 || 2 < 0 || 0 > 4

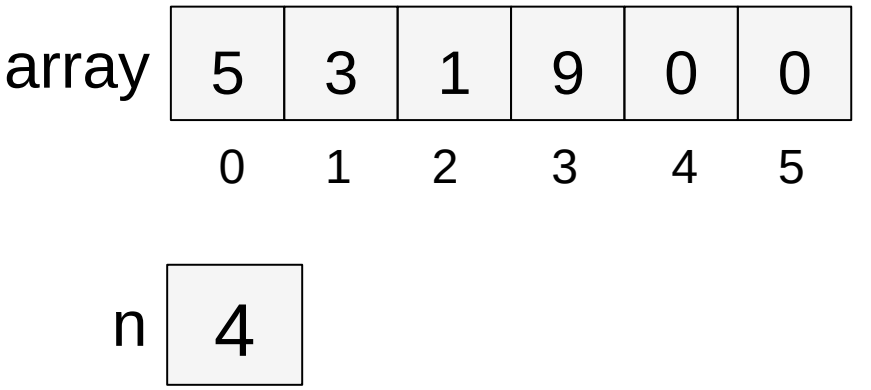


Algoritmo em C

```
// Exemplo: inserir(4,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 4

i 4



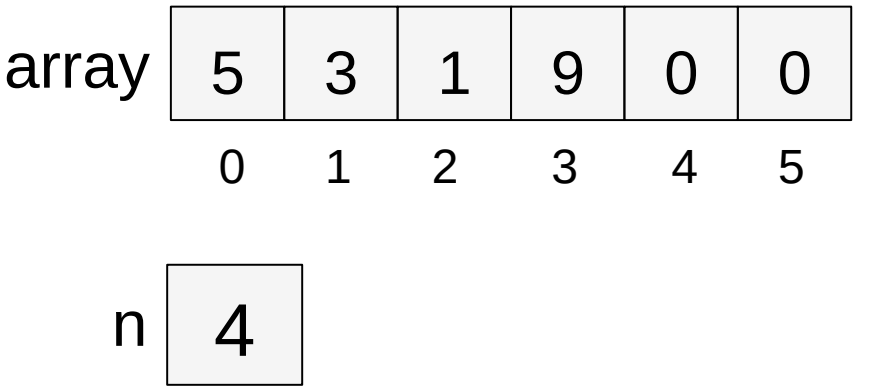
Algoritmo em C

```
// Exemplo: inserir(4,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 4

i 4

true: 4 > 2



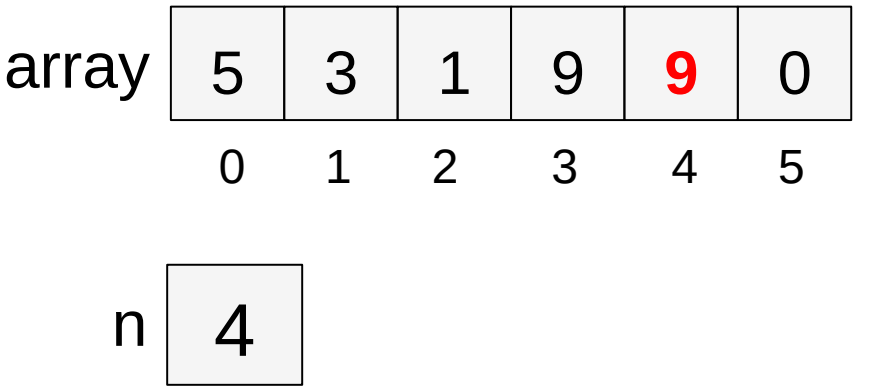
Algoritmo em C

```
// Exemplo: inserir(4,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 4

i 4

array[4] ← array[3]

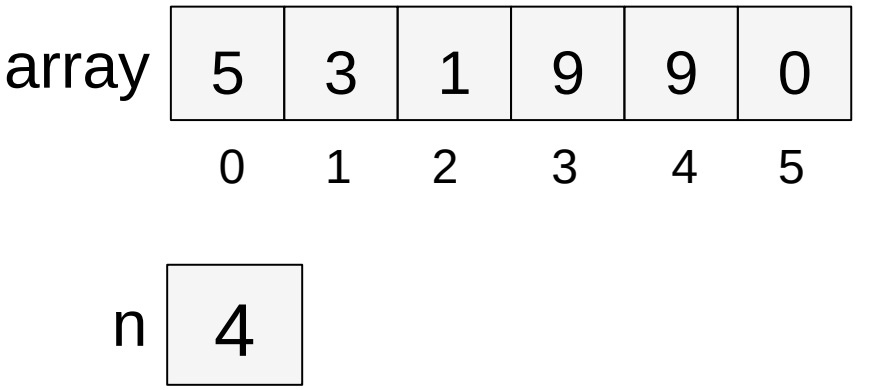


Algoritmo em C

```
// Exemplo: inserir(4,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 4

i 3



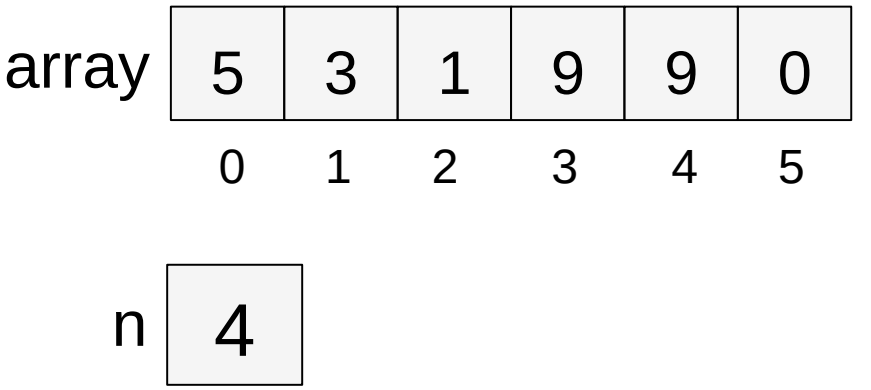
Algoritmo em C

```
// Exemplo: inserir(4,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 4

i 3

true: 3 > 2

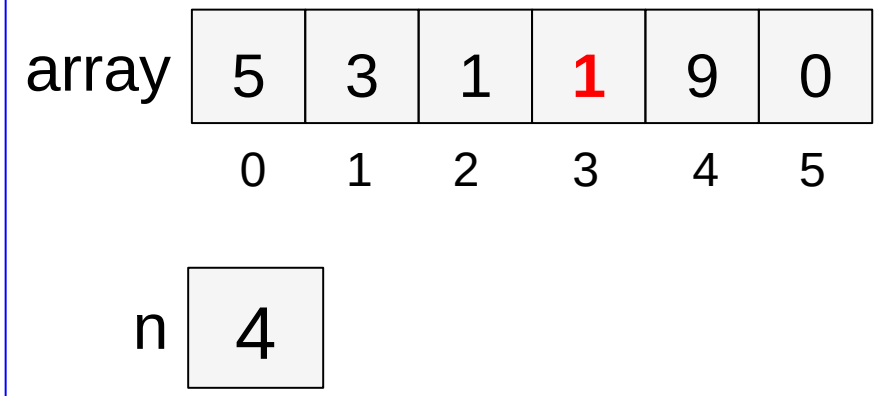


Algoritmo em C

```
// Exemplo: inserir(4,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 4 i 3

array[3] ← array[2]

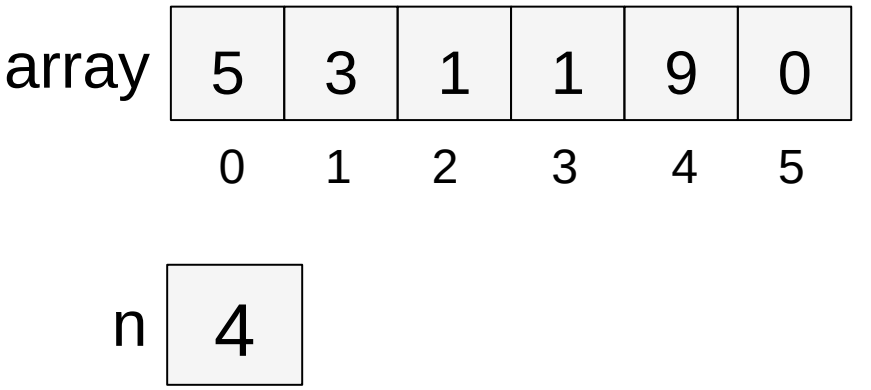


Algoritmo em C

```
// Exemplo: inserir(4,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 4

i 2



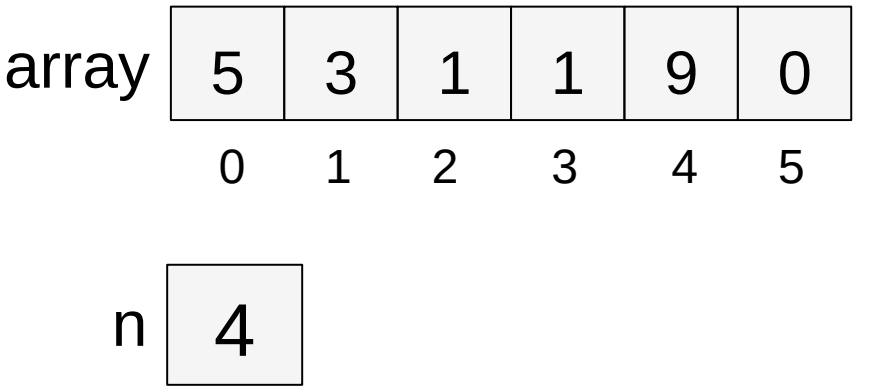
Algoritmo em C

```
// Exemplo: inserir(4,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 4

i 2

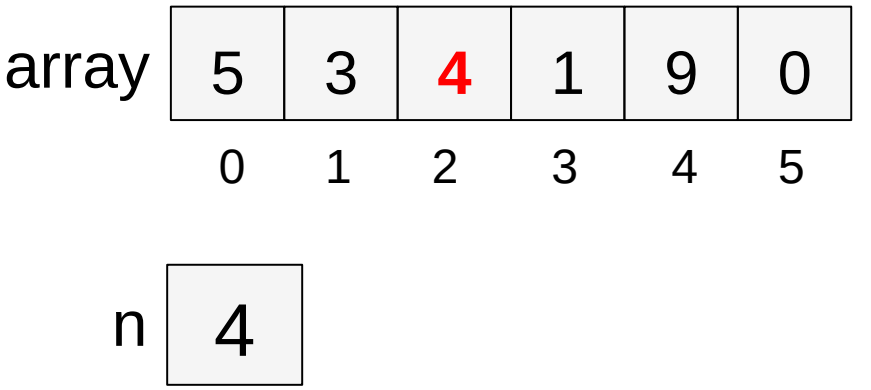
false: 2 > 2



Algoritmo em C

```
// Exemplo: inserir(4,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

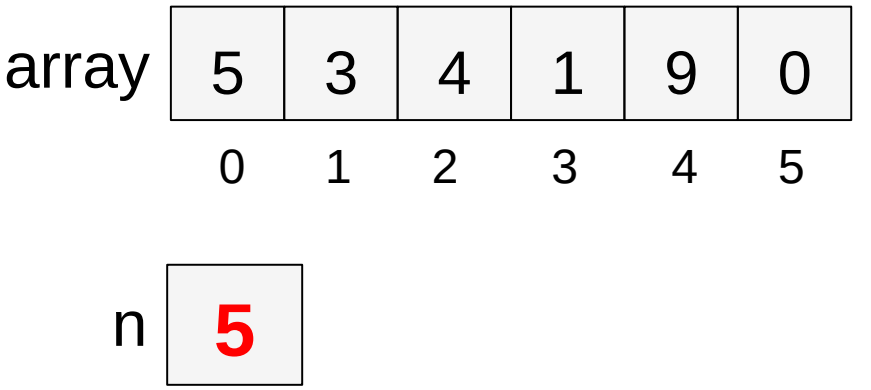
pos 2 x 4



Algoritmo em C

```
// Exemplo: inserir(4,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

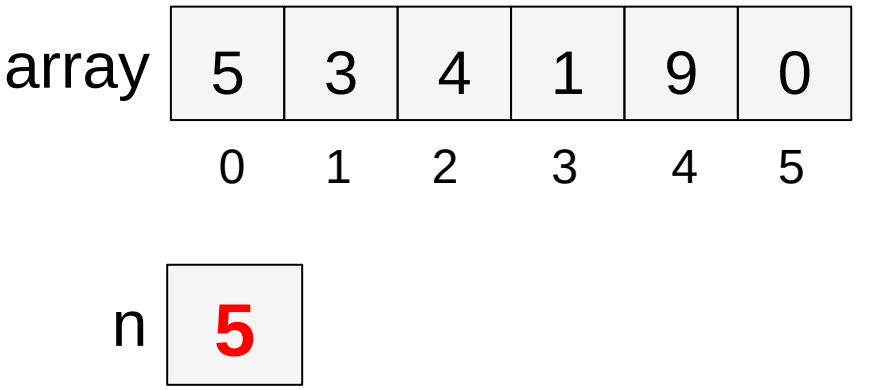
pos 2 x 4



Algoritmo em C

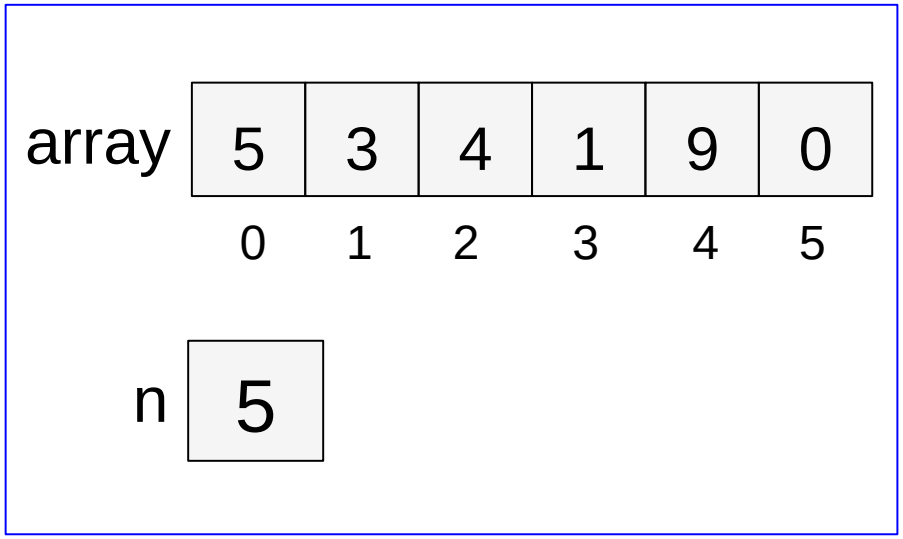
```
// Exemplo: inserir(4,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 4



Algoritmo em C

```
// Exemplo: inserir(7,2)                    pos 2    x 7
void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

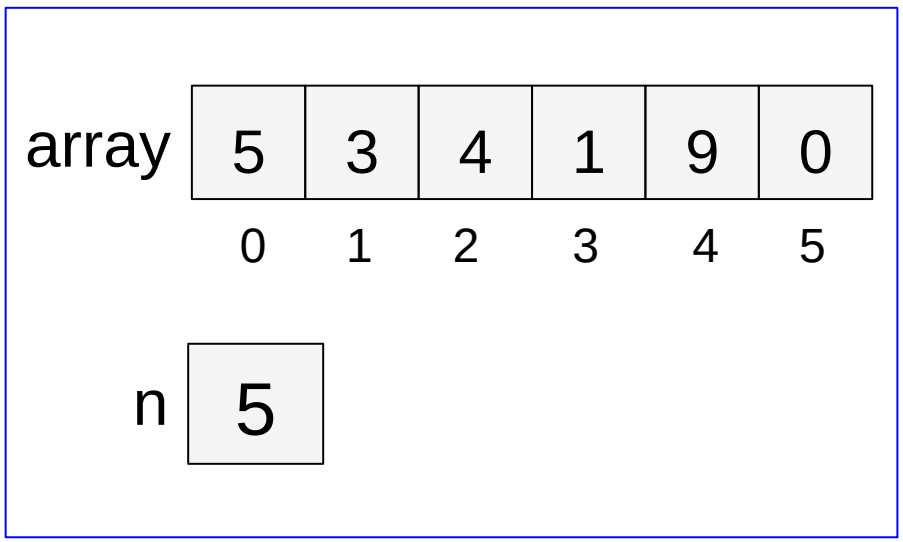


Algoritmo em C

```
// Exemplo: inserir(7,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
      array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 7

false: 5 >= 6 || 2 < 0 || 0 > 4

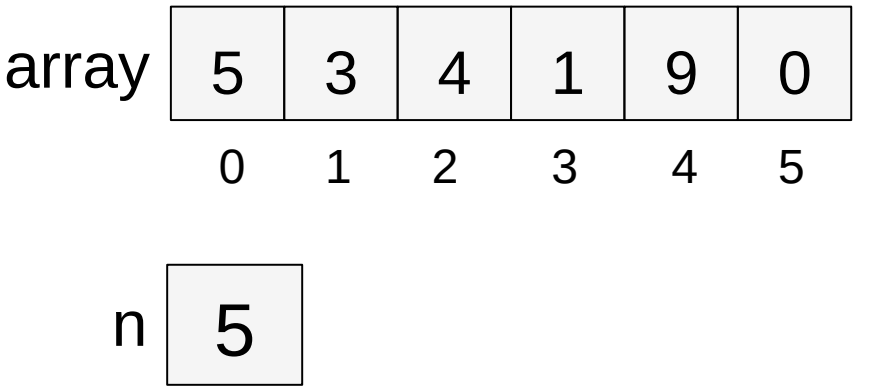


Algoritmo em C

```
// Exemplo: inserir(7,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 7

i 5



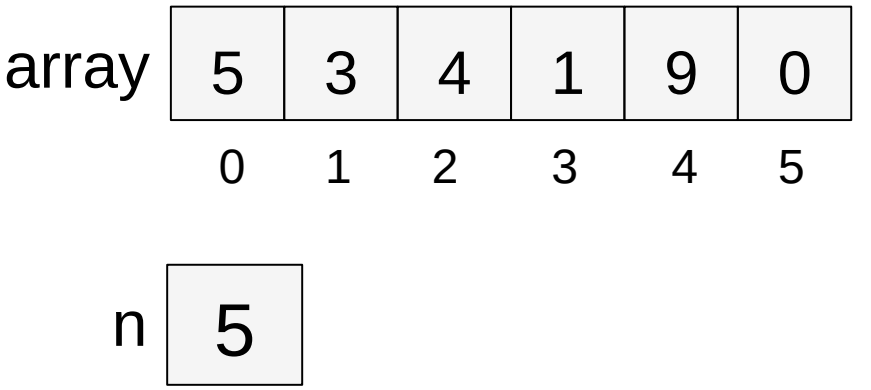
Algoritmo em C

```
// Exemplo: inserir(7,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 7

i 5

true: 5 > 2

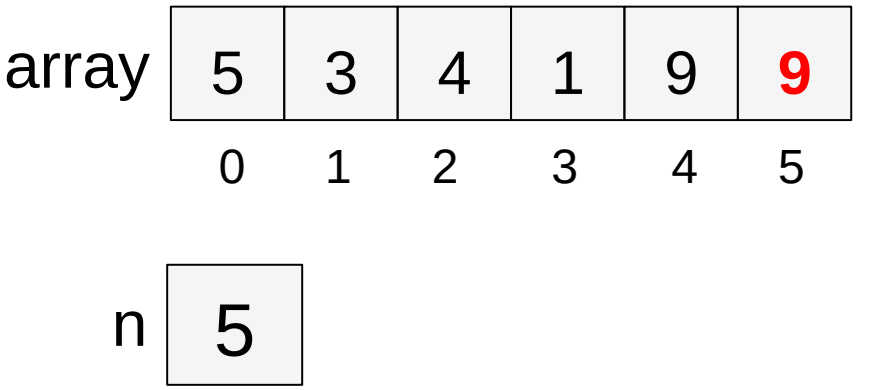


Algoritmo em C

```
// Exemplo: inserir(7,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 7 i 5

array[5] ← array[4]

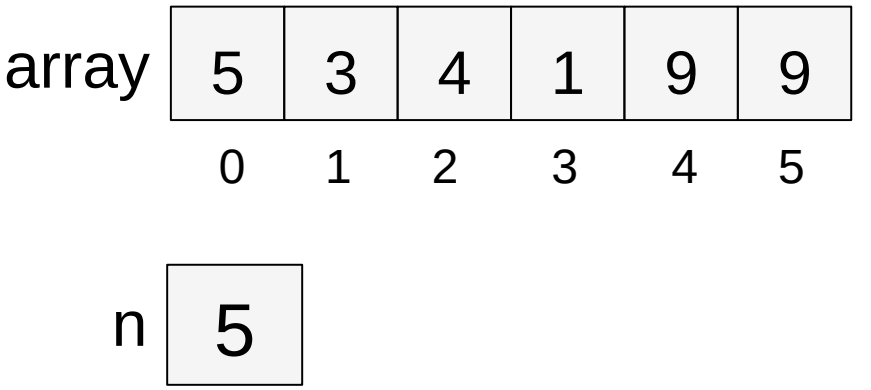


Algoritmo em C

```
// Exemplo: inserir(7,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 7

i 4



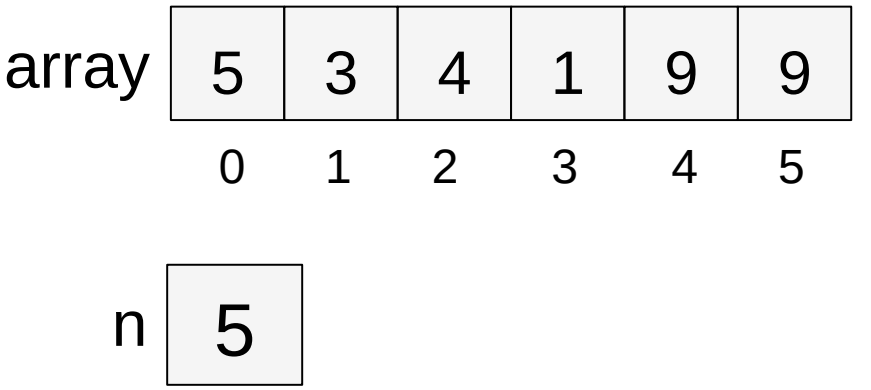
Algoritmo em C

```
// Exemplo: inserir(7,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 7

i 4

true: 4 > 2

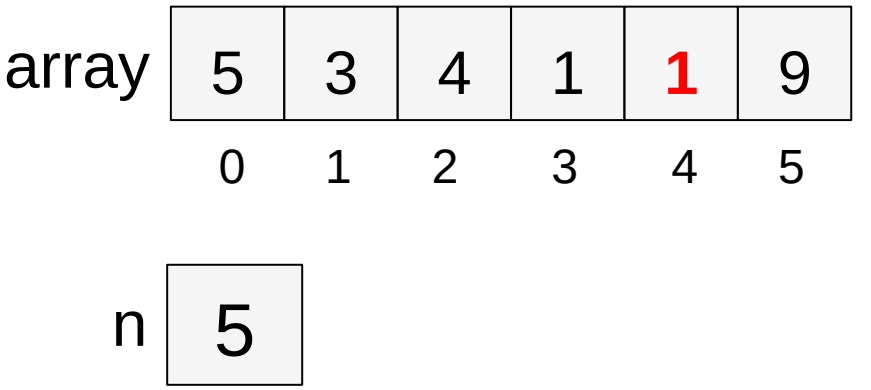


Algoritmo em C

```
// Exemplo: inserir(7,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 7 i 4

array[4] ← array[3]

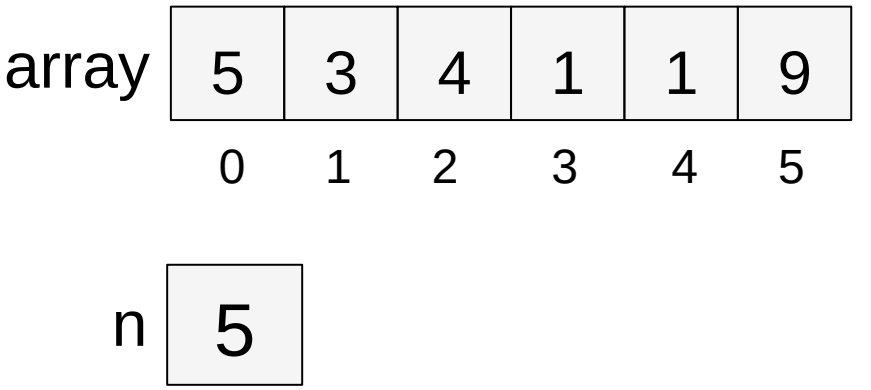


Algoritmo em C

```
// Exemplo: inserir(7,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 7

i 3



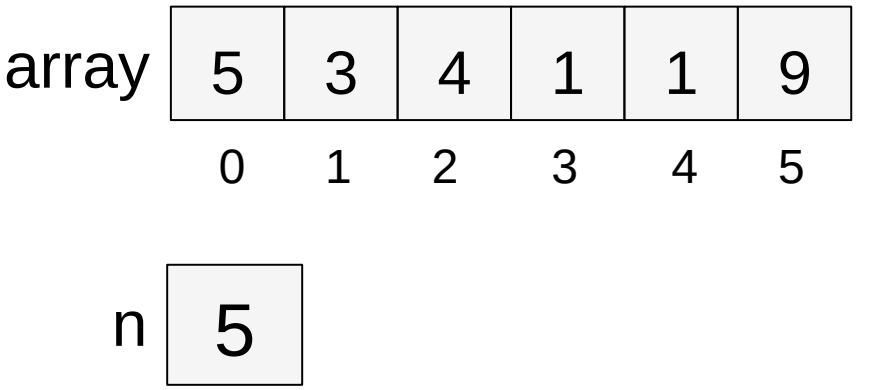
Algoritmo em C

```
// Exemplo: inserir(7,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 7

i 3

true: 3 > 2

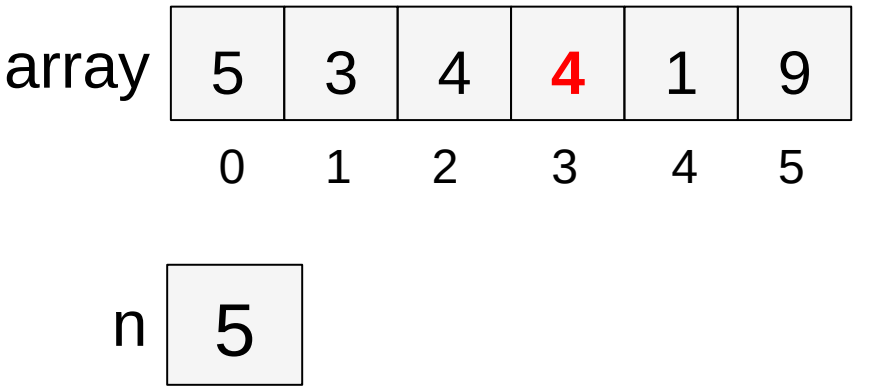


Algoritmo em C

```
// Exemplo: inserir(7,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 7 i 3

array[3] ← array[2]

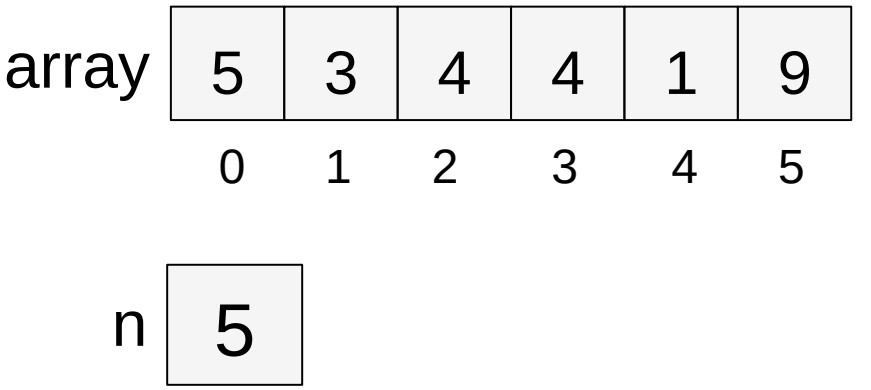


Algoritmo em C

```
// Exemplo: inserir(7,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 7

i 2



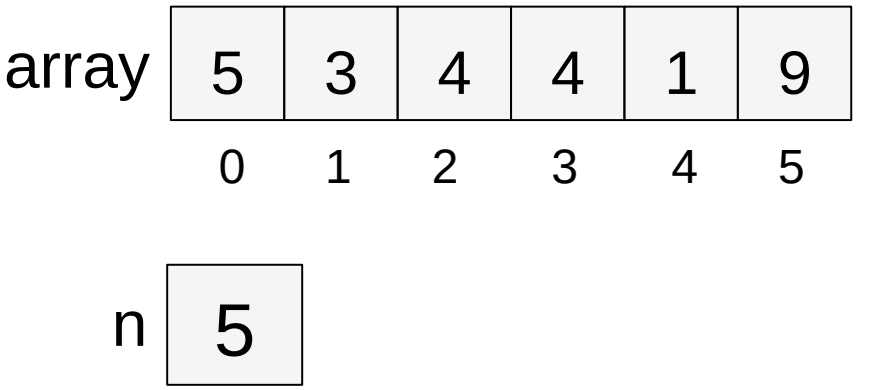
Algoritmo em C

```
// Exemplo: inserir(7,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 7

i 2

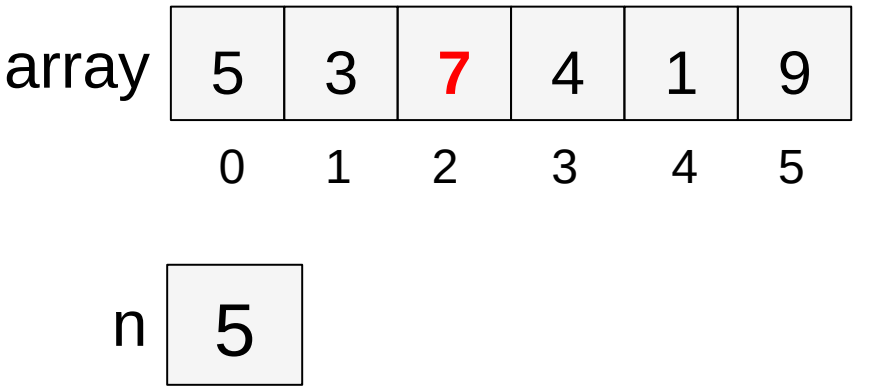
false: 2 > 2



Algoritmo em C

```
// Exemplo: inserir(7,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
      exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
      array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

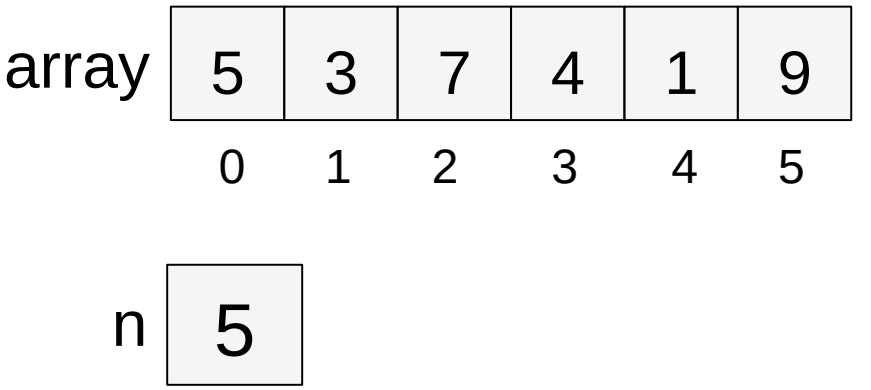
pos 2 x 7



Algoritmo em C

```
// Exemplo: inserir(7,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

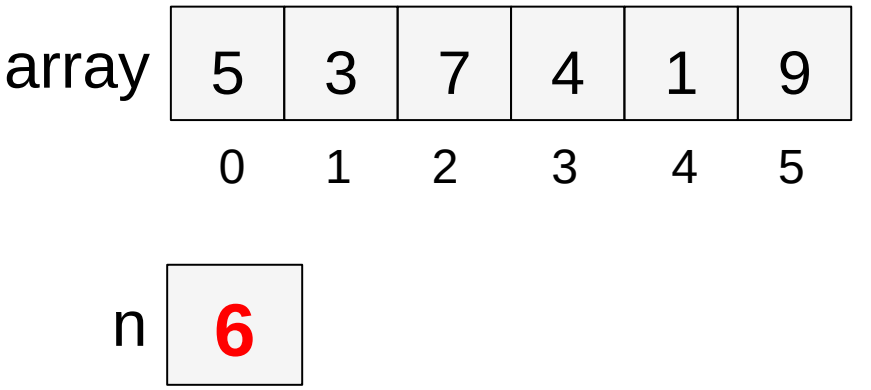
pos 2 x 7



Algoritmo em C

```
// Exemplo: inserir(7,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

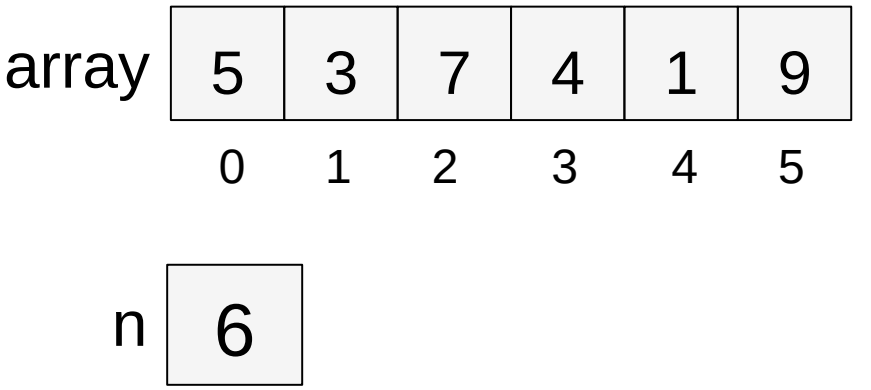
pos 2 x 7



Algoritmo em C

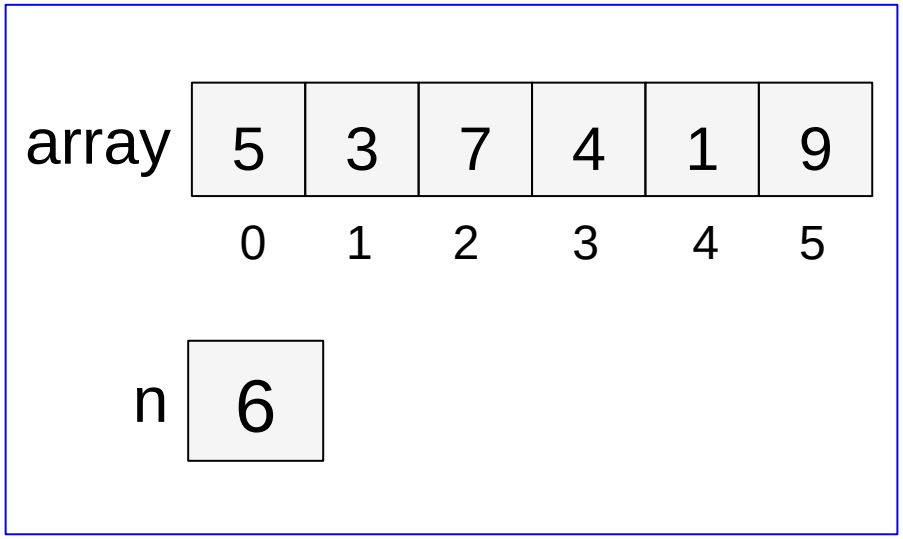
```
// Exemplo: inserir(7,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

pos 2 x 7



Algoritmo em C

```
// Exemplo: inserir(2,2)                    pos 2   x 2
void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

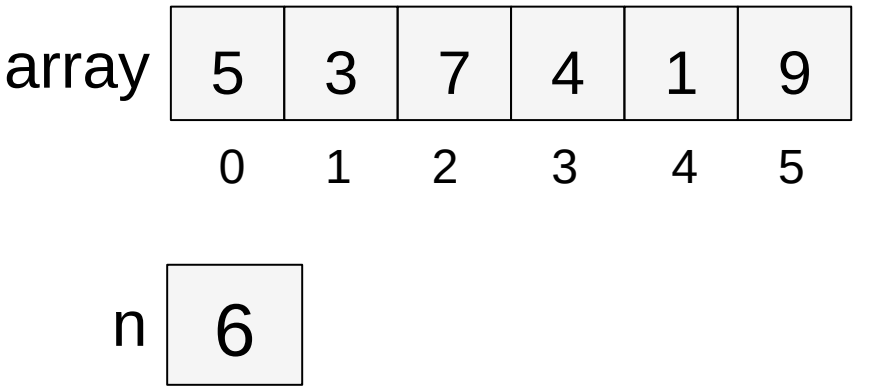


Algoritmo em C

```
// Exemplo: inserir(2,2)

void inserir(int x, int pos) {

 if (n >= MAXTAM || pos < 0 || pos > n)
     exit(1);

   //levar elementos para o fim do array
   for (int i = n; i > pos; i--){
     array[i] = array[i-1];
   }

   array[pos] = x;
   n++;
}
```

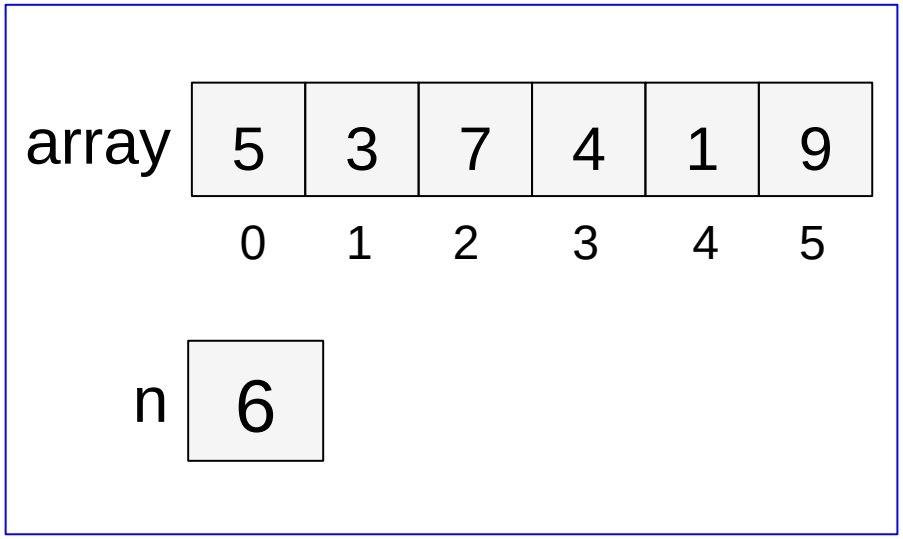
pos 2 x 2

true: 6 >= 6 || ... || ...



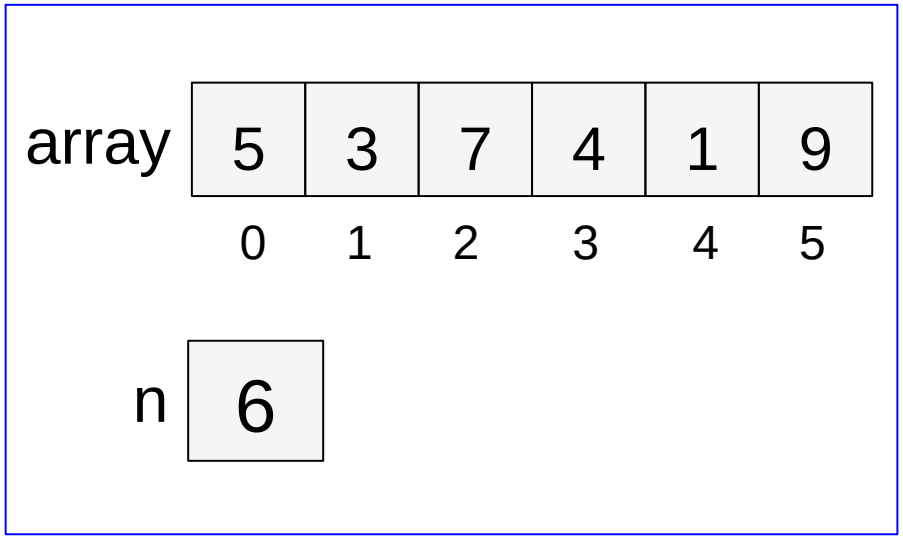
Algoritmo em C

```
int array[MAXTAM];
int n;


void start(){
    n = 0;
}

void inserirInicio(int x)   {  ...  }
void inserirFim(int x)   {  ...  }
void inserir(int x, int pos)   {  ...  }
int removerInicio()   {  ...  }
int removerFim()   {  ...  }
int remover(int pos)   {  ...  }
void mostrar () {  ...  }
```



Algoritmo em C

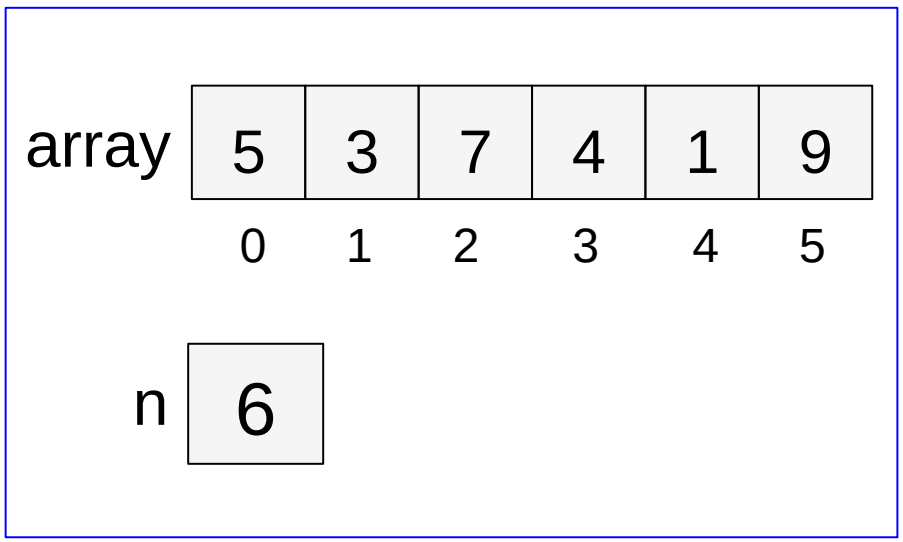
```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```



Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

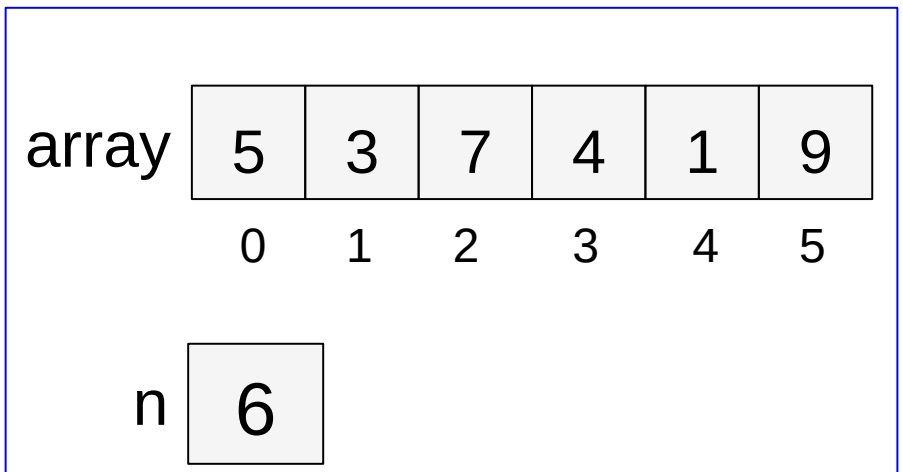
false: 6 == 0



Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

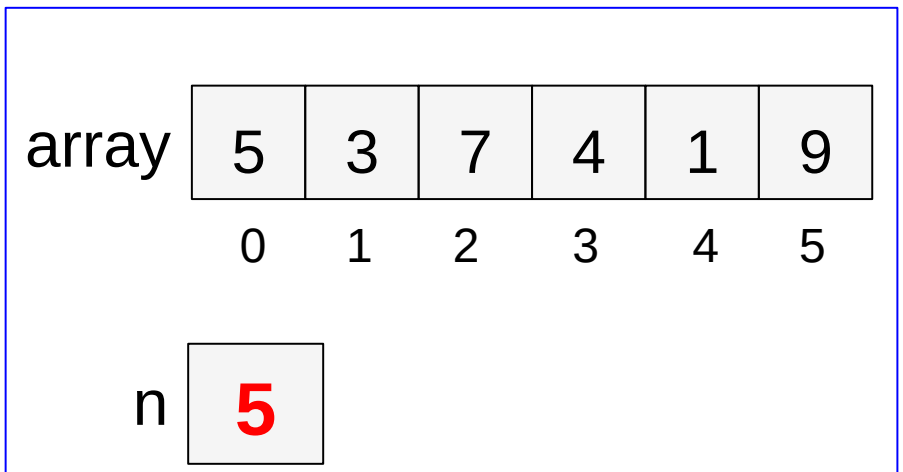
resp **5**



Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp: 5

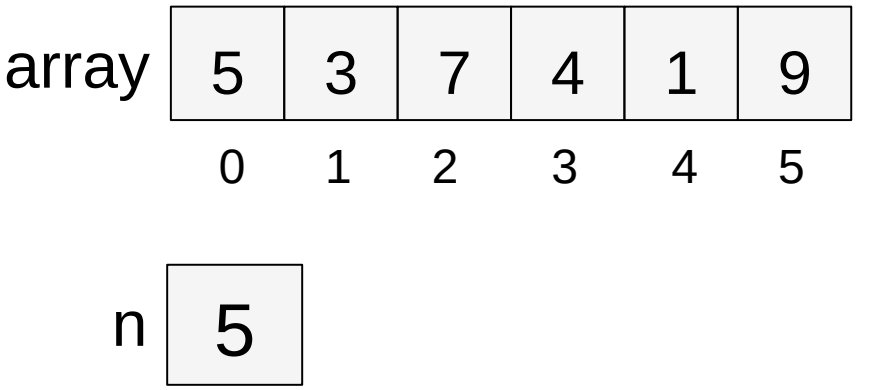


Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 5

i 0



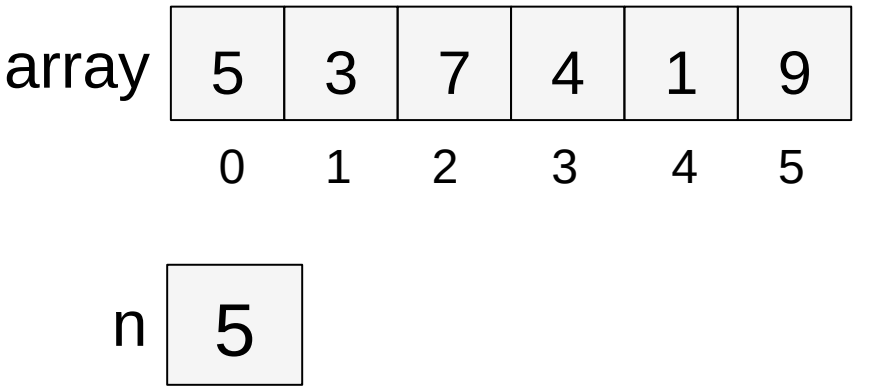
Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 5

i 0

true: 0 < 5



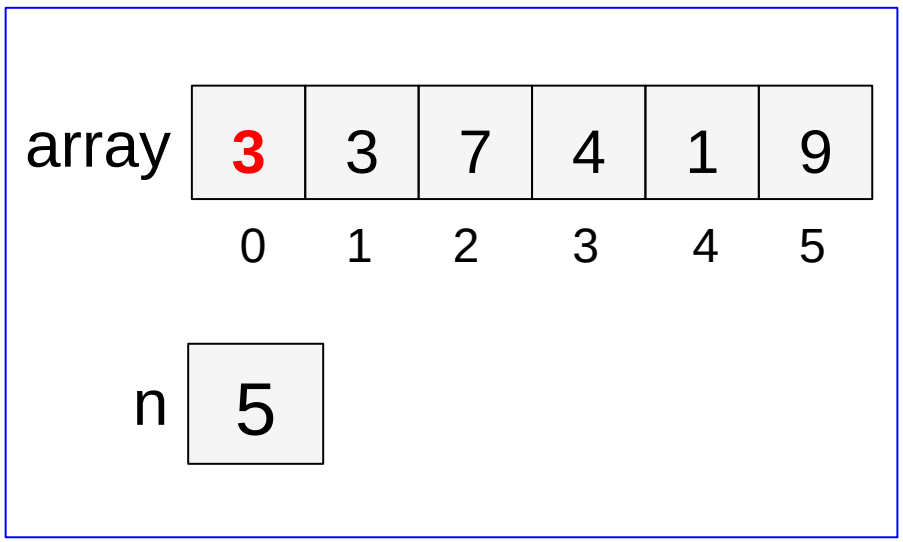
Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 5

i 0

array[0] ← array[1]

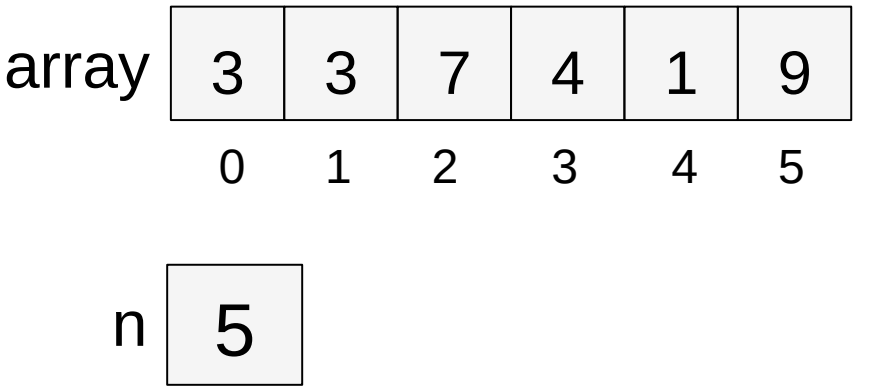


Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp: 5

i: 1



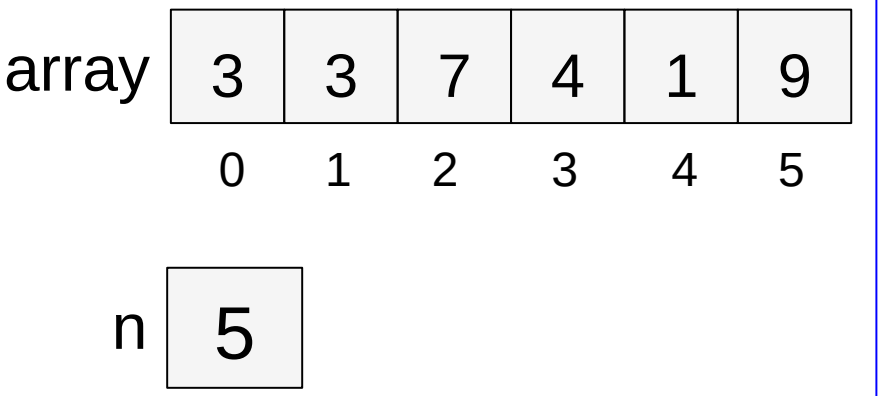
Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 5

i 1

true: 1 < 5



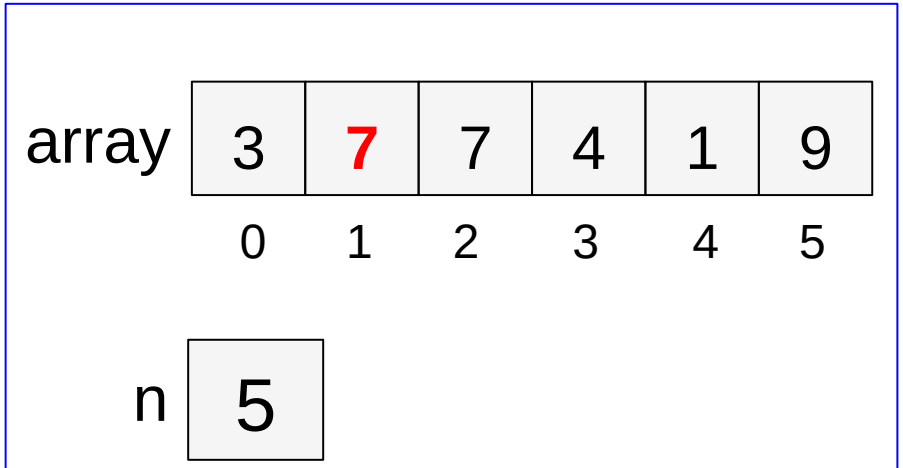
Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 5

i 1

array[1] ← array[2]

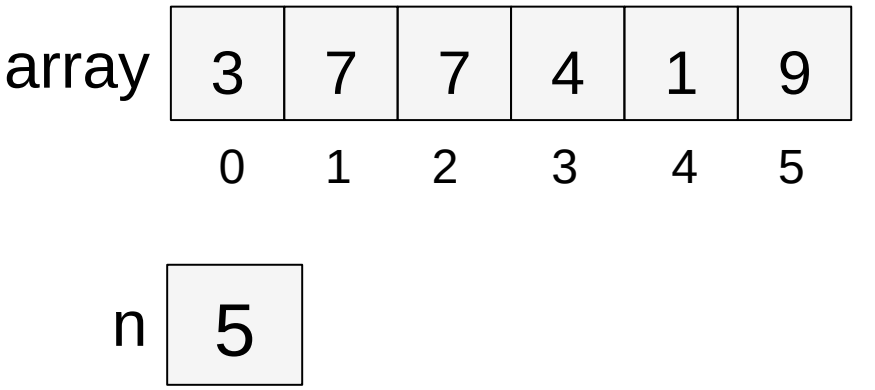


Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp: 5

i: 2



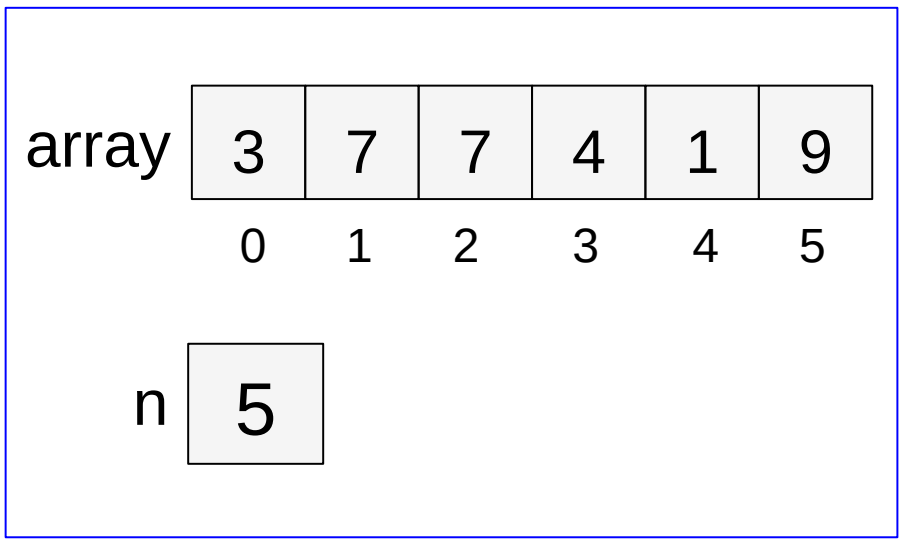
Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 5

i 2

true: 2 < 5



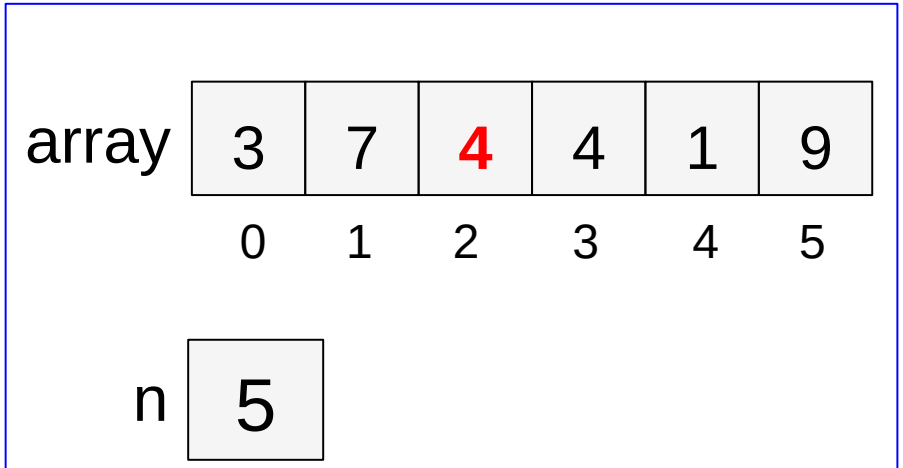
Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 5

i 2

array[2] ← array[3]

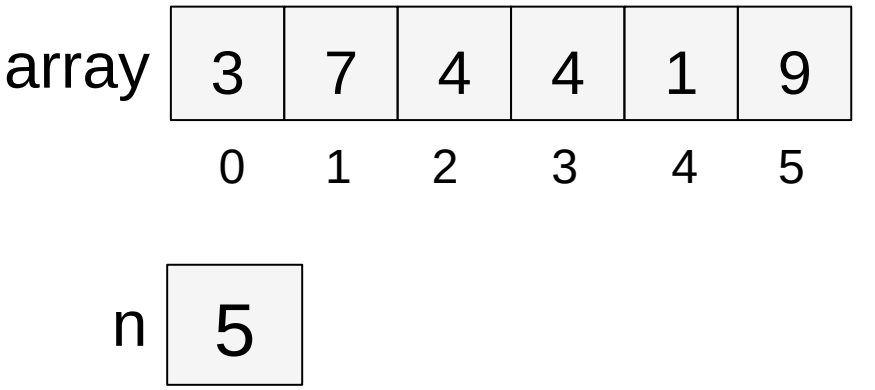


Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp: 5

i: 3



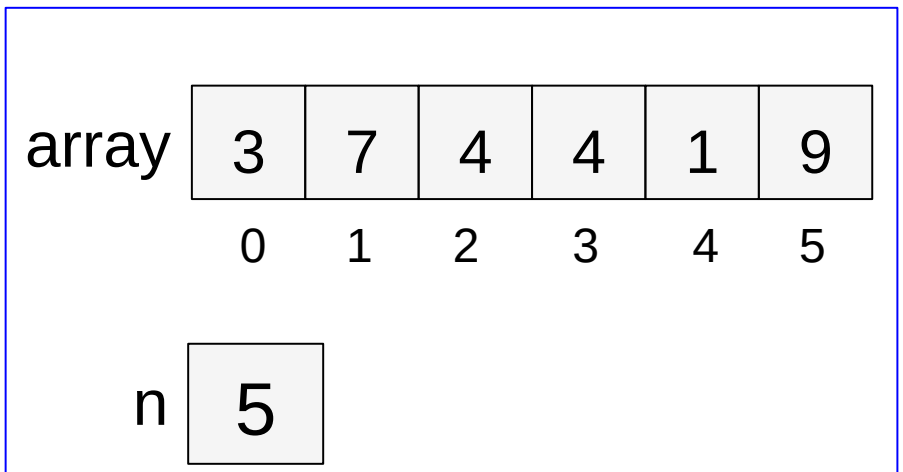
Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 5

i 3

true: 3 < 5



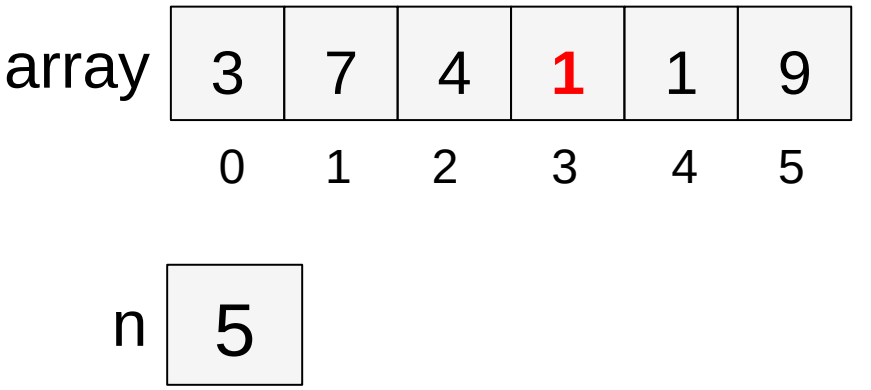
Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 5

i 3

array[3] ← array[4]

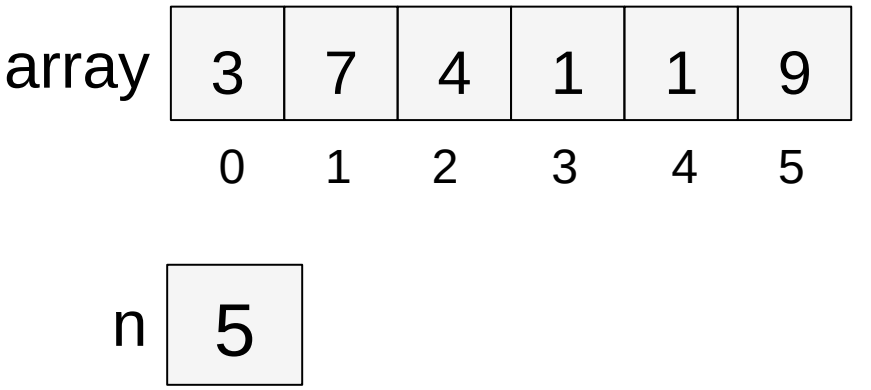


Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp: 5

i: 4



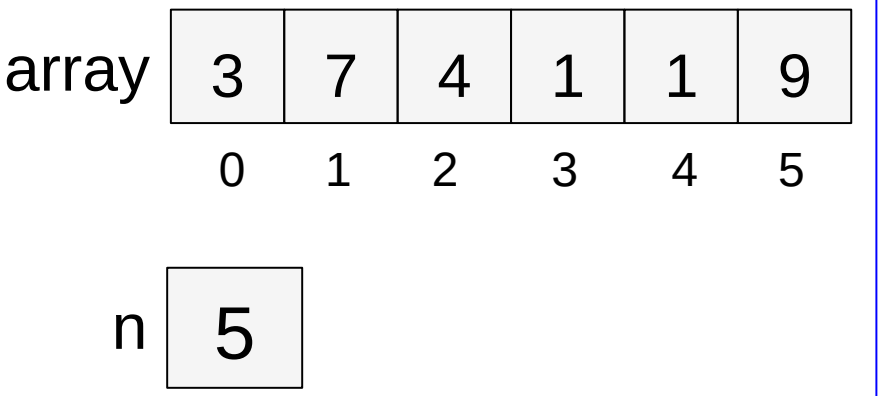
Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 5

i 4

true: 4 < 5



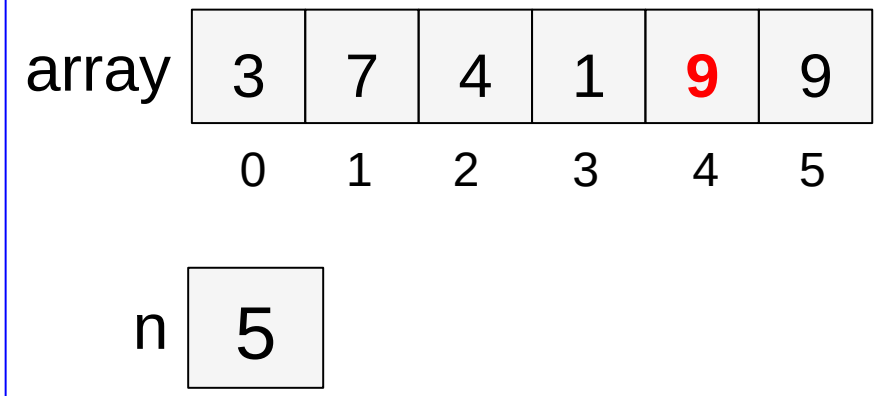
Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 5

i 4

array[4] ← array[5]

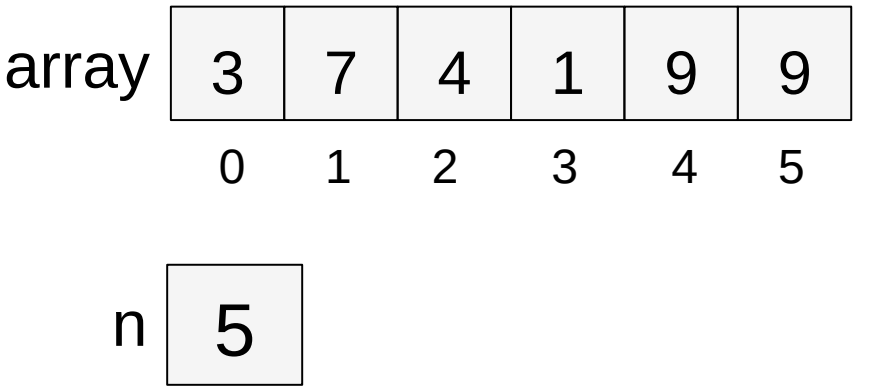


Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp: 5

i: 5



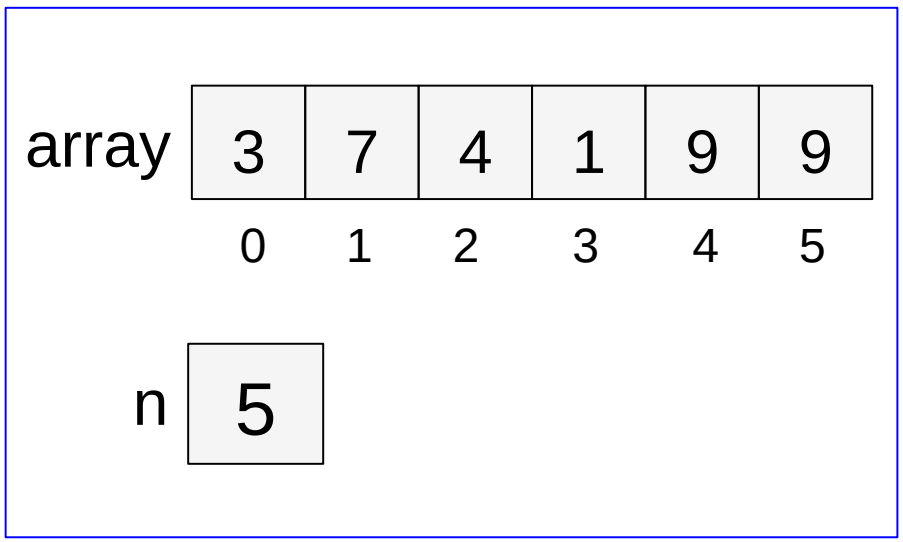
Algoritmo em C

```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 5

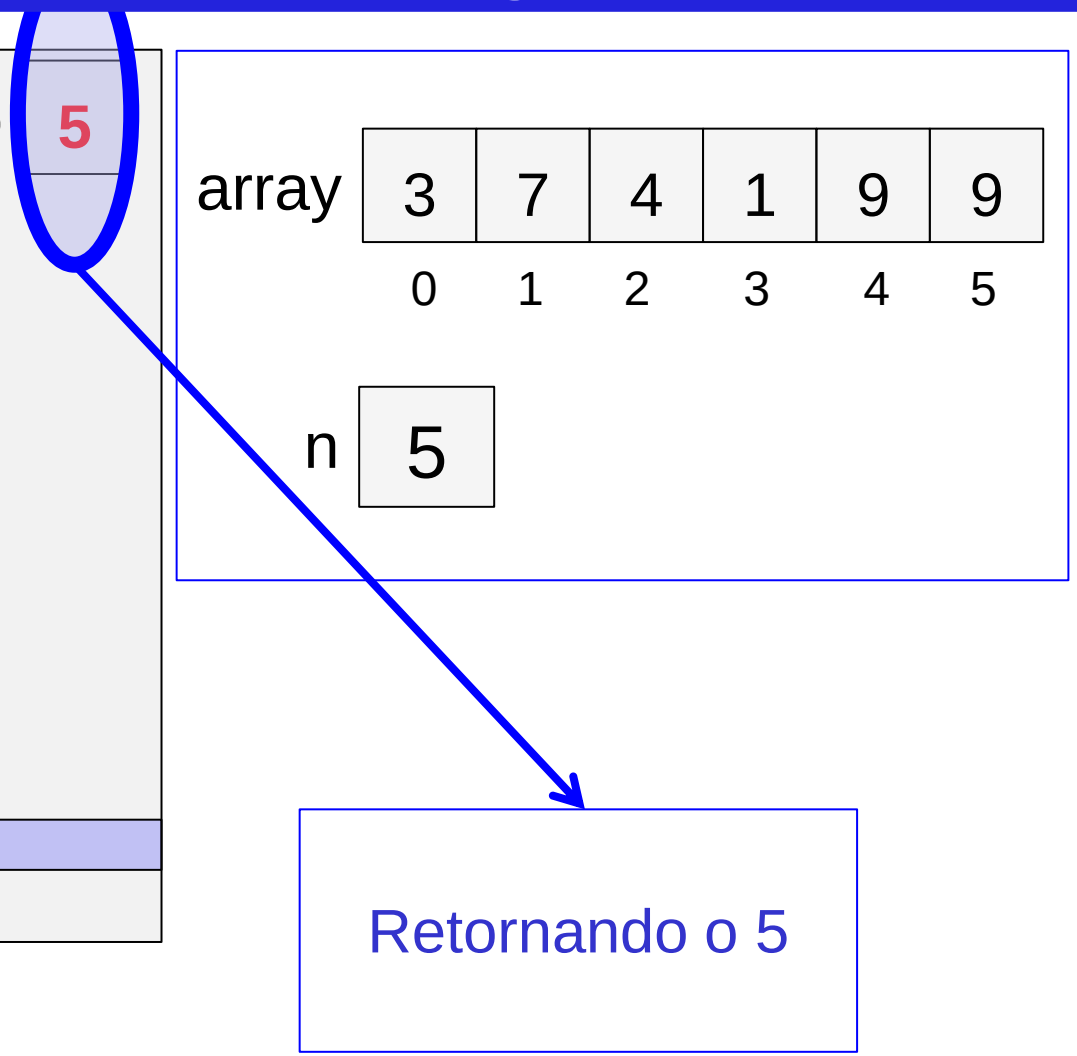
i 5

false: 5 < 5



Algoritmo em C

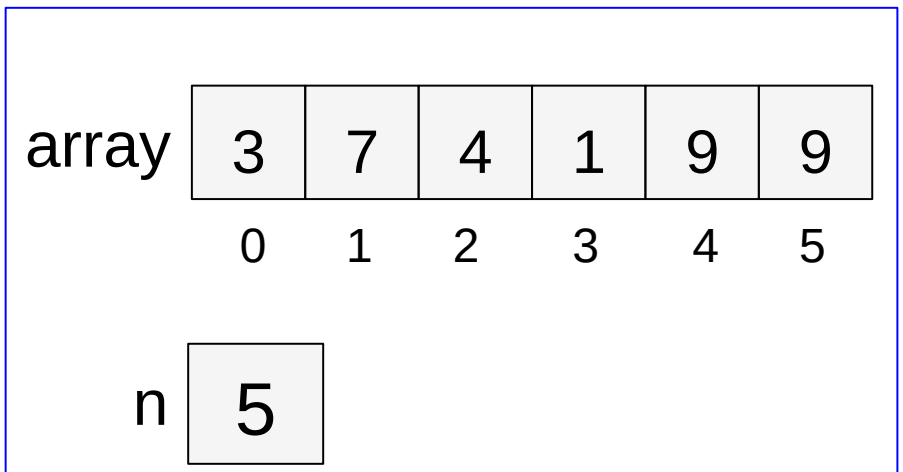
```c
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```



Algoritmo em C

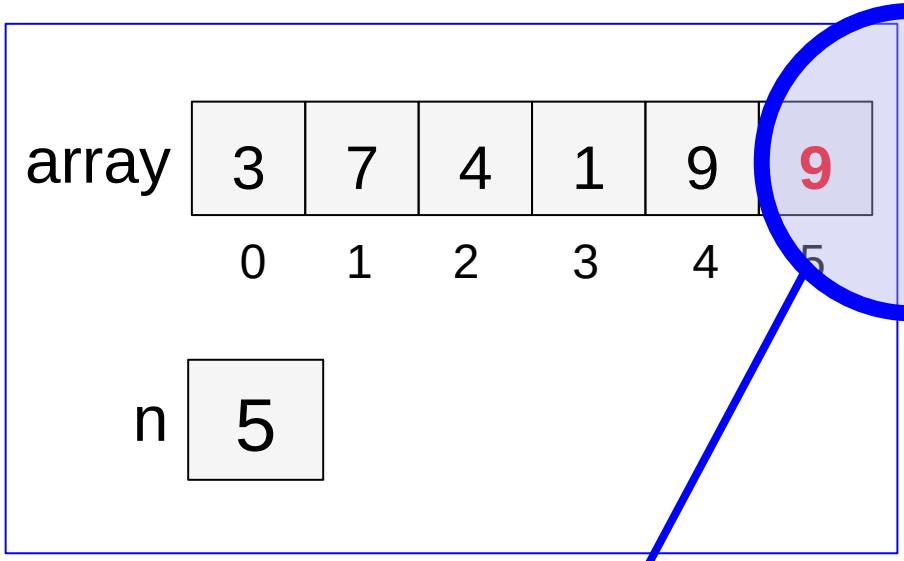
```
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp: 5



Algoritmo em C

```c
// Exemplo: removerInicio()

int removerInicio() {
   if (n == 0)
      exit(1);

   int resp = array[0];
   n--;

   for (int i = 0; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```



Este nove foi ou não removido?

Exercício

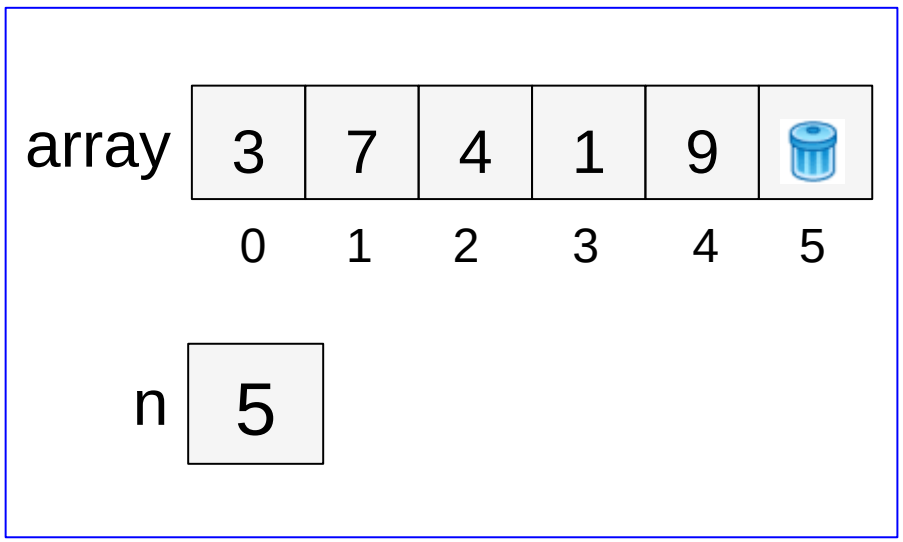
- Diferencie remoção lógica e física
- Diferencie formatação lógica (rápida) e física
- No seu SO, o que acontece quando enviamos um arquivo para a lixeira? E quando o excluímos definitivamente?

Algoritmo em C

```
int array[MAXTAM];
int n;


void start(){
    n = 0;
}

void inserirInicio(int x)   {  ...  }
void inserirFim(int x)   {  ...  }
void inserir(int x, int pos)   {  ...  }
int removerInicio()   {  ...  }
int removerFim()   {  ...  }
int remover(int pos)   {  ...  }
void mostrar () {  ...  }
```

Algoritmo em C

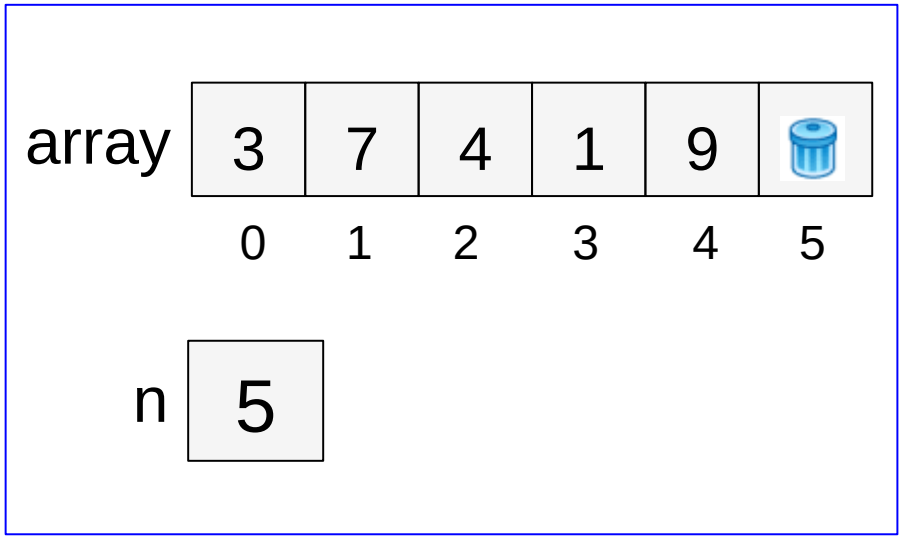
```
// Exemplo: removerFim()

int removerFim() {
   if (n == 0)
      exit(1);

   return array[--n];
}
```



Algoritmo em C

```
// Exemplo: removerFim()

int removerFim() {
   if (n == 0)
      exit(1);

   return array[--n];
}
```

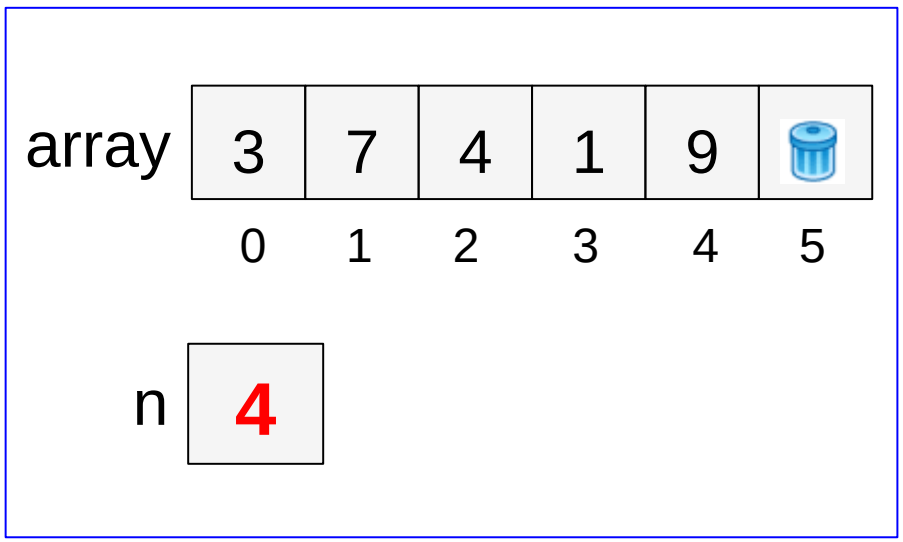
false: 5 == 0



Algoritmo em C

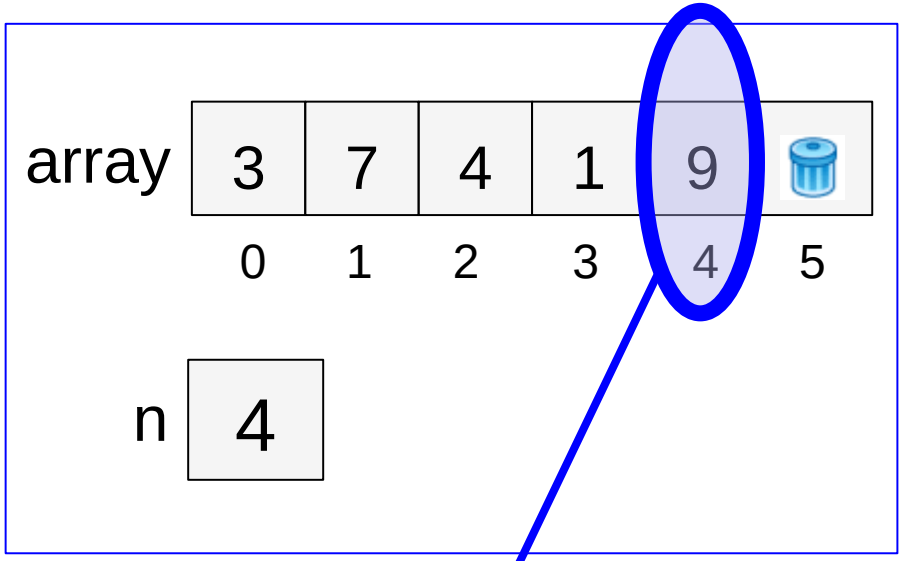
```
// Exemplo: removerFim()

int removerFim() {
   if (n == 0)
      exit(1);

   return array[--n];
}
```

Primeiro o --,
depois o retorno



Algoritmo em C

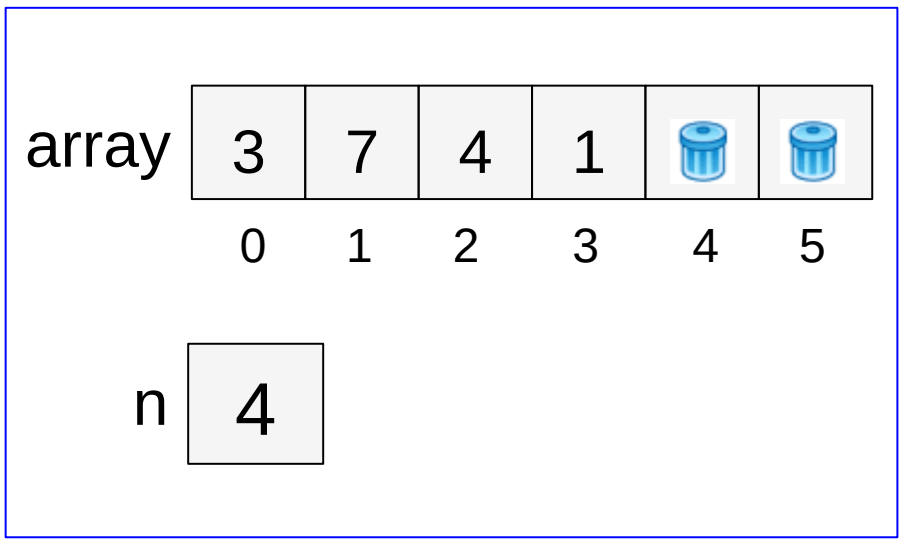
```
// Exemplo: removerFim()

int removerFim() {
   if (n == 0)
      exit(1);

   return array[--n];
}
```



Retornando o 9

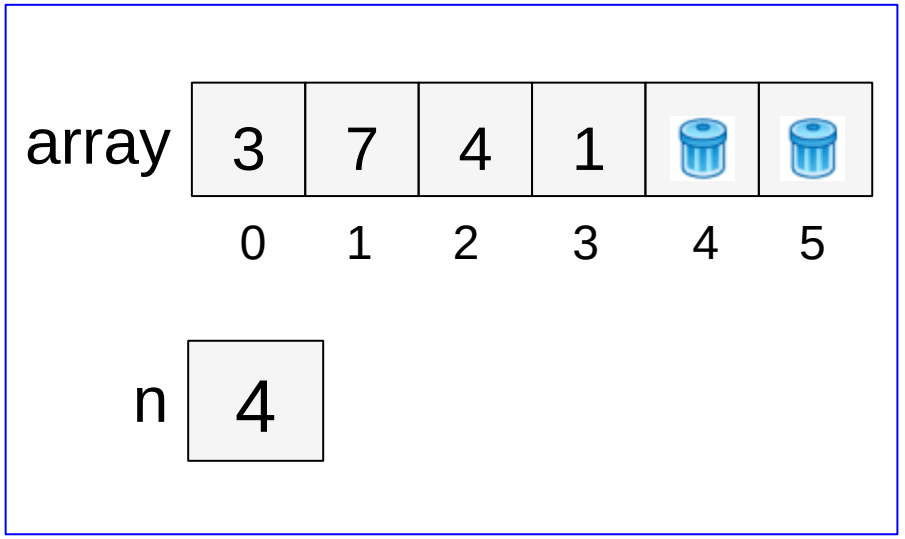
Algoritmo em C

```
// Exemplo: removerFim()

int removerFim() {
   if (n == 0)
      exit(1);

   return array[--n];
}
```



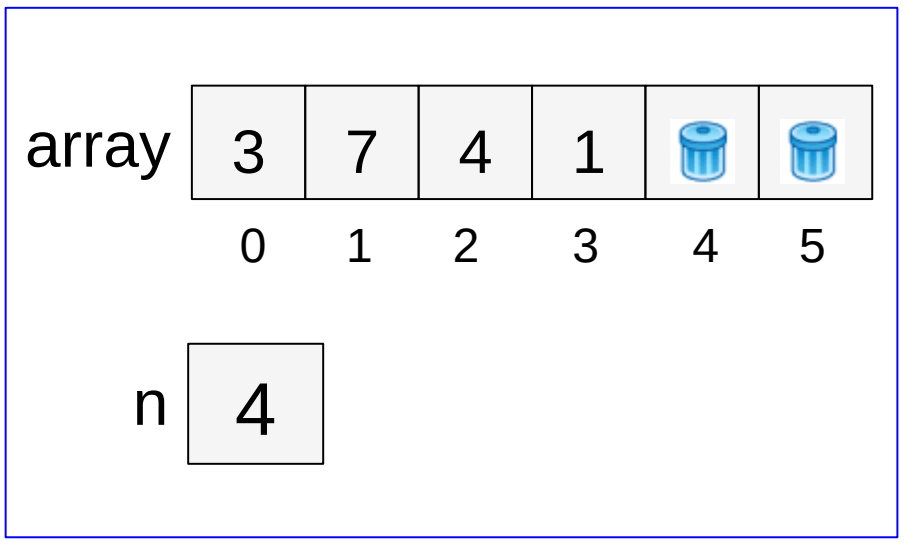
Algoritmo em C

```
int array[MAXTAM];
int n;


void start(){
    n = 0;
}

void inserirInicio(int x)   {  ...  }
void inserirFim(int x)   {  ...  }
void inserir(int x, int pos)   {  ...  }
int removerInicio()   {  ...  }
int removerFim()   {  ...  }
int remover(int pos)   {  ...  }
void mostrar () {  ...  }
```



Algoritmo em C

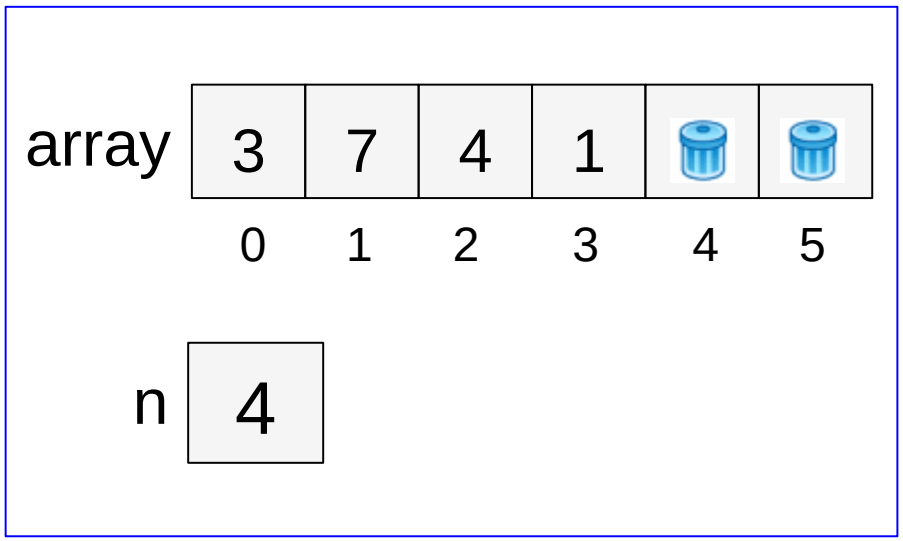
```
// Exemplo: remover(2)

int remover(int pos) {

   if (n == 0 || pos < 0 || pos >= n)
      exit(1);

   int resp = array[pos];
   n--;

   for (int i = pos; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```



Algoritmo em C

```
// Exemplo: remover(2)

int remover(int pos) {

   if (n == 0 || pos < 0 || pos >= n)
      exit(1);

   int resp = array[pos];
   n--;

   for (int i = pos; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

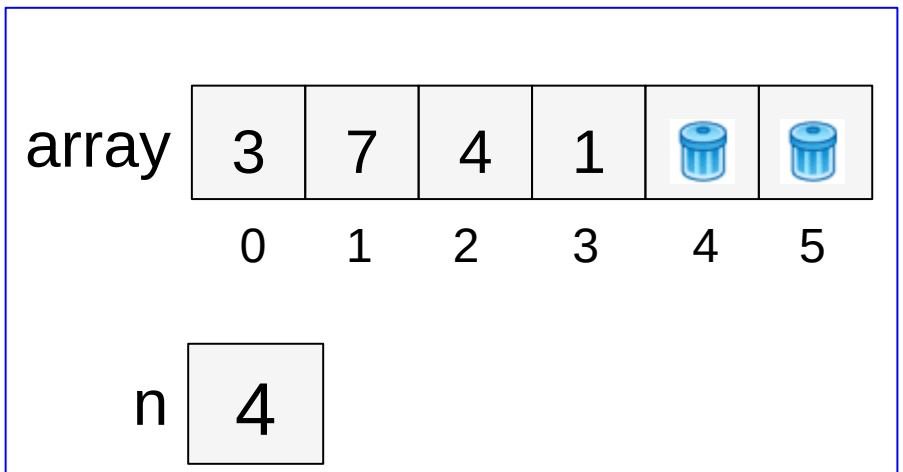
false: 5 == 0 || 2 < 0 || 2 >= 4



Algoritmo em C

```
// Exemplo: remover(2)

int remover(int pos) {

   if (n == 0 || pos < 0 || pos >= n)
      exit(1);

   int resp = array[pos];
   n--;

   for (int i = pos; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

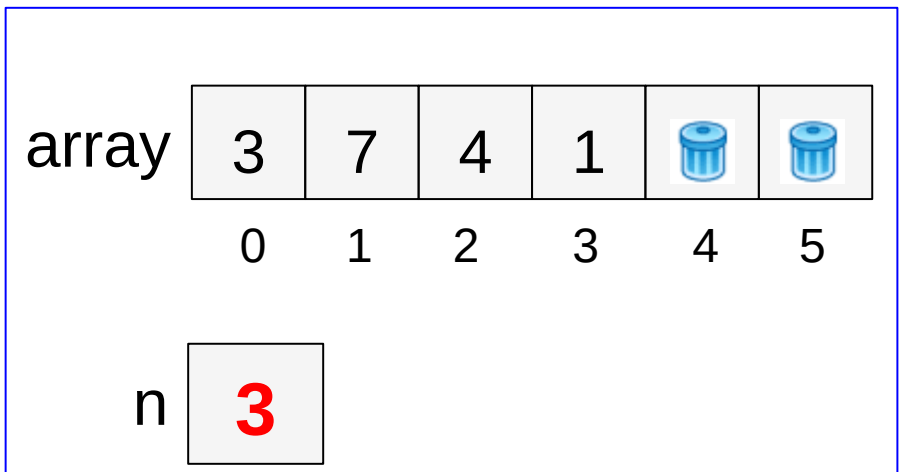
resp **4**



Algoritmo em C

```
// Exemplo: remover(2)

int remover(int pos) {

   if (n == 0 || pos < 0 || pos >= n)
      exit(1);

   int resp = array[pos];
   n--;

   for (int i = pos; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp: 4

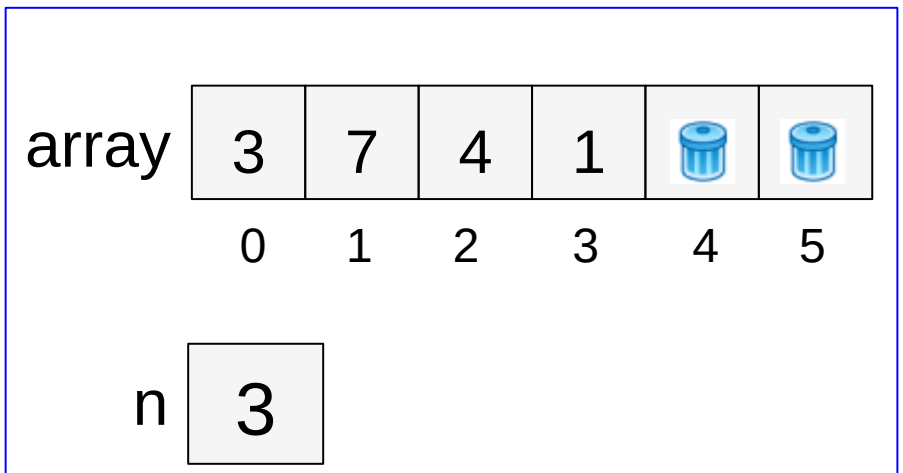


Algoritmo em C

```
// Exemplo: remover(2)

int remover(int pos) {

   if (n == 0 || pos < 0 || pos >= n)
      exit(1);

   int resp = array[pos];
   n--;

   for (int i = pos; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 4

i 2



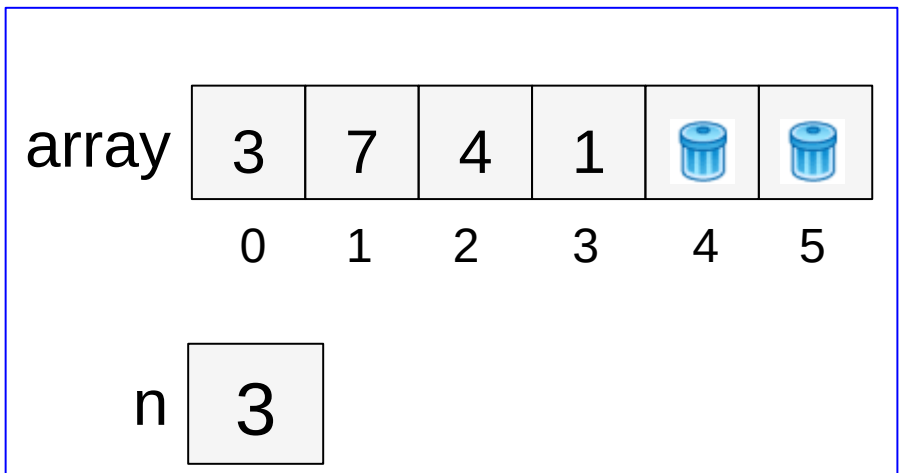
Algoritmo em C

```
// Exemplo: remover(2)

int remover(int pos) {

   if (n == 0 || pos < 0 || pos >= n)
      exit(1);

   int resp = array[pos];
   n--;

   for (int i = pos; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 4

i 2

true: 2 < 3



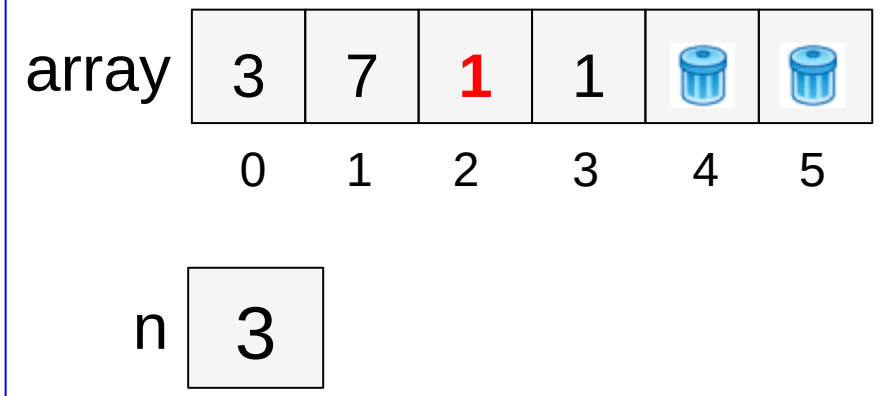
Algoritmo em C

```
// Exemplo: remover(2)

int remover(int pos) {

   if (n == 0 || pos < 0 || pos >= n)
      exit(1);

   int resp = array[pos];
   n--;

   for (int i = pos; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 4

i 2

array[2] ← array[3]

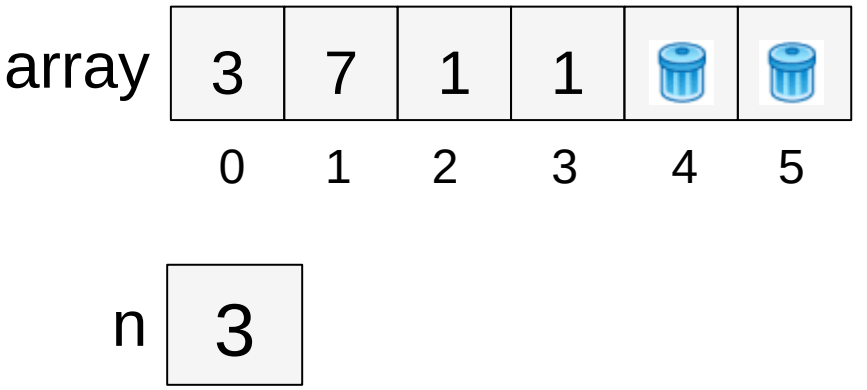


Algoritmo em C

```
// Exemplo: remover(2)

int remover(int pos) {

   if (n == 0 || pos < 0 || pos >= n)
      exit(1);

   int resp = array[pos];
   n--;

   for (int i = pos; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 4

i 3



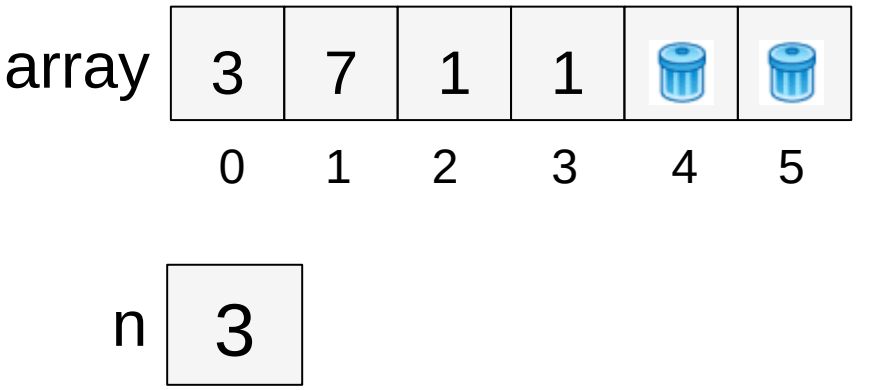
Algoritmo em C

```
// Exemplo: remover(2)

int remover(int pos) {

   if (n == 0 || pos < 0 || pos >= n)
      exit(1);

   int resp = array[pos];
   n--;

   for (int i = pos; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 4

i 3

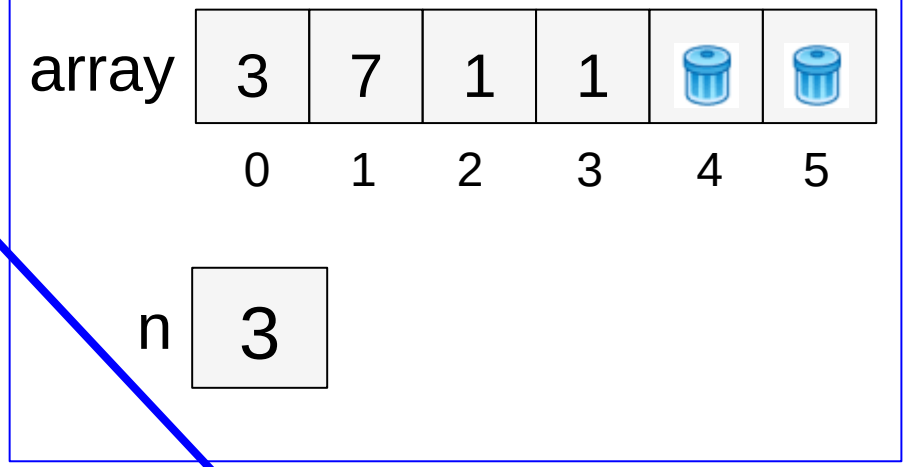
false: 3 < 3



Algoritmo em C

```
// Exemplo: remover(2)

int remover(int pos) {

   if (n == 0 || pos < 0 || pos >= n)
      exit(1);

   int resp = array[pos];
   n--;

   for (int i = pos; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 4

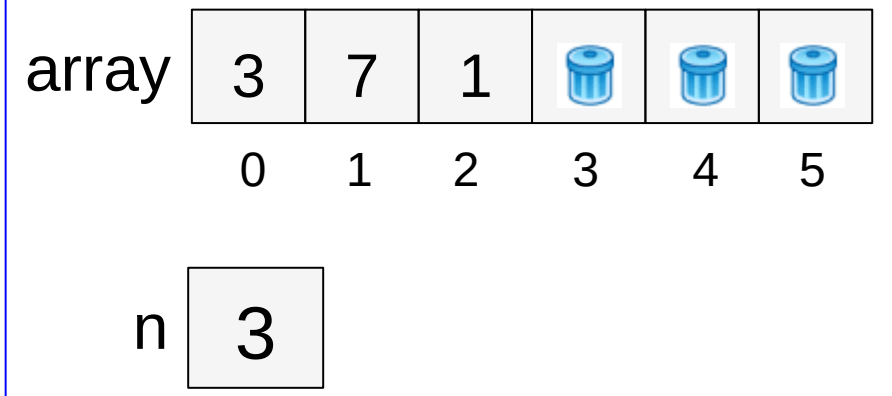


Retornando o 4

Algoritmo em C

```
// Exemplo: remover(2)

int remover(int pos) {

   if (n == 0 || pos < 0 || pos >= n)
      exit(1);

   int resp = array[pos];
   n--;

   for (int i = pos; i < n; i++){
      array[i] = array[i+1];
   }

   return resp;
}
```

resp 4

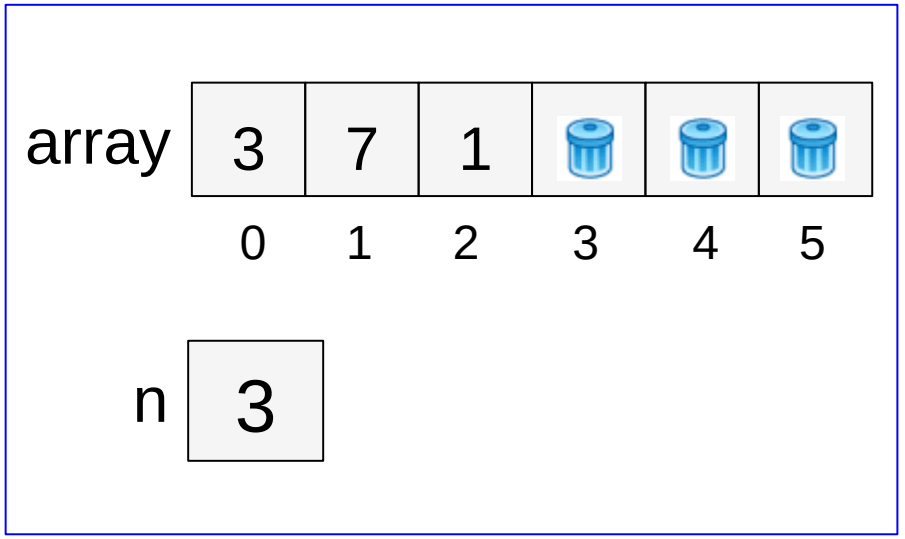


Algoritmo em C

```
int array[MAXTAM];
int n;


void start(){
    n = 0;
}

void inserirInicio(int x)   {  ...  }
void inserirFim(int x)   {  ...  }
void inserir(int x, int pos)   {  ...  }
int removerInicio()   {  ...  }
int removerFim()   {  ...  }
int remover(int pos)   {  ...  }
void mostrar () {  ...  }
```

Algoritmo em C

```
void mostrar (){
   printf("[ ");
   for (int i = 0; i < n; i++){
      printf(array[i] + "  ");
   }
   printf(" ]");
}
```



Exercício: O que será mostrado na tela?

Tela

Agenda

- Lista
- **Pilha**
- Fila

Introdução

- As pilhas são um Tipo Abstrato de Dados (TAD) no qual o primeiro elemento que entra é o último a sair
  - ***F****irst* ***I****n,* ***L****ast* ***O****ut* (FILO)
- Tem basicamente os métodos de inserir (empilhar) e remover (desempilhar)

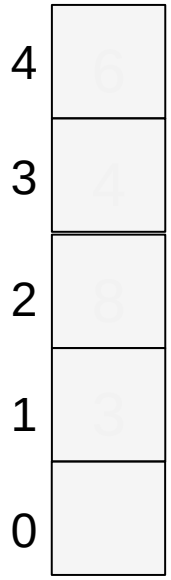
Exemplos

Exercício

- Dado o código da lista (métodos II, IF, I, RI, RF e R), como podemos alterá-lo para criarmos uma pilha? Apresente as duas soluções possíveis. Por que a segunda não é interessante?

Exercício

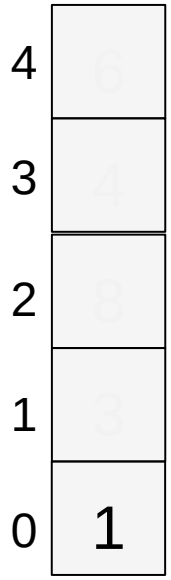
- Primeira solução IF e RF
  - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:



- Segunda solução II e RI (inserção e remoção não eficientes)

Exercício

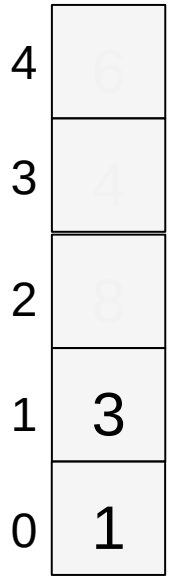
- Primeira solução IF e RF
  - Por exemplo, inserindo o **1**, 3, 5 e 7 e efetuando duas remoções teremos:



- Segunda solução II e RI (inserção e remoção não eficientes)

Exercício

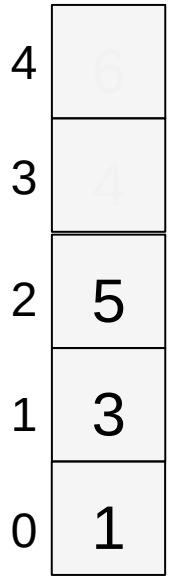
- Primeira solução IF e RF
  - Por exemplo, inserindo o 1, **3**, 5 e 7 e efetuando duas remoções teremos:



- Segunda solução II e RI (inserção e remoção não eficientes)

Exercício

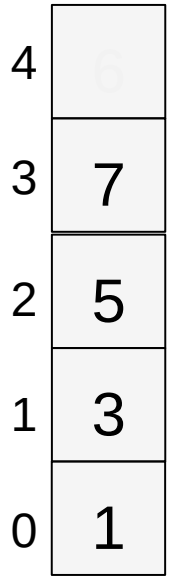
- Primeira solução IF e RF
  - Por exemplo, inserindo o 1, 3, **5** e 7 e efetuando duas remoções teremos:



- Segunda solução II e RI (inserção e remoção não eficientes)

Exercício

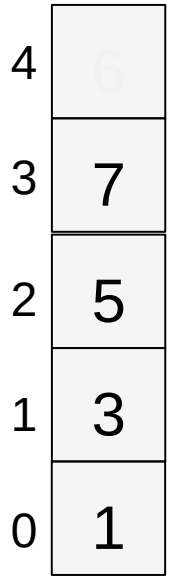
- Primeira solução IF e RF
  - Por exemplo, inserindo o 1, 3, 5 e **7** e efetuando duas remoções teremos:



- Segunda solução II e RI (inserção e remoção não eficientes)

Exercício

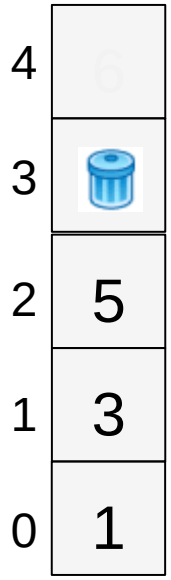
- Primeira solução IF e RF
  - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:



- Segunda solução II e RI (inserção e remoção não eficientes)

Exercício

- Primeira solução IF e RF
  - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:



Na primeira remoção, retiramos o número 7

- Segunda solução II e RI (inserção e remoção não eficientes)

Exercício

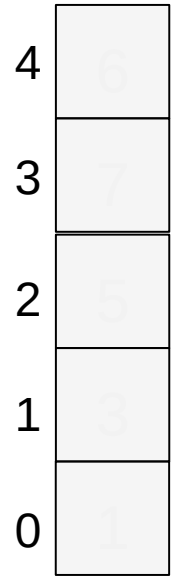
- Primeira solução IF e RF
  - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:

Na segunda remoção, retiramos o número 5

- Segunda solução II e RI (inserção e remoção não eficientes)

Exercício

- Primeira solução IF e RF



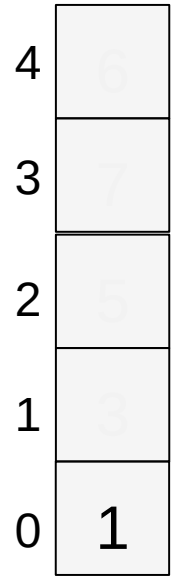
- Segunda solução II e RI (inserção e remoção não eficientes)
  - Por exemplo, inserindo o 1, 3, 5 e 7 e

efetuando duas remoções teremos:

Em cada inserção ou remoção, movemos todos os elementos

Exercício

- Primeira solução IF e RF



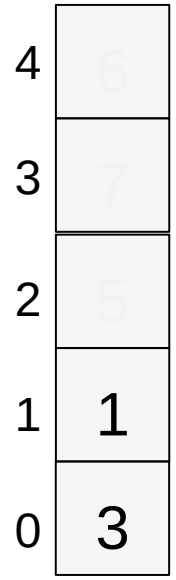
- Segunda solução II e RI (inserção e remoção não eficientes)
  - Por exemplo, inserindo o **1**, 3, 5 e 7 e

efetuando duas remoções teremos:

Em cada inserção ou remoção, movemos todos os elementos

Exercício

- Primeira solução IF e RF



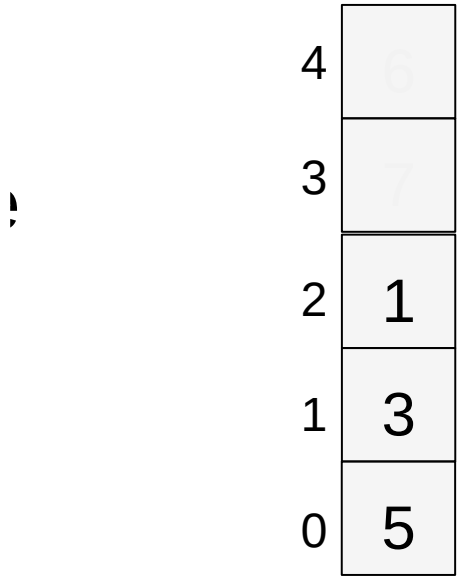
- Segunda solução II e RI (inserção e remoção não eficientes)
  - Por exemplo, inserindo o 1, **3**, 5 e 7 e

efetuando duas remoções teremos:

Em cada inserção ou remoção, movemos todos os elementos

Exercício

- Primeira solução IF e RF



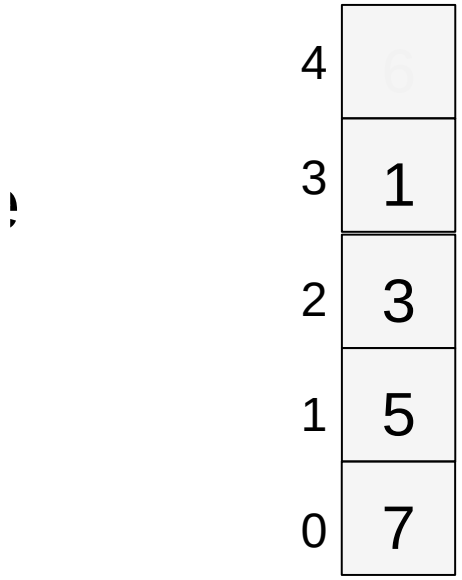
- Segunda solução II e RI (inserção e remoção não eficientes)
  - Por exemplo, inserindo o 1, 3, **5** e 7 e

efetuando duas remoções teremos:

Em cada inserção ou remoção, movemos todos os elementos

Exercício

- Primeira solução IF e RF



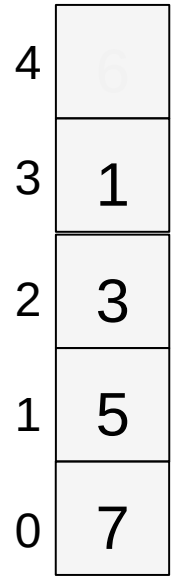
- Segunda solução II e RI (inserção e remoção não eficientes)
  - Por exemplo, inserindo o 1, 3, 5 e **7** e

efetuando duas remoções teremos:

Em cada inserção ou remoção, movemos todos os elementos

Exercício

- Primeira solução IF e RF



Primeira remoção: Retorna o 7 e move todos os demais

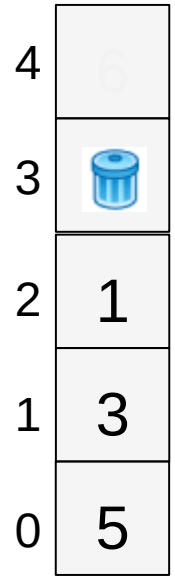
- Segunda solução II e RI (inserção e remoção não eficientes)
  - Por exemplo, inserindo o 1, 3, 5 e 7 e

efetuando duas remoções teremos:

Em cada inserção ou remoção, movemos todos os elementos

Exercício

- Primeira solução IF e RF



Primeira remoção: Retorna o 7 e move todos os demais

- Segunda solução II e RI (inserção e remoção não eficientes)
  - Por exemplo, inserindo o 1, 3, 5 e 7 e

efetuando duas remoções teremos:

Em cada inserção ou remoção, movemos todos os elementos

Exercício

- Primeira solução IF e RF

- Segunda soluçã ção não eficientes)
  - Por exemplo,

efetuando duas remoções teremos:

Primeira remoção: Retorna o 7 e move todos os demais

Segunda remoção: Retorna o 5 e move todos os demais

Em cada inserção ou remoção, movemos todos os elementos

Agenda

- Lista
- Pilha
- **Fila**

Introdução

- As filas são um Tipo Abstrato de Dados (TAD) no qual o primeiro elemento que entra é o primeiro a sair
  - ***F****irst* ***I****n,* ***F****irst* ***O****ut* (FIFO)
- Tem basicamente os métodos de inserir (enfileirar) e remover (desenfileirar)

Exemplos

Exercício

- Dado o código da lista (métodos II, IF, I, RI, RF e R), como podemos alterá-lo para criarmos uma fila? Apresente as duas soluções possíveis e mostre a desvantagem de cada uma

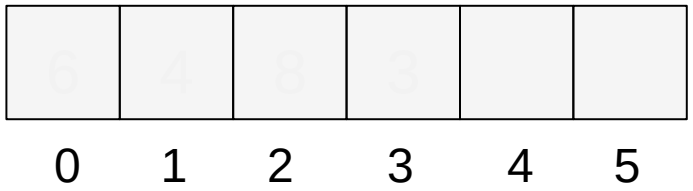
Exercício

- Primeira solução IF e RI (**remoção não é eficiente**)

- Segunda solução II e RF (**inserção não é eficiente**)

Exercício

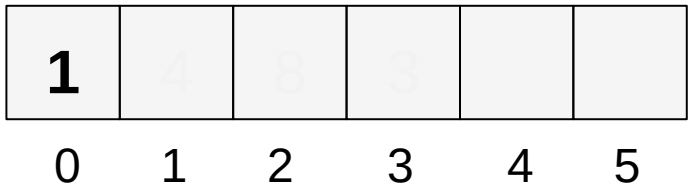
- Primeira solução IF e RI (remoção não é eficiente)
  - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:



- Segunda solução II e RF (inserção não é eficiente)

Exercício

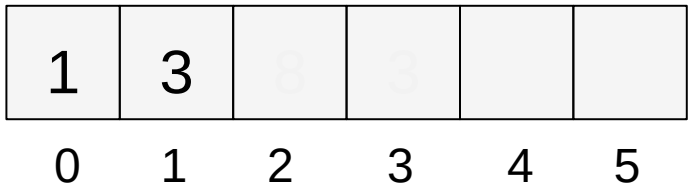
- Primeira solução IF e RI (remoção não é eficiente)
  - Por exemplo, inserindo o **1**, 3, 5 e 7 e efetuando duas remoções teremos:



- Segunda solução II e RF (inserção não é eficiente)

Exercício

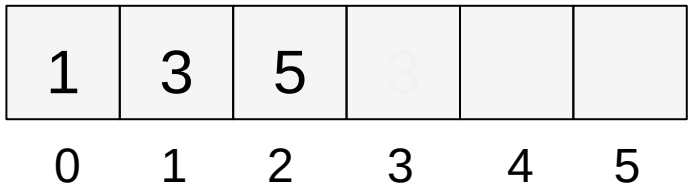
- Primeira solução IF e RI (remoção não é eficiente)
  - Por exemplo, inserindo o 1, **3**, 5 e 7 e efetuando duas remoções teremos:



- Segunda solução II e RF (inserção não é eficiente)

Exercício

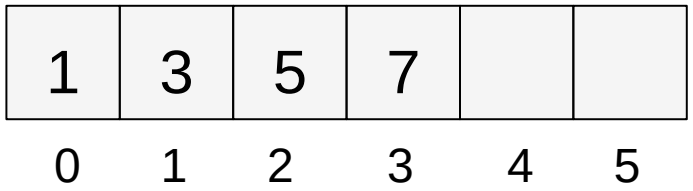
- Primeira solução IF e RI (remoção não é eficiente)
  - Por exemplo, inserindo o 1, 3, **5** e 7 e efetuando duas remoções teremos:



- Segunda solução II e RF (inserção não é eficiente)

Exercício

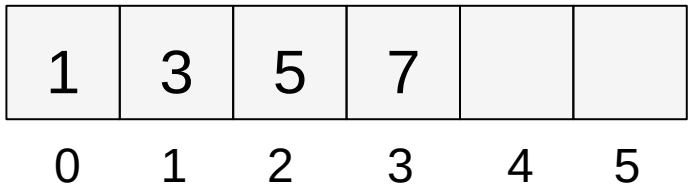
- Primeira solução IF e RI (remoção não é eficiente)
  - Por exemplo, inserindo o 1, 3, 5 e **7** e efetuando duas remoções teremos:



- Segunda solução II e RF (inserção não é eficiente)

Exercício

- Primeira solução IF e RI (**remoção não é eficiente**)
  - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:

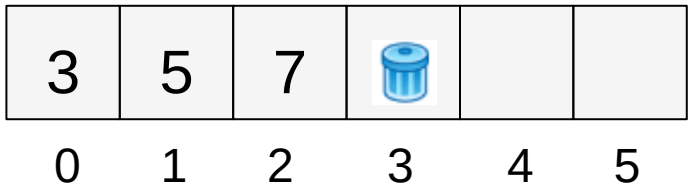


- Segunda solução (eficiente)

Primeira remoção: Retorna o 1 e move todos os demais

Exercício

- Primeira solução IF e RI (**remoção não é eficiente**)
  - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:



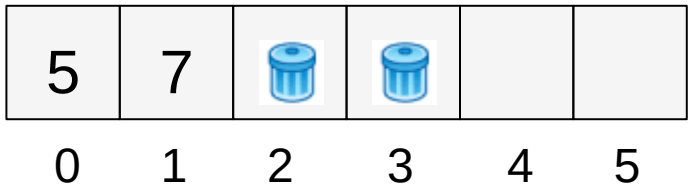
- Segunda soluçã

Primeira remoção: Retorna o 1 e move todos os demais

eficiente)

Exercício

- Primeira solução IF e RI (**remoção não é eficiente**)
  - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:



- Segunda soluçã eficiente)

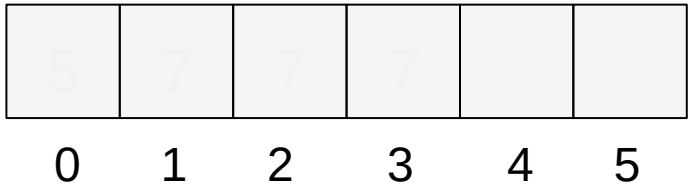
Primeira remoção: Retorna o 1 e move todos os demais

Segunda remoção: Retorna o 3 e move todos os demais

Exercício

- Primeira solução IF e RI (remoção não é eficiente)

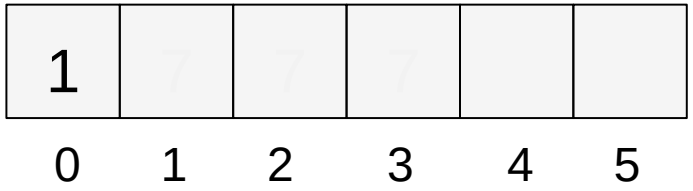
- Segunda solução II e RF (inserção não é eficiente)
  - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:



Exercício

- Primeira solução IF e RI (remoção não é eficiente)

- Segunda solução II e RF (inserção não é eficiente)
  - Por exemplo, inserindo o **1**, 3, 5 e 7 e efetuando duas remoções teremos:

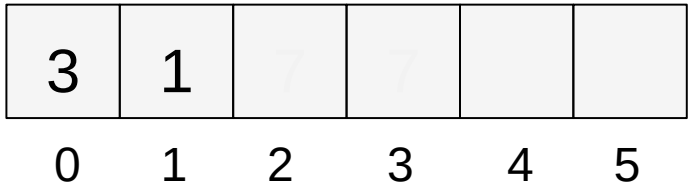


Exercício

- Primeira solução IF e RI (remoção não é eficiente)

Cada inserção: Move todos os elementos já cadastrados

- Segunda solução II e RF (**inserção não é eficiente**)
  - Por exemplo, inserindo o 1, **3**, 5 e 7 e efetuando duas remoções teremos:

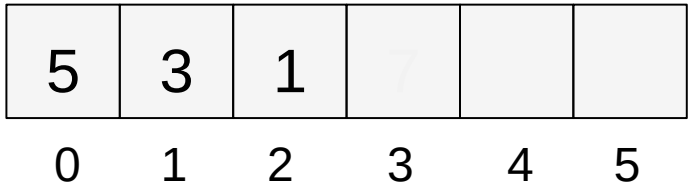


Exercício

- Primeira solução IF e RI (remoção não é eficiente)

Cada inserção: Move todos
os elementos já cadastrados

- Segunda solução II e RF (**inserção não é eficiente**)
  - Por exemplo, inserindo o 1, 3, **5** e 7 e efetuando duas remoções teremos:

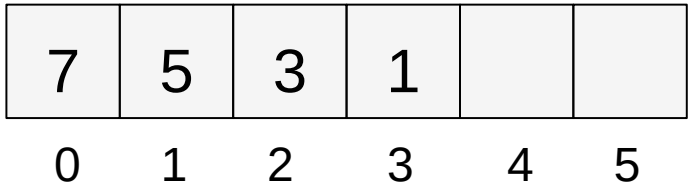


Exercício

- Primeira solução IF e RI (remoção não é eficiente)

Cada inserção: Move todos
os elementos já cadastrados

- Segunda solução II e RF (**inserção não é eficiente**)
  - Por exemplo, inserindo o 1, 3, 5 e **7** e efetuando duas remoções teremos:

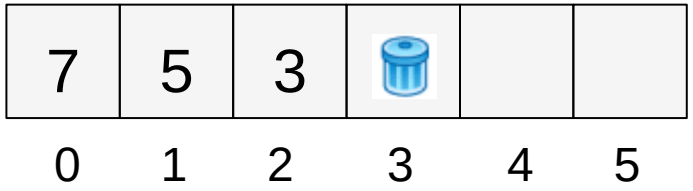


Exercício

- Primeira solução IF e RI (remoção não é eficiente)

Na primeira remoção,
retiramos o número 1

- Segunda solução II e RF (inserção não é eficiente)
  - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:

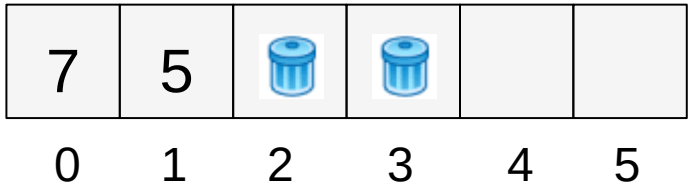


Exercício

- Primeira solução IF e RI (remoção não é eficiente)

Na segunda remoção,
retiramos o número 3

- Segunda solução II e RF (inserção não é eficiente)
  - Por exemplo, inserindo o 1, 3, 5 e 7 e efetuando duas remoções teremos:

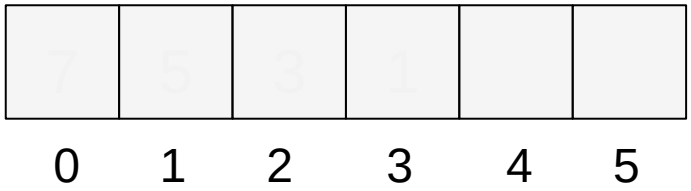


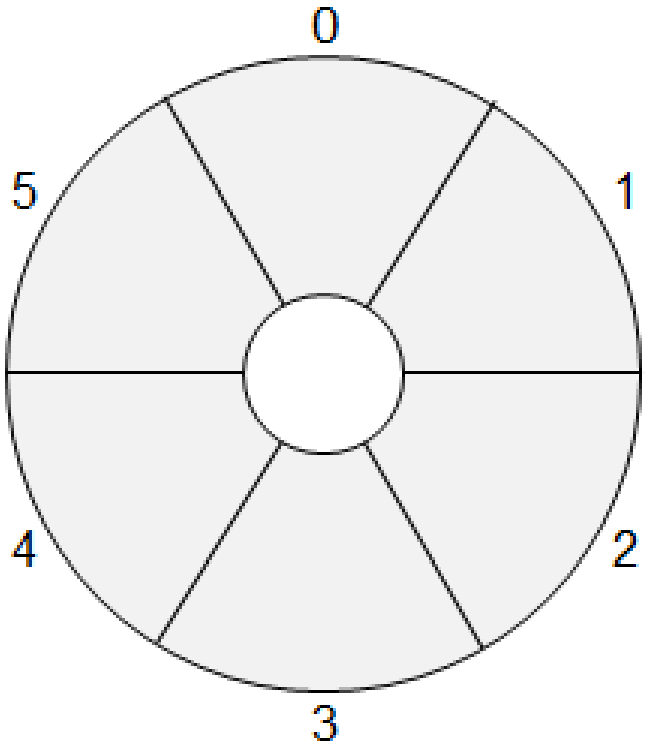
Dúvida

- Como implementar uma fila sem que uma das operações desloque todos os elementos?

Dúvida

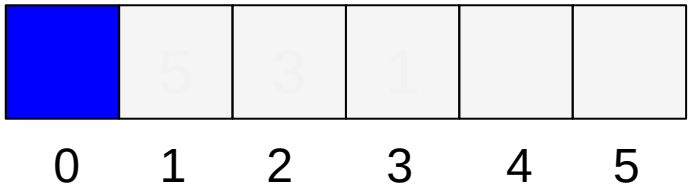
- Como implementar uma fila sem que uma das operações desloque todos os elementos?
  - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira

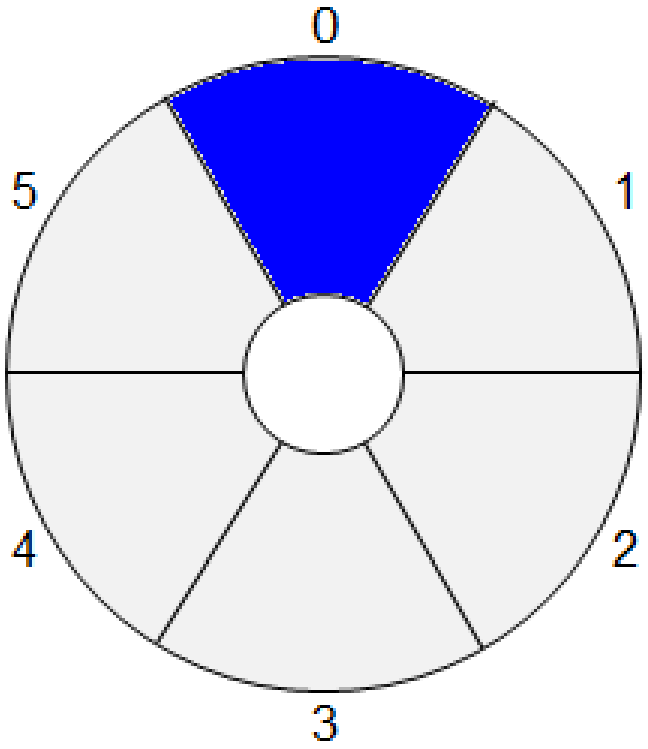




Dúvida

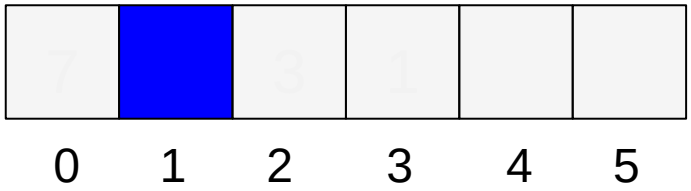
- Como implementar uma fila sem que uma das operações desloque todos os elementos?
  - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira

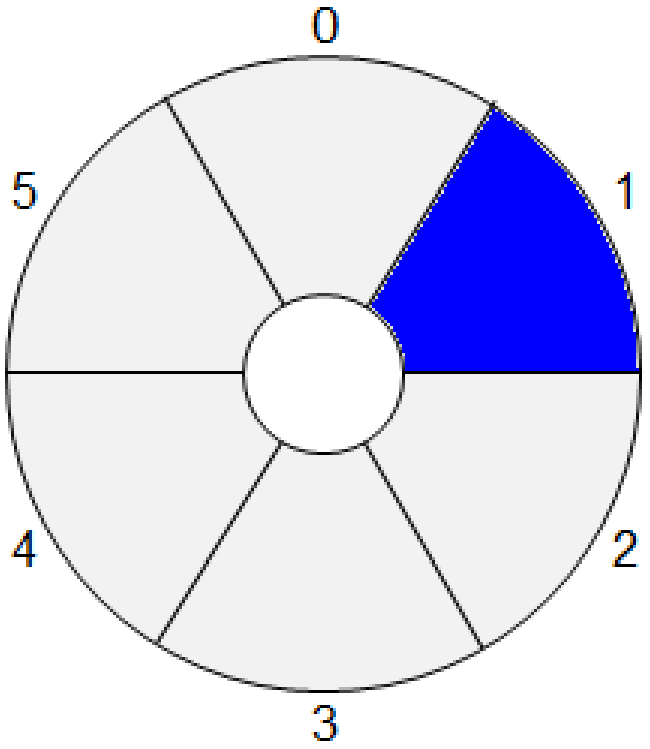




Dúvida

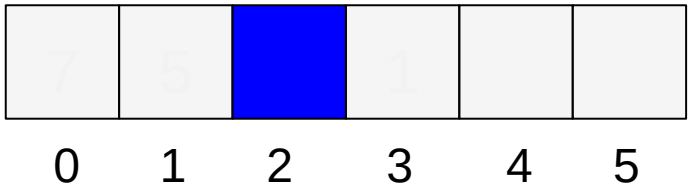
- Como implementar uma fila sem que uma das operações desloque todos os elementos?
  - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira

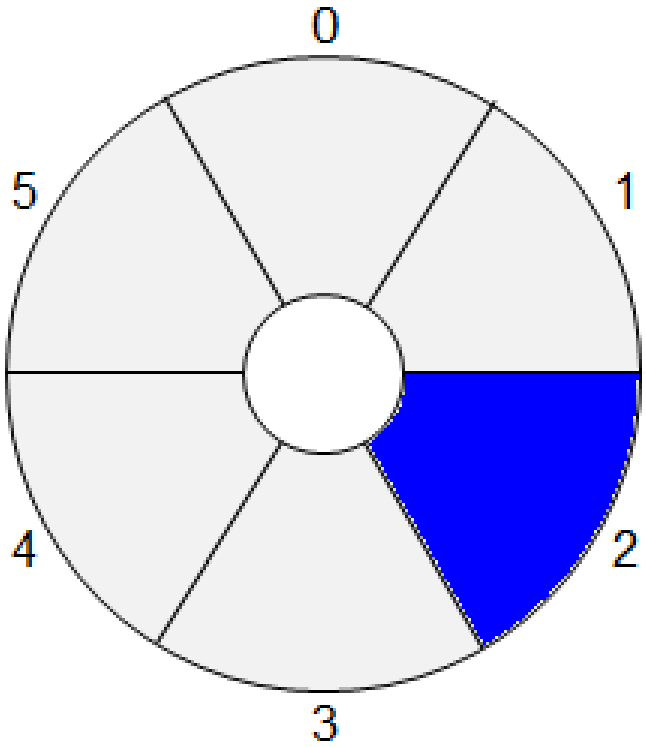




Dúvida

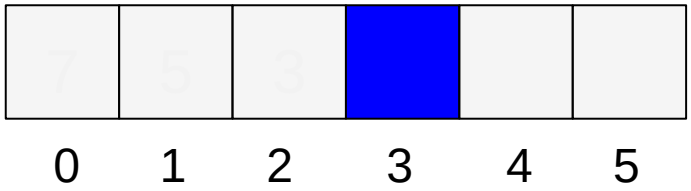
- Como implementar uma fila sem que uma das operações desloque todos os elementos?
  - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira

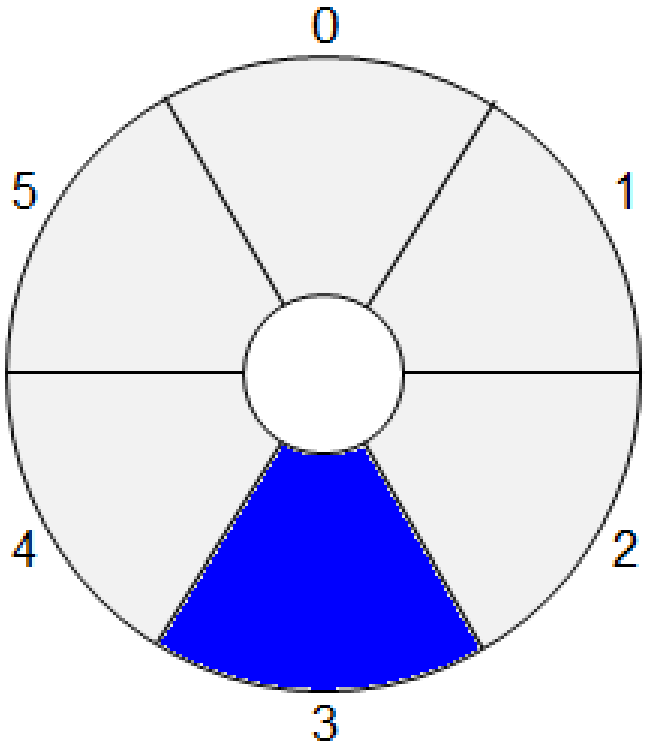




Dúvida

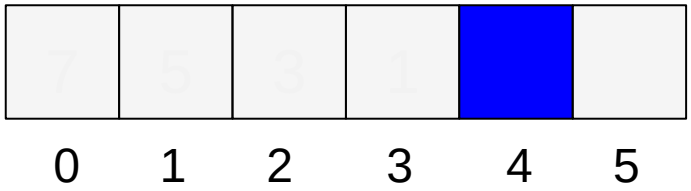
- Como implementar uma fila sem que uma das operações desloque todos os elementos?
  - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira

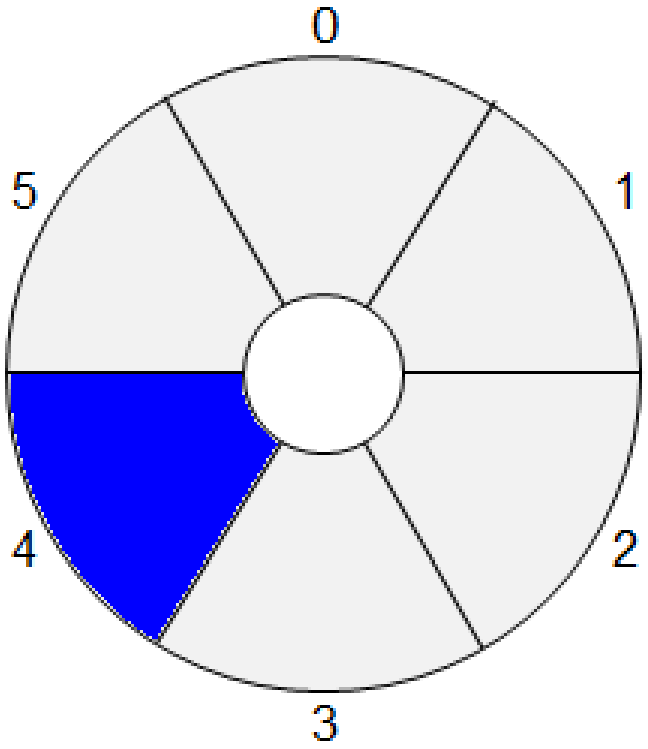




Dúvida

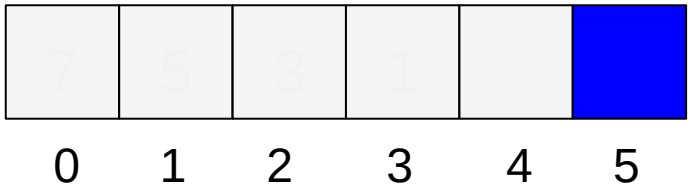
- Como implementar uma fila sem que uma das operações desloque todos os elementos?
  - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira

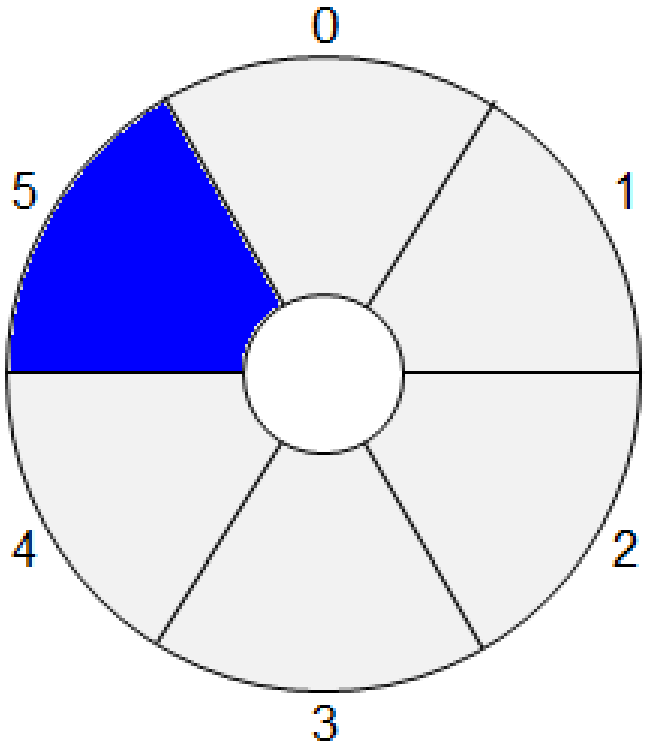




Dúvida

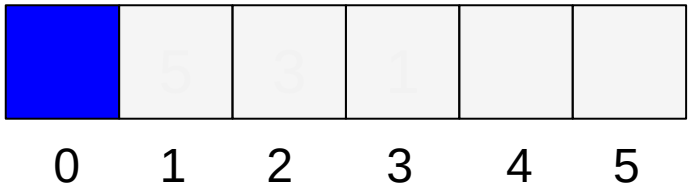
- Como implementar uma fila sem que uma das operações desloque todos os elementos?
  - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira

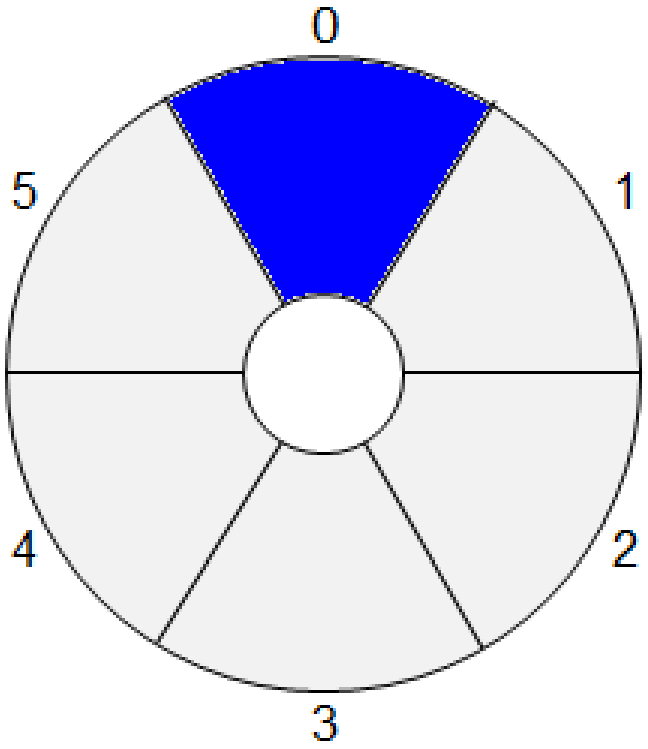




Dúvida

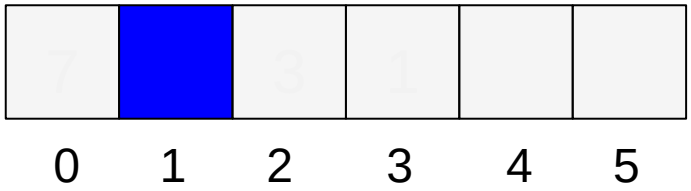
- Como implementar uma fila sem que uma das operações desloque todos os elementos?
  - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira

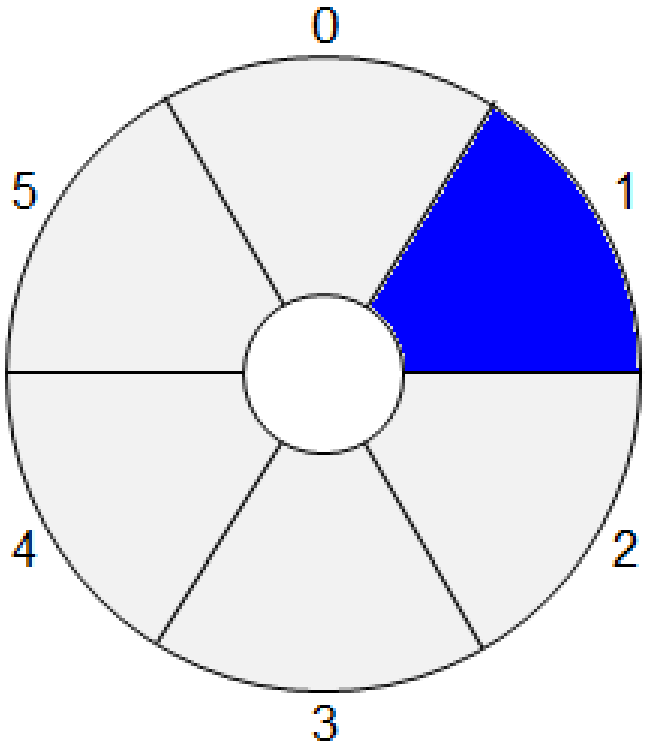




Dúvida

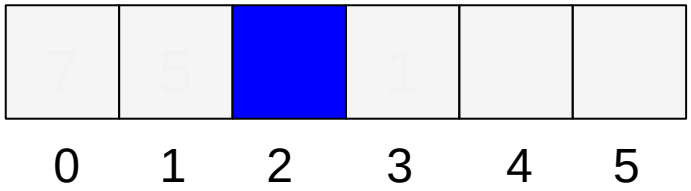
- Como implementar uma fila sem que uma das operações desloque todos os elementos?
  - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira

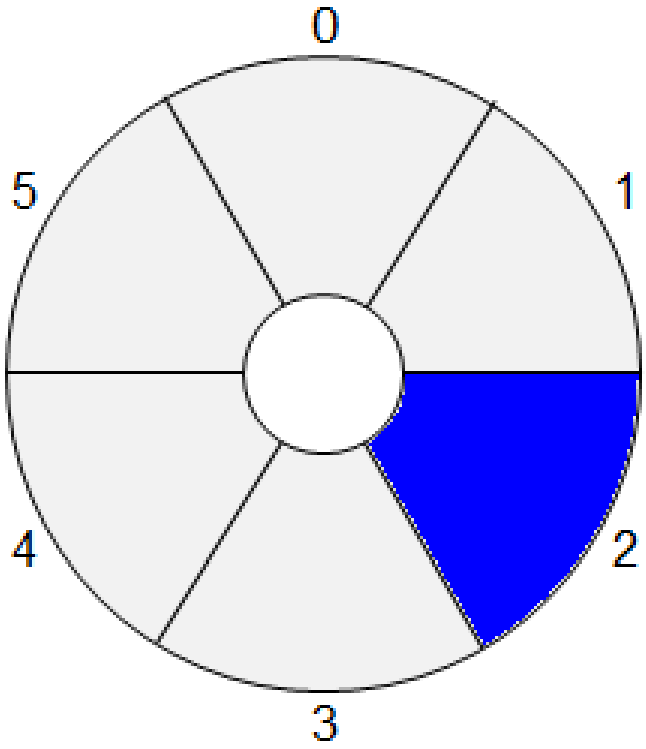




Dúvida

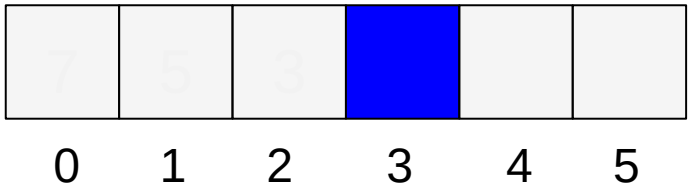
- Como implementar uma fila sem que uma das operações desloque todos os elementos?
  - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira

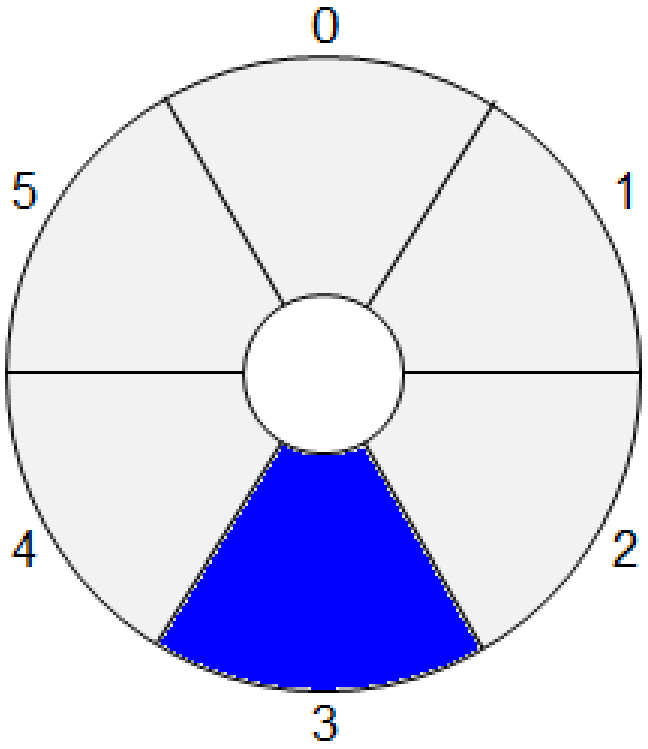




Dúvida

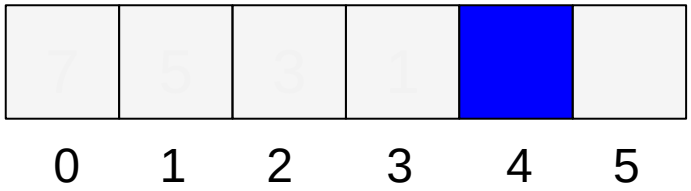
- Como implementar uma fila sem que uma das operações desloque todos os elementos?
  - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira

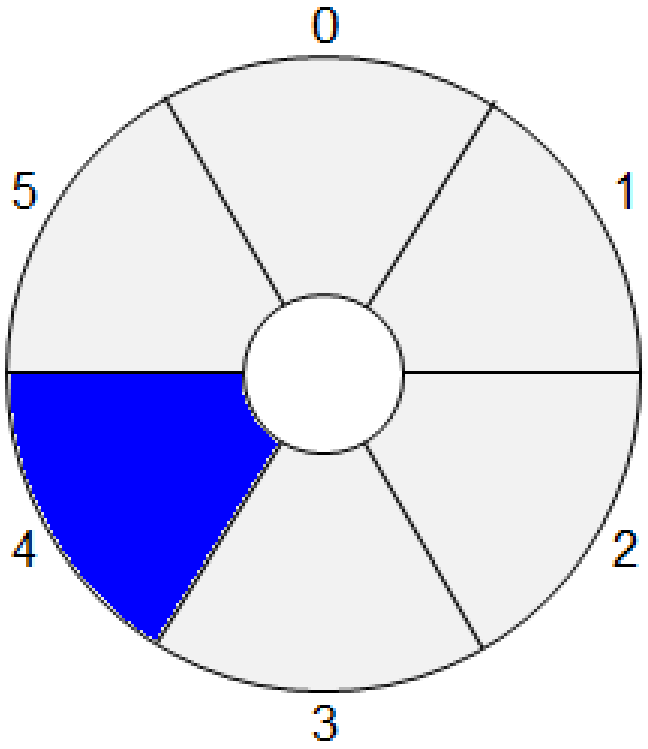




Dúvida

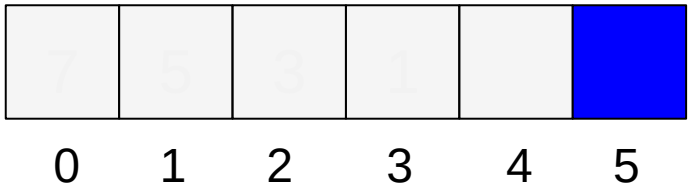
- Como implementar uma fila sem que uma das operações desloque todos os elementos?
  - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira

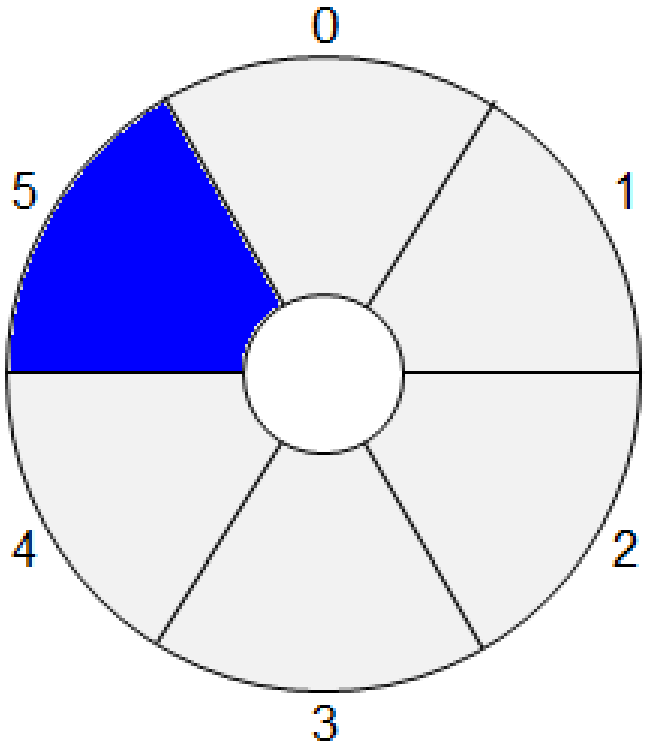




Dúvida

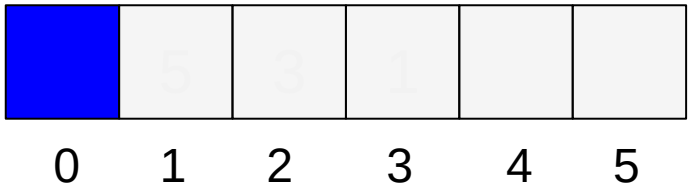
- Como implementar uma fila sem que uma das operações desloque todos os elementos?
  - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira

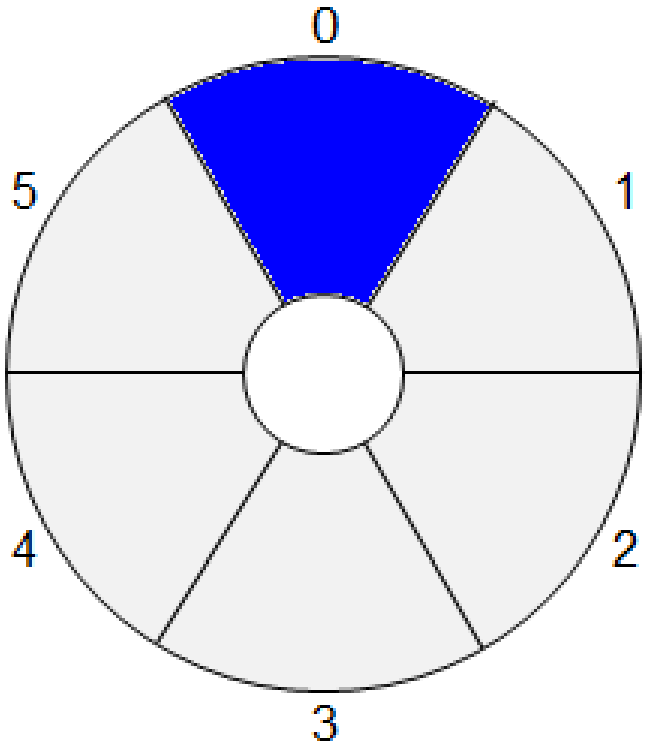




Dúvida

- Como implementar uma fila sem que uma das operações desloque todos os elementos?
  - Fazendo uma fila circular, ou seja, depois da última posição, retornamos para a primeira





Exercício

- 0 % 5 =
- 1 % 5 =
- 2 % 5 =
- 3 % 5 =
- 4 % 5 =
- 5 % 5 =
- 6 % 5 =
- 7 % 5 =
- 8 % 5 =
- 9 % 5 =
- 10 % 5 =
- 2 % 5 = 1
- 12 % 5 =
- 4 % 5 = 3
- 14 % 5 =

Exercício

- 0 % 5 = 0
- 1 % 5 = 1
- 2 % 5 = 2
- 3 % 5 = 3
- 4 % 5 = 4
- 5 % 5 = 0
- 6 % 5 = 1
- 7 % 5 = 2
- 8 % 5 = 3
- 9 % 5 = 4
- 10 % 5 = 0
- 11 % 5 = 1
- 12 % 5 = 2
- 13 % 5 = 3
- 14 % 5 = 4

Exercício

- Faça o quadro de memória do programa abaixo

```
n = 0;
n = (n + 1) % 5;
n = (n + 1) % 5;
n = (n + 1) % 5;
n = (n + 1) % 5;
n = (n + 1) % 5;
n = (n + 1) % 5;
n = (n + 1) % 5;
n = (n + 1) % 5;
n = (n + 1) % 5;
n = (n + 1) % 5;
```

Algoritmo em C

```
int array[MAXTAM+1];
int primeiro, ultimo;


void start(){
   primeiro = ultimo = 0;
}
void inserir(int x) { ... }
int remover() { ... }
void mostrar (){ ... }
```

Algoritmo em C

```
int array[MAXTAM+1];
int primeiro, ultimo;


void start(){
   primeiro = ultimo = 0;
}
void inserir(int x) { ... }
int remover() { ... }
void mostrar (){ ... }
```

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

Algoritmo em C

```
int array[MAXTAM+1];
int primeiro, ultimo;

void start(){
   primeiro = ultimo = 0;
}
void inserir(int x) { ... }
int remover() { ... }
void mostrar (){ ... }
```

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

Algoritmo em C

```
int array[MAXTAM+1];
int primeiro, ultimo;

void start(){
   primeiro = ultimo = 0;
}
void inserir(int x) { ... }
int remover() { ... }
void mostrar (){ ... }
```

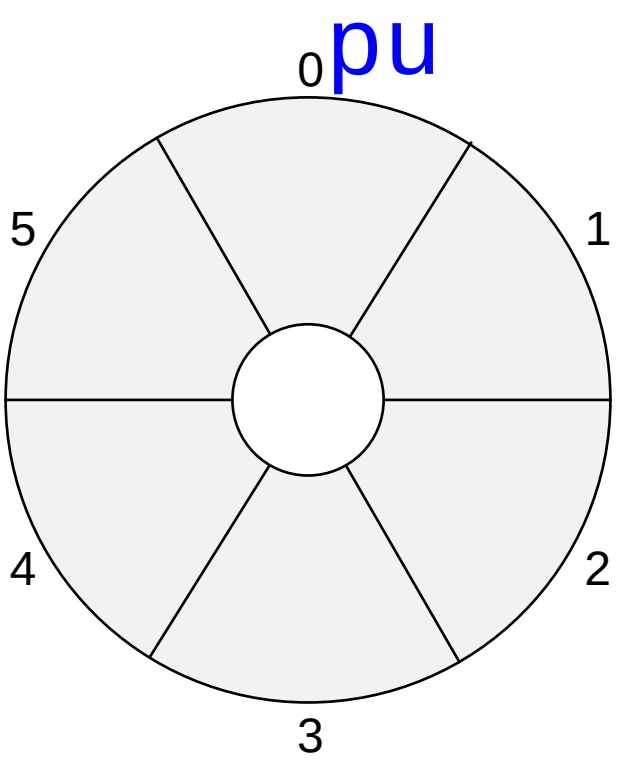
Vamos reservar uma unidade a mais, contudo, nossa fila caberá somente a quantidade solicitada

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

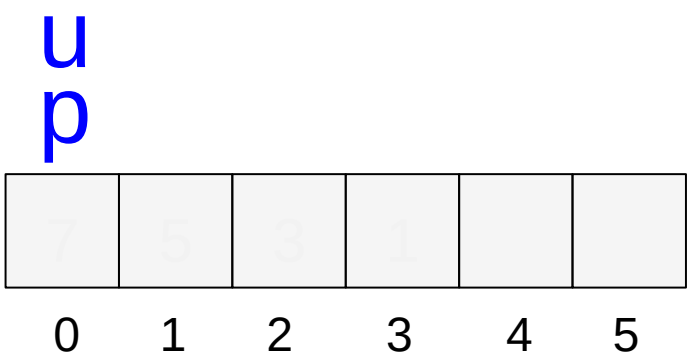
Algoritmo em C

```
int array[MAXTAM+1];
int primeiro, ultimo;
```

```
void start(){
   primeiro = ultimo = 0;
}
void inserir(int x) { ... }
int remover() { ... }
void mostrar (){ ... }
```

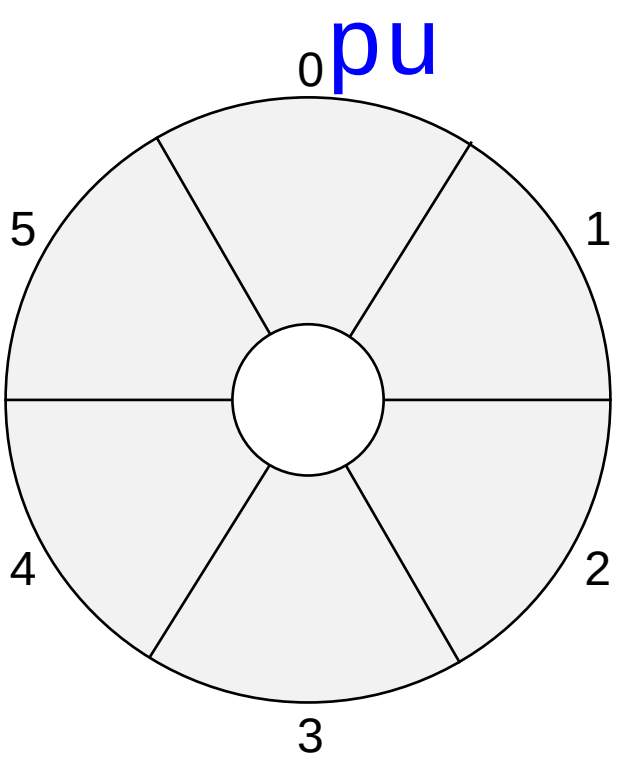


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

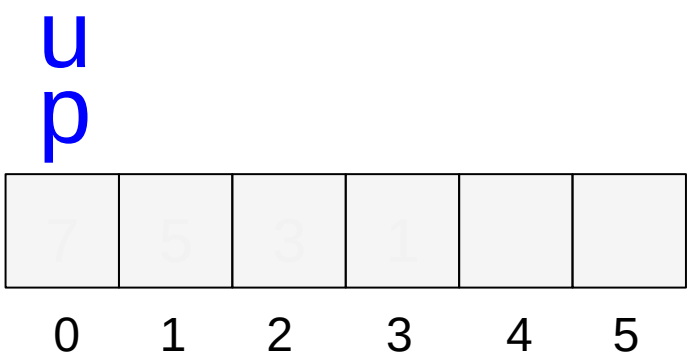


Algoritmo em C

```
//Inserir(1)

void inserir(int x) {

   if (((ultimo + 1) % MAXTAM) == primeiro)
      exit(1);

   array[ultimo] = x;
   ultimo = (ultimo + 1) % MAXTAM;
}
```



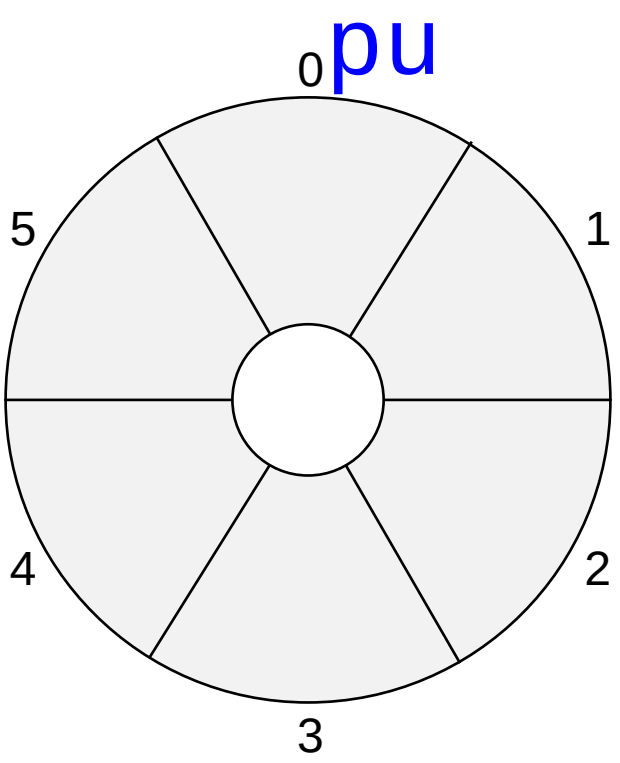
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



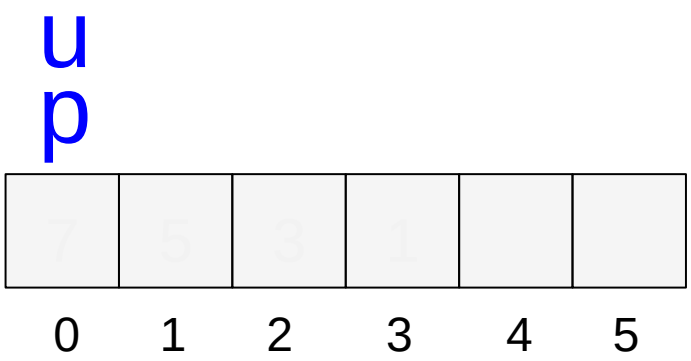
Algoritmo em C

```
//Inserir(1)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
        exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

false: 0 + 1 % 6 == 0

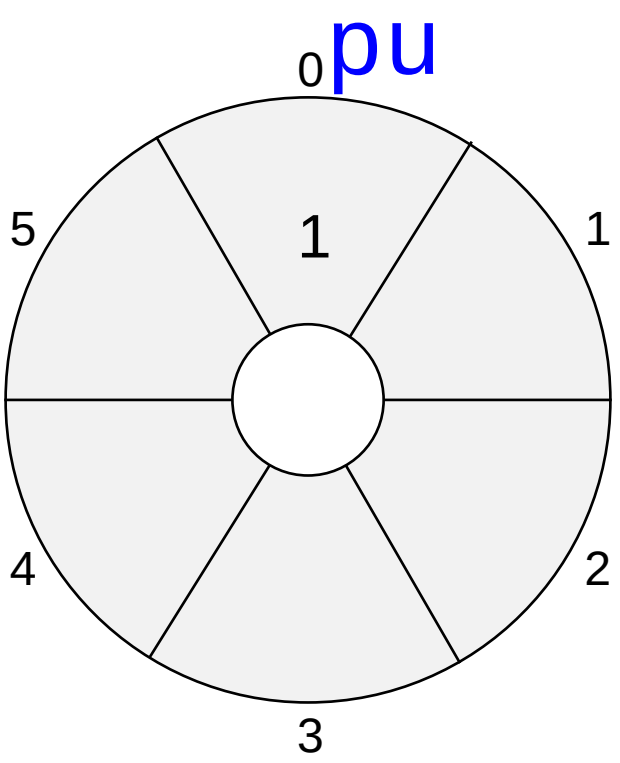


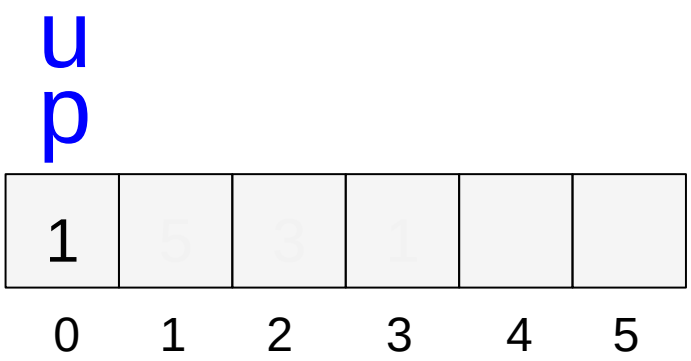
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em C

```
//Inserir(1)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

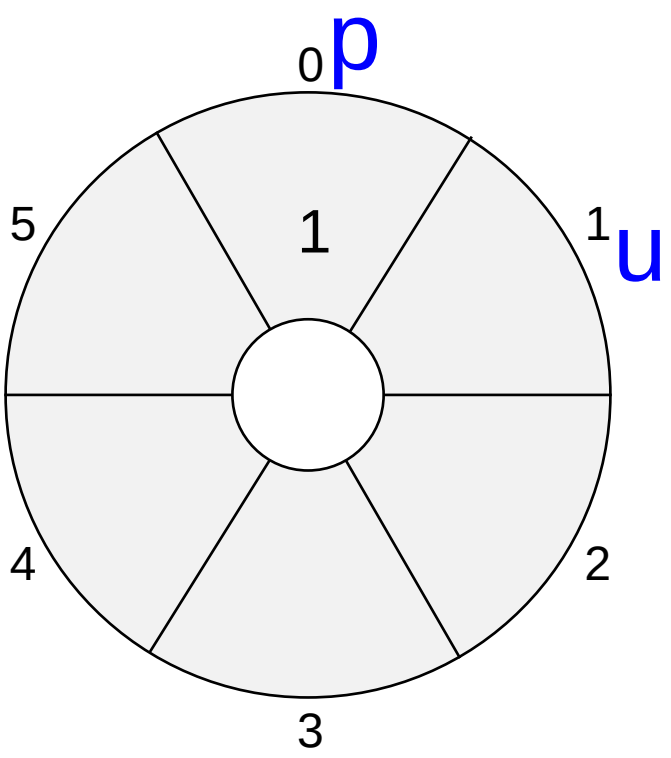




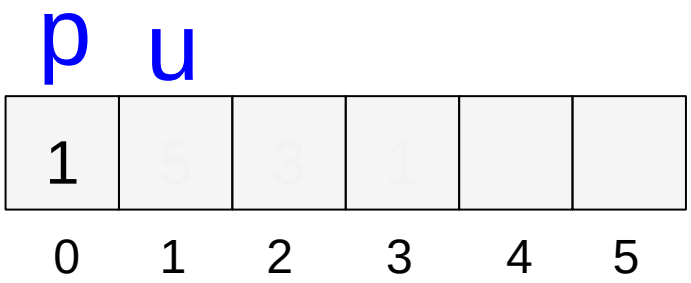
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

Algoritmo em C

```
//Inserir(1)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

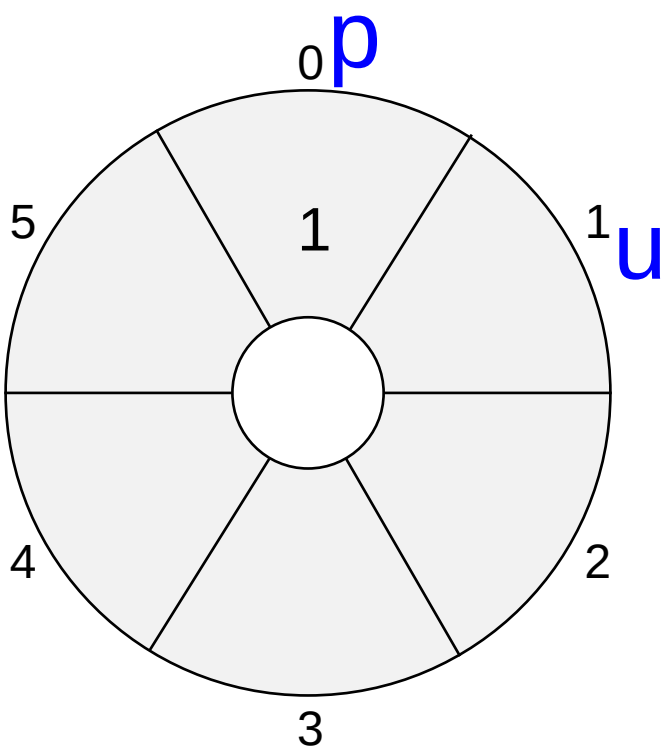


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

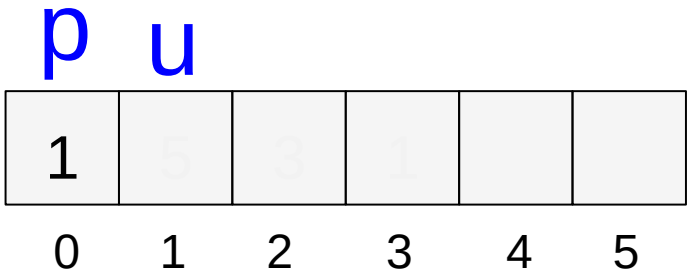


Algoritmo em C

```
//Inserir(1)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

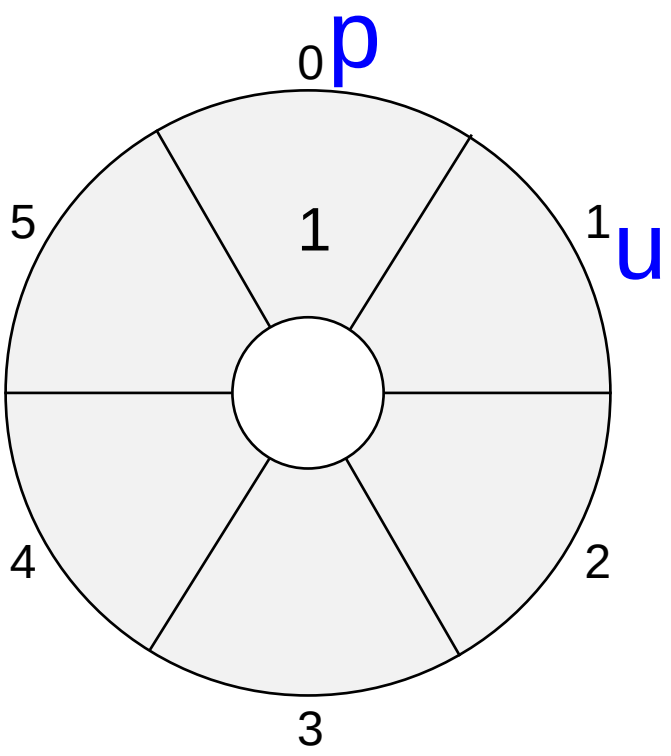


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

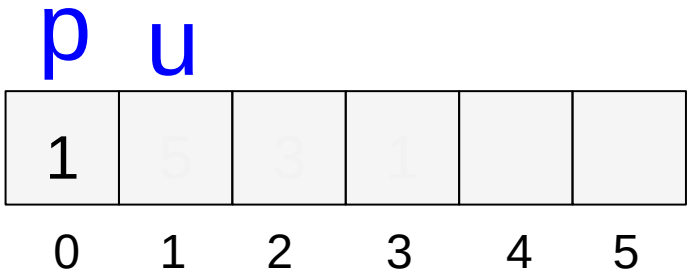


Algoritmo em C

```
//Inserir(3)

void inserir(int x) {

   if (((ultimo + 1) % MAXTAM) == primeiro)
      exit(1);

   array[ultimo] = x;
   ultimo = (ultimo + 1) % MAXTAM;
}
```



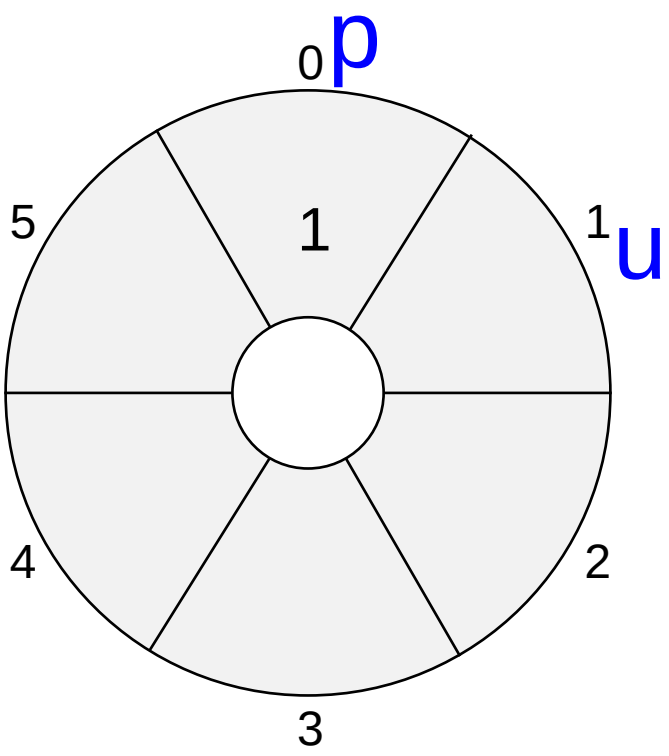
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



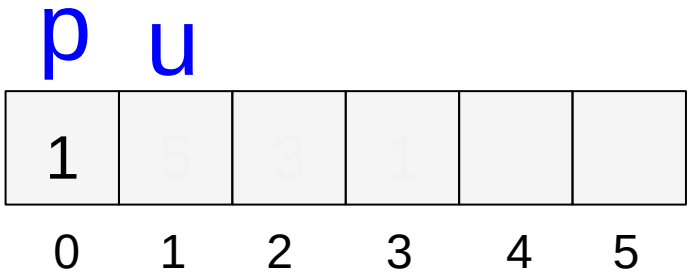
Algoritmo em C

```
//Inserir(3)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
        exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

false: 1 + 1 % 6 == 0

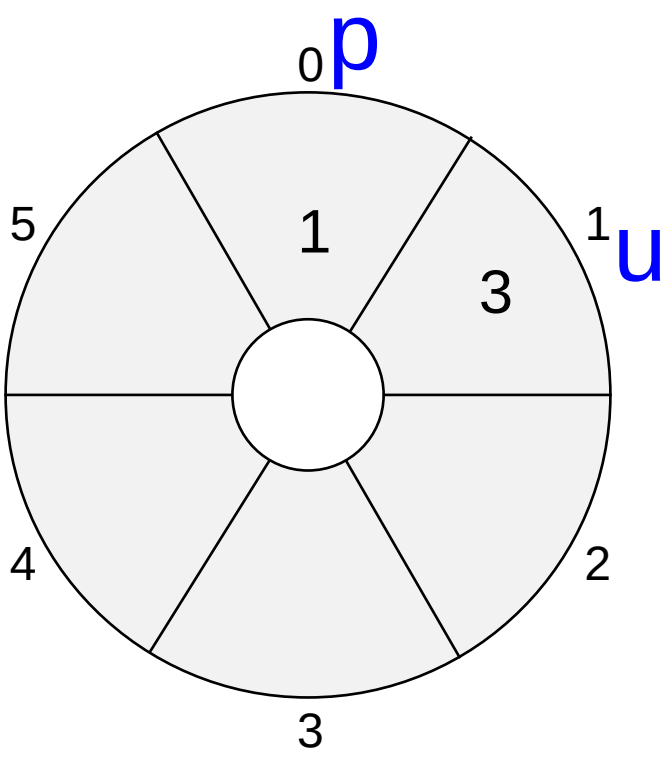


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

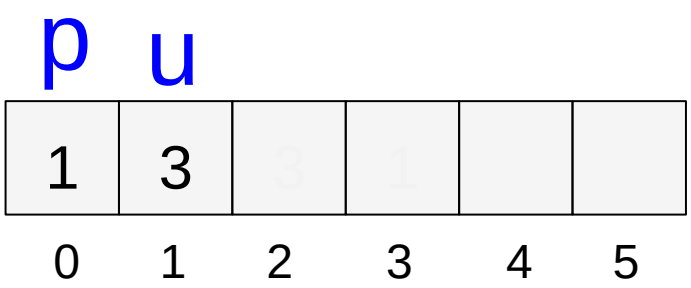


Algoritmo em C

```
//Inserir(3)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

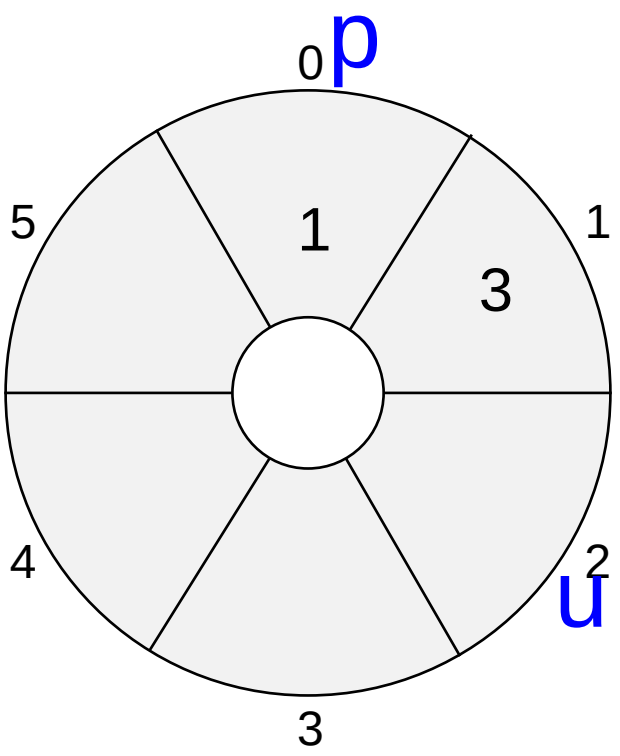


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

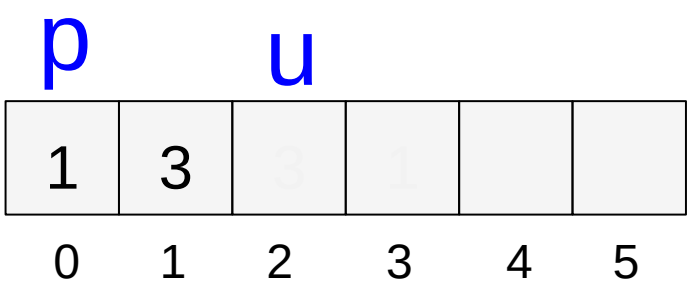


Algoritmo em C

```
//Inserir(3)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

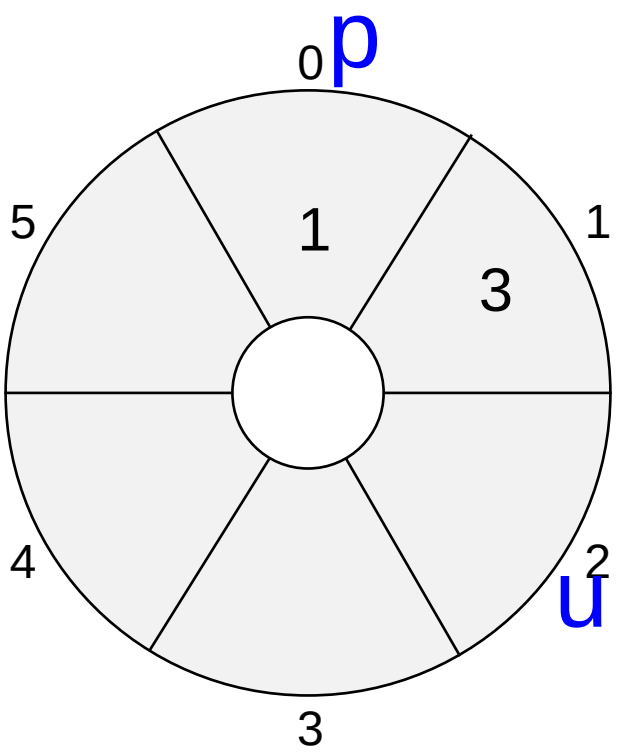


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

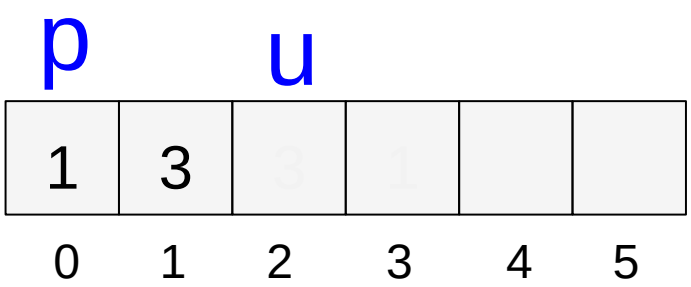


Algoritmo em C

```
//Inserir(3)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

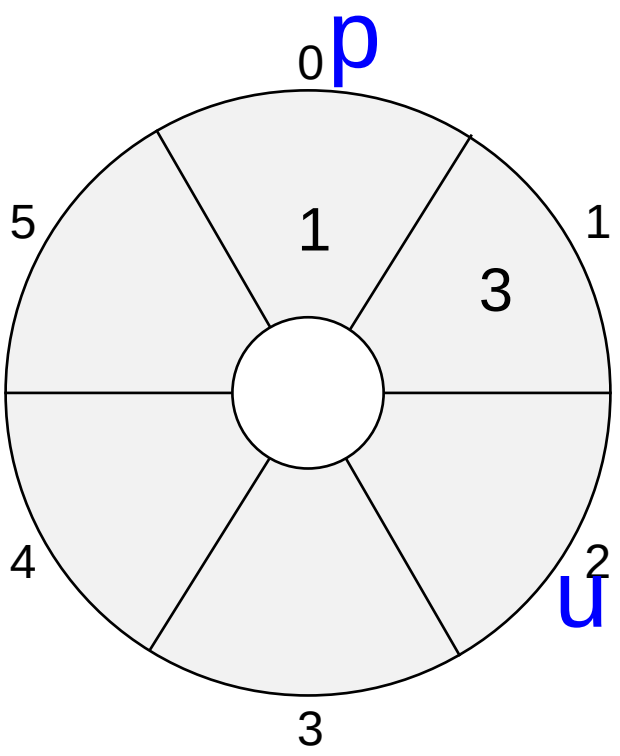


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

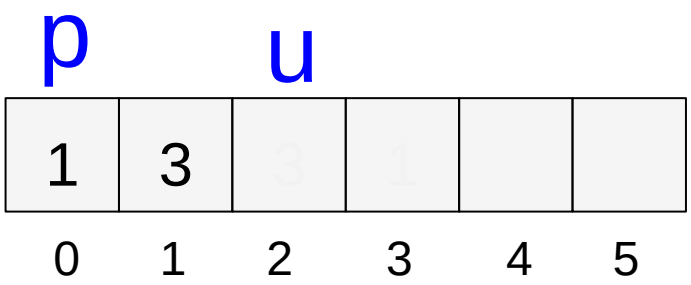


Algoritmo em C

```
//Inserir(5)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```



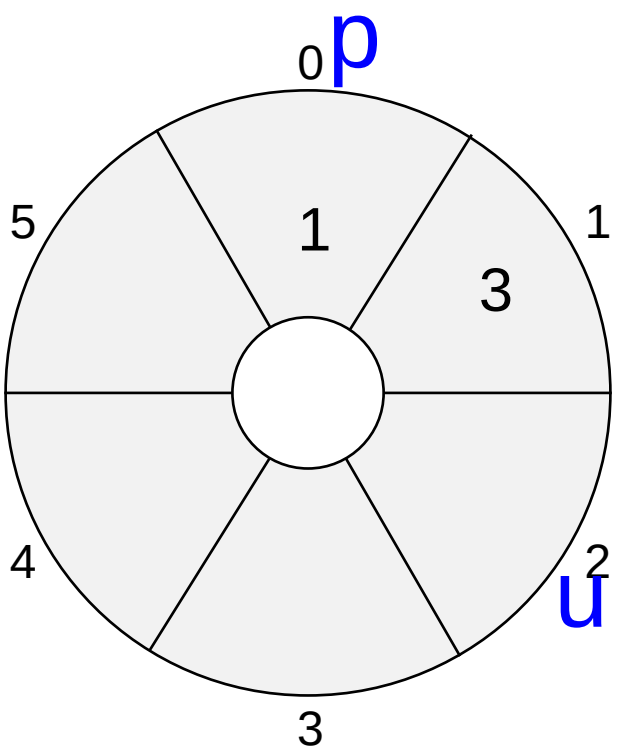
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



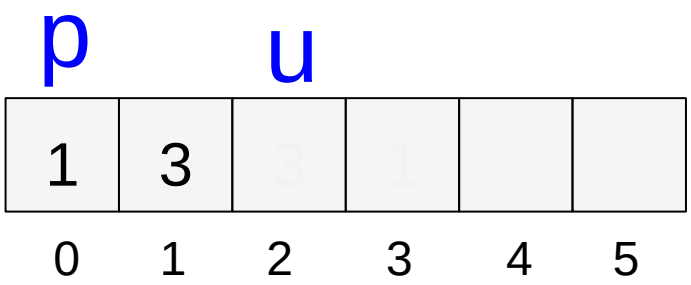
Algoritmo em C

```
//Inserir(5)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
        exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

false: 2 + 1 % 6 == 0

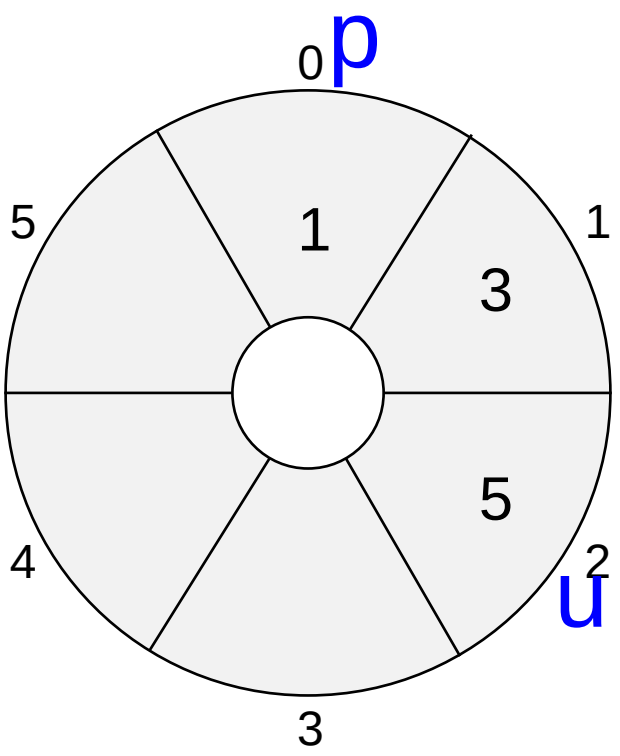


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

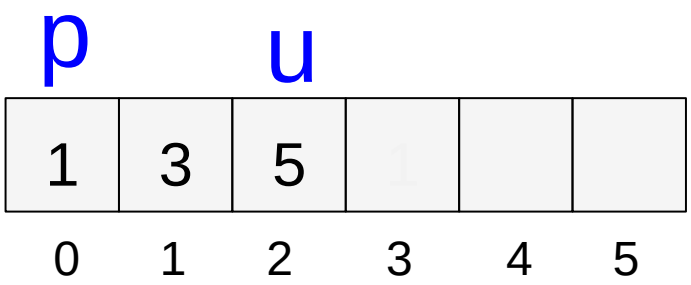


Algoritmo em C

```
//Inserir(5)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

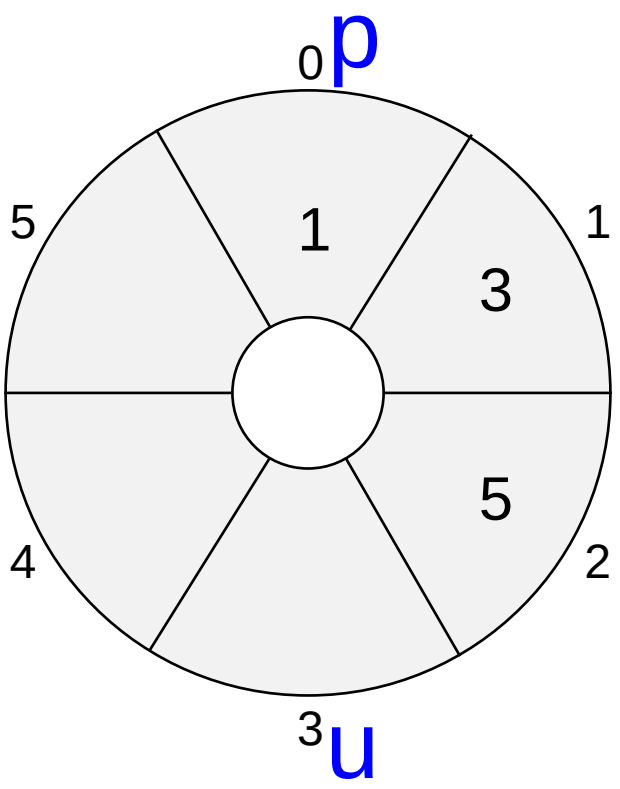


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

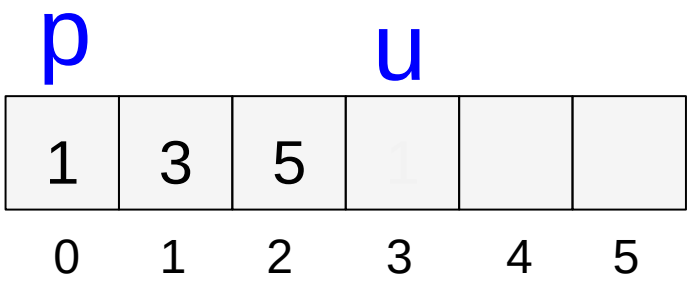


Algoritmo em C

```
//Inserir(5)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

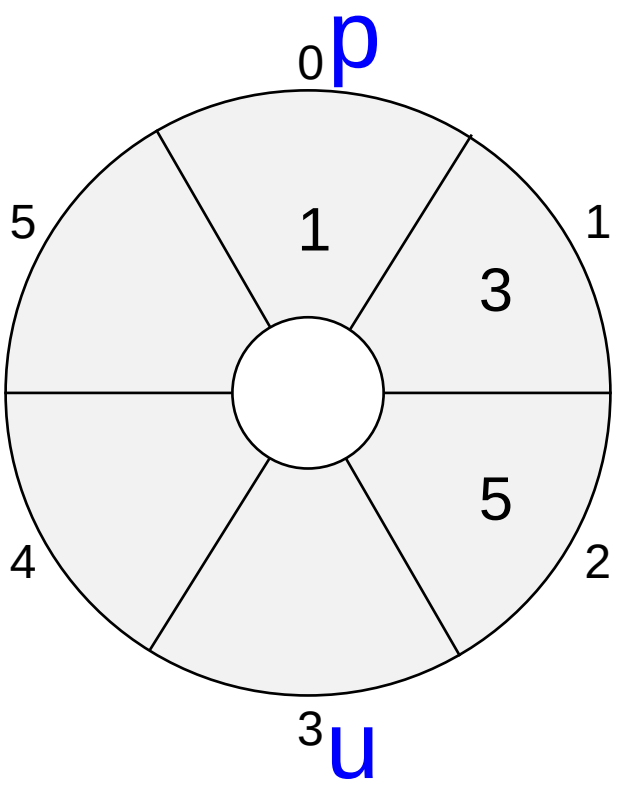


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

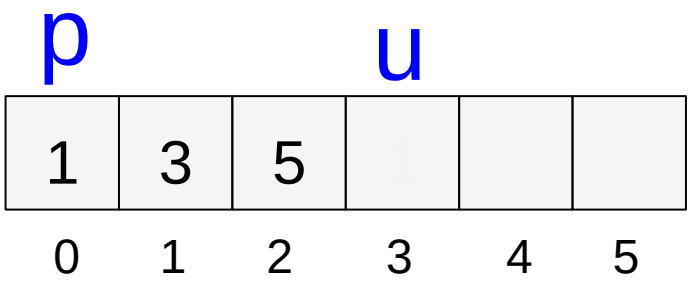


Algoritmo em C

```
//Inserir(5)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

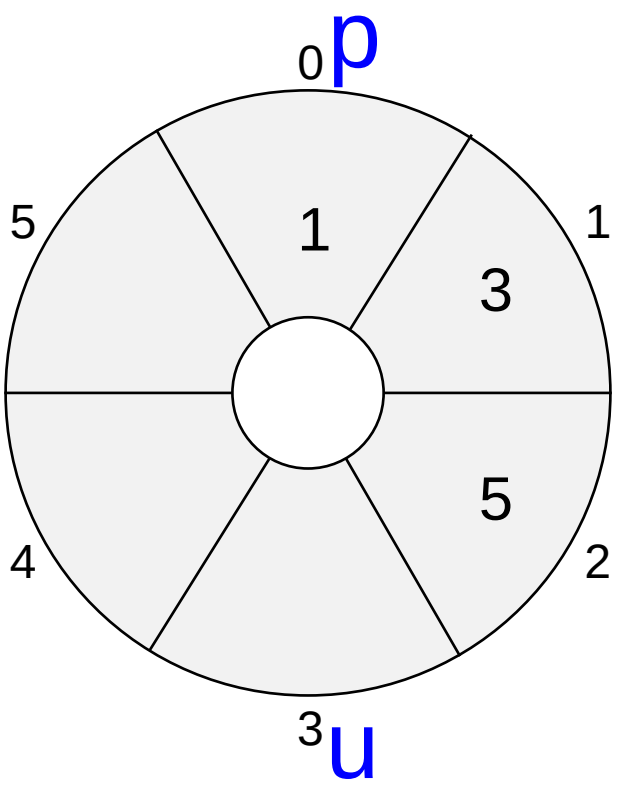


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

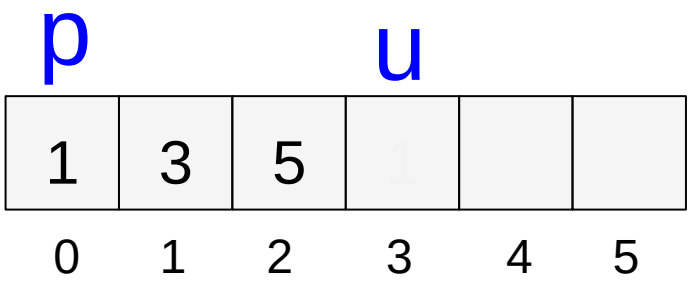


Algoritmo em C

```
//Inserir(7)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```



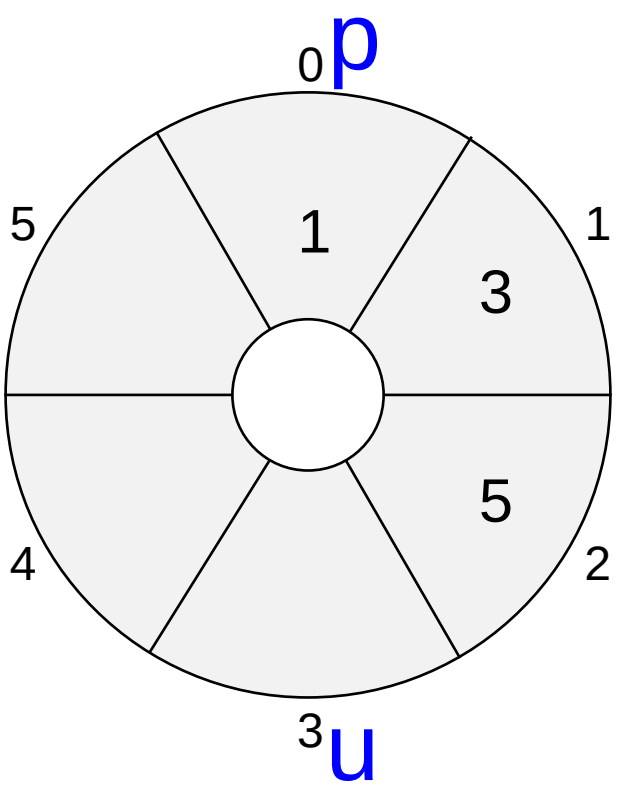
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



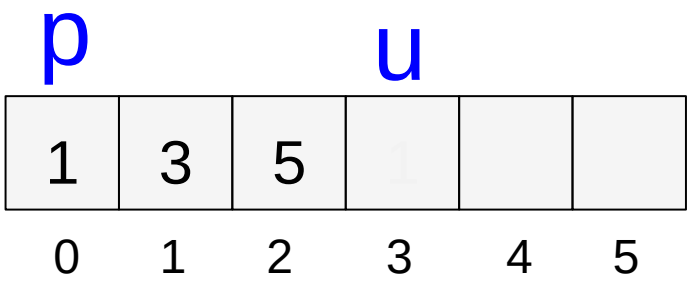
Algoritmo em C

```
//Inserir(7)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
        exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

false: 3 + 1 % 6 == 0

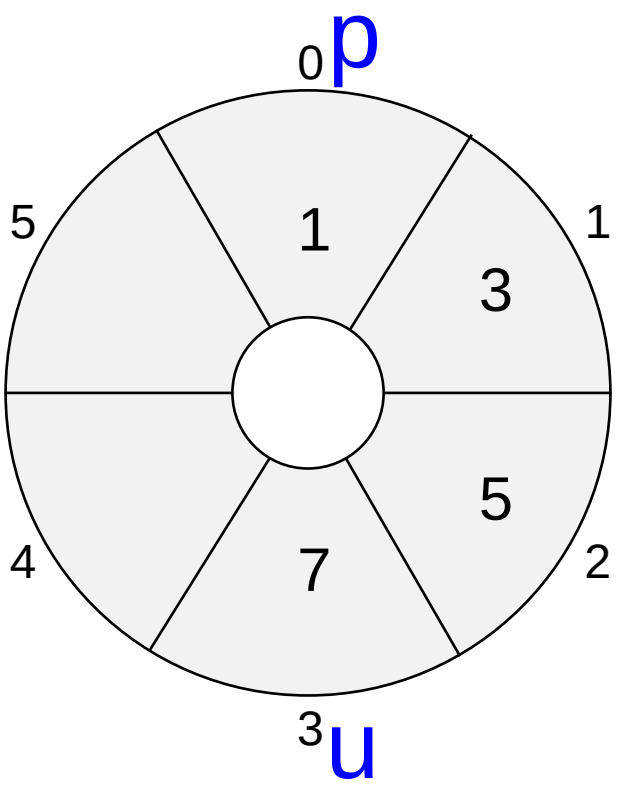


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

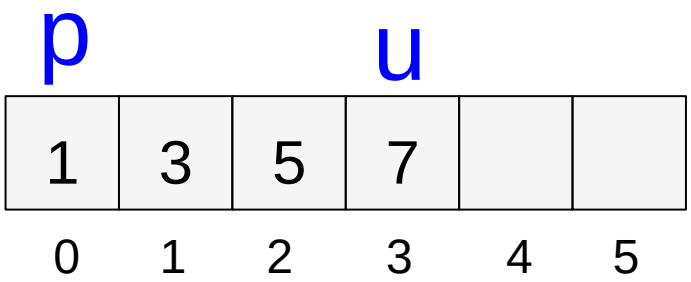


Algoritmo em C

```
//Inserir(7)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

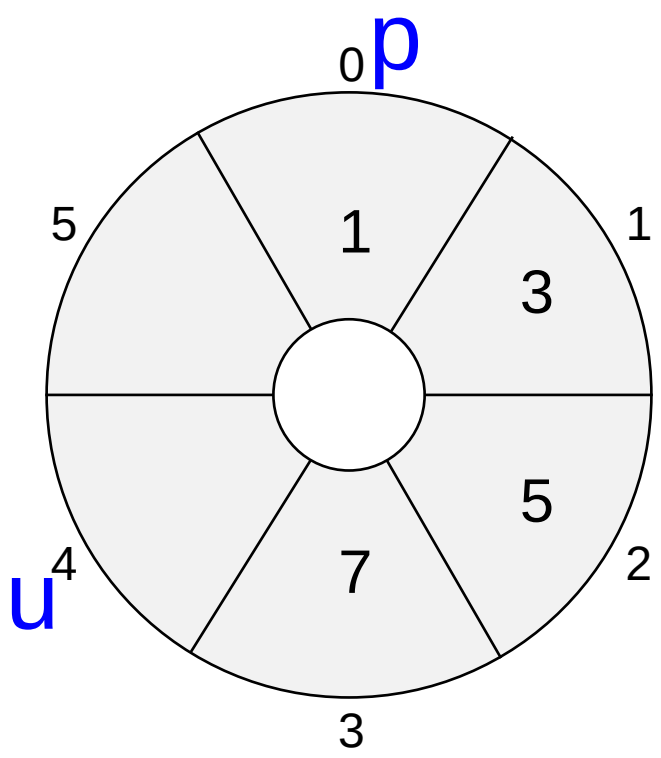


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

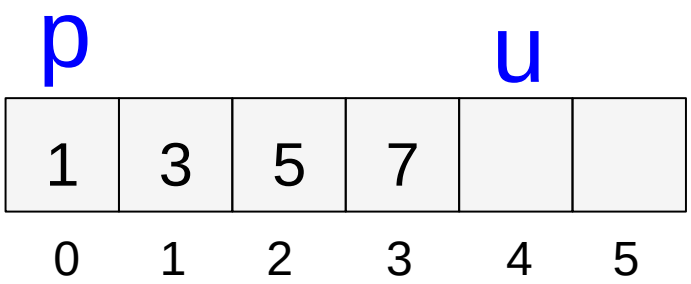


Algoritmo em C

```
//Inserir(7)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

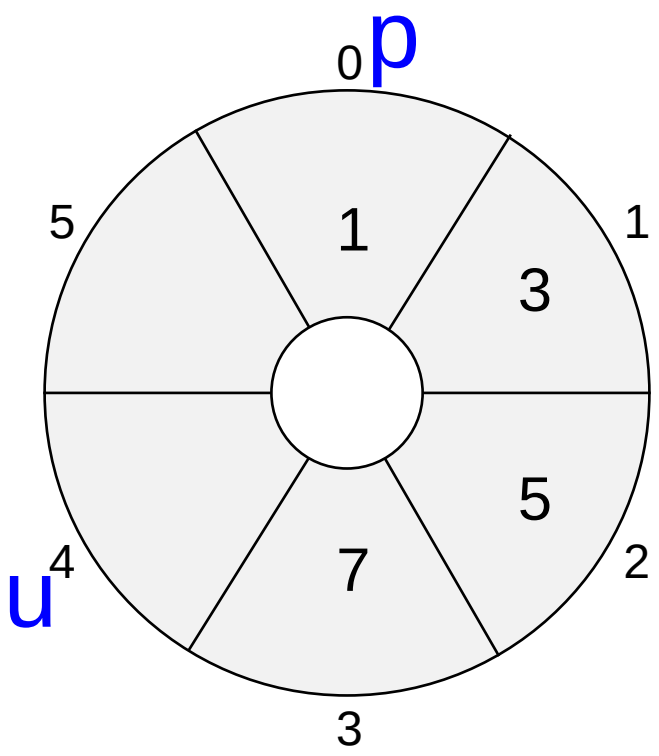


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

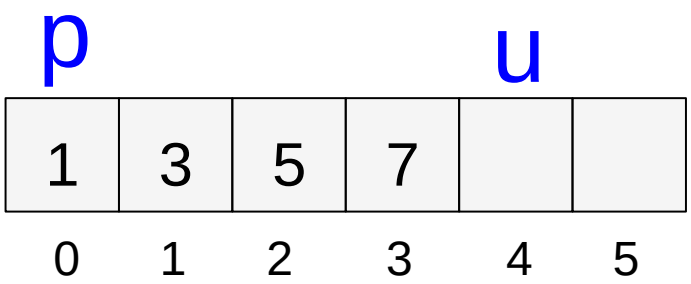


Algoritmo em C

```
//Inserir(7)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

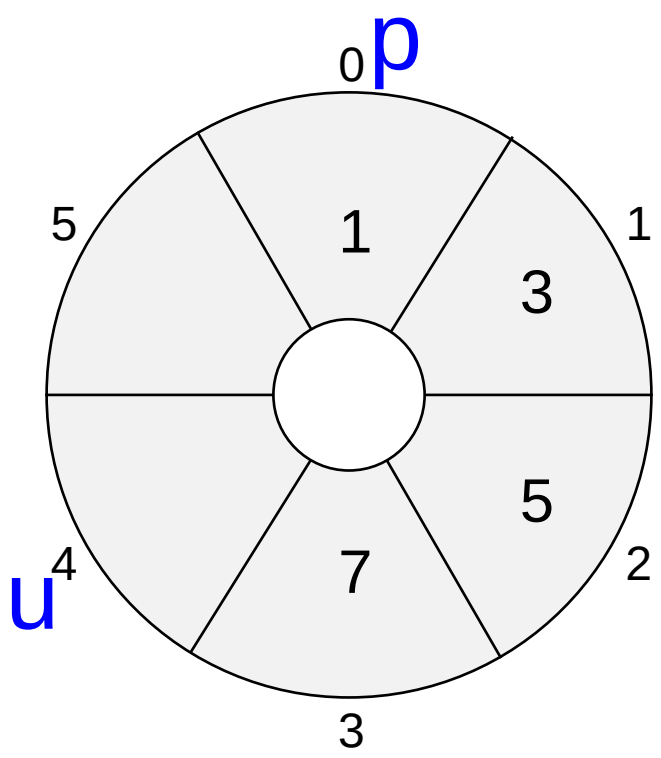


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

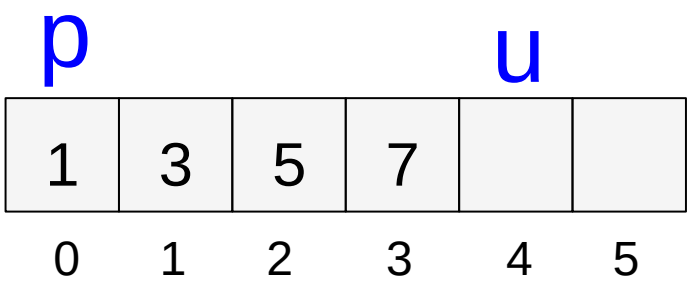


Algoritmo em C

```
//Inserir(9)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```



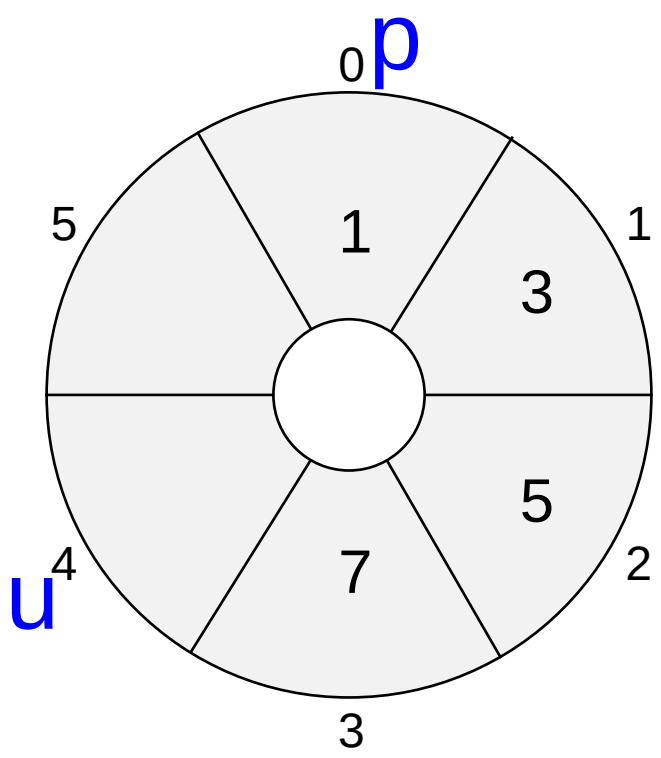
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



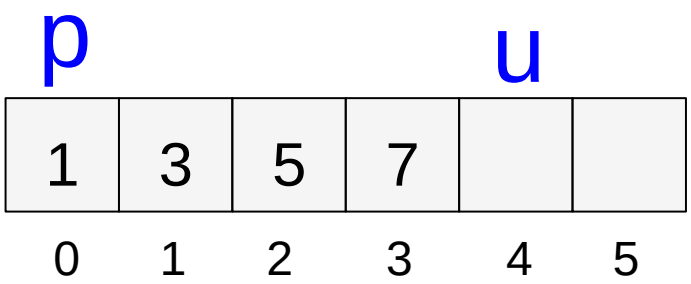
Algoritmo em C

```
//Inserir(9)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
        exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

false: 4 + 1 % 6 == 0

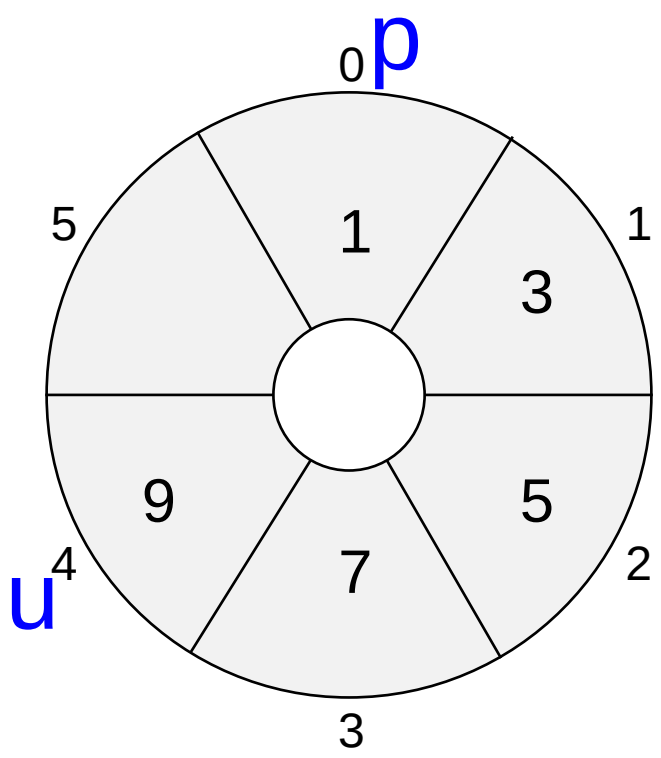


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

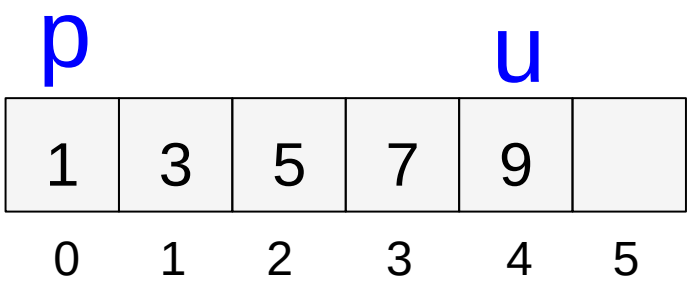


Algoritmo em C

```
//Inserir(9)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

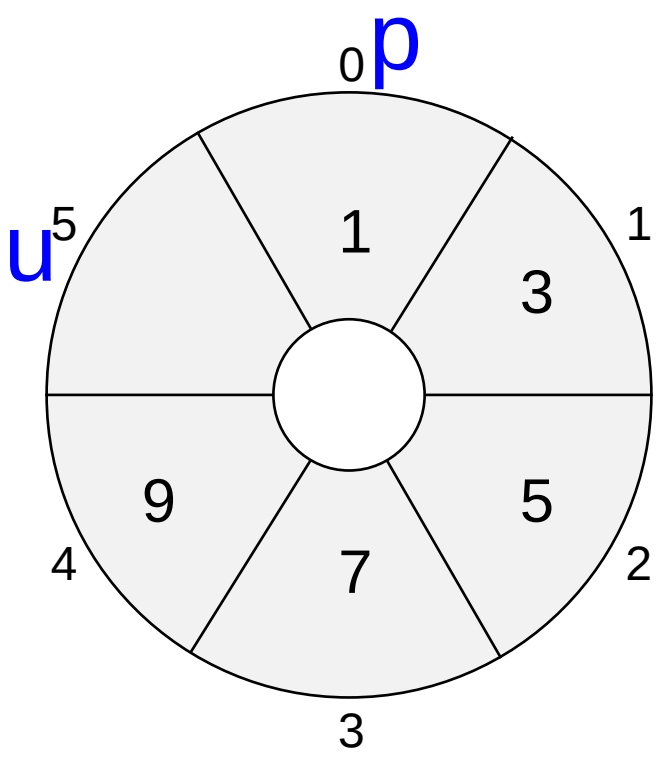


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

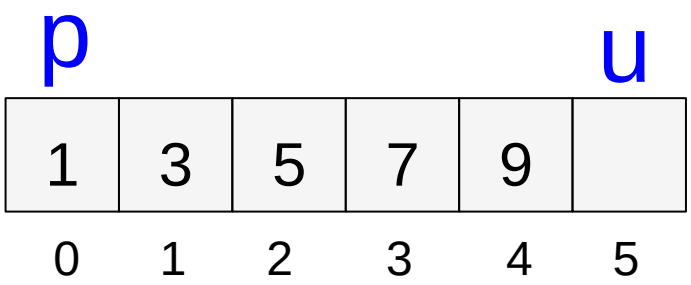


Algoritmo em C

```
//Inserir(9)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

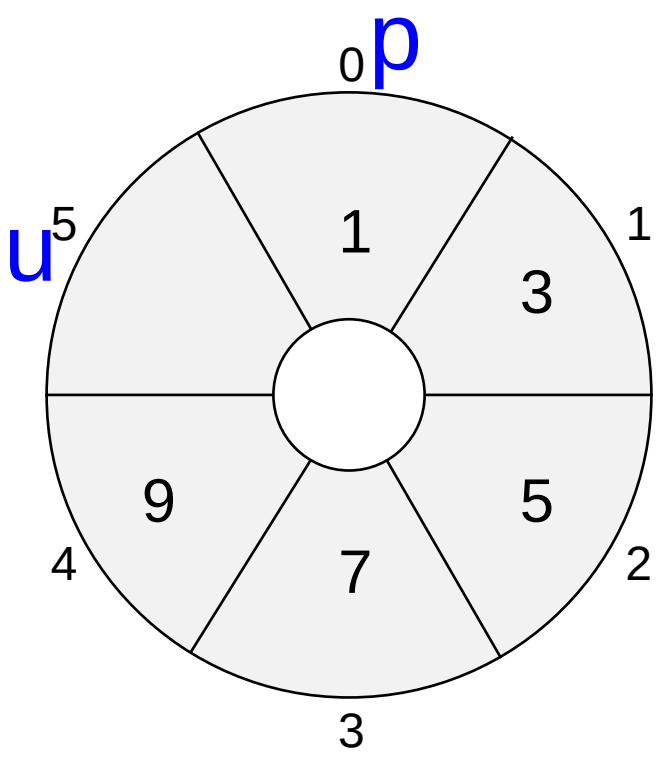


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

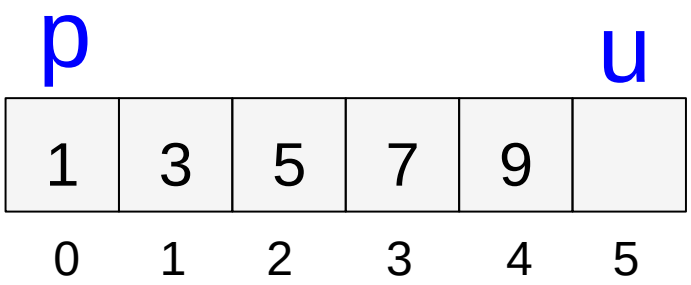


Algoritmo em C

```
//Inserir(9)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

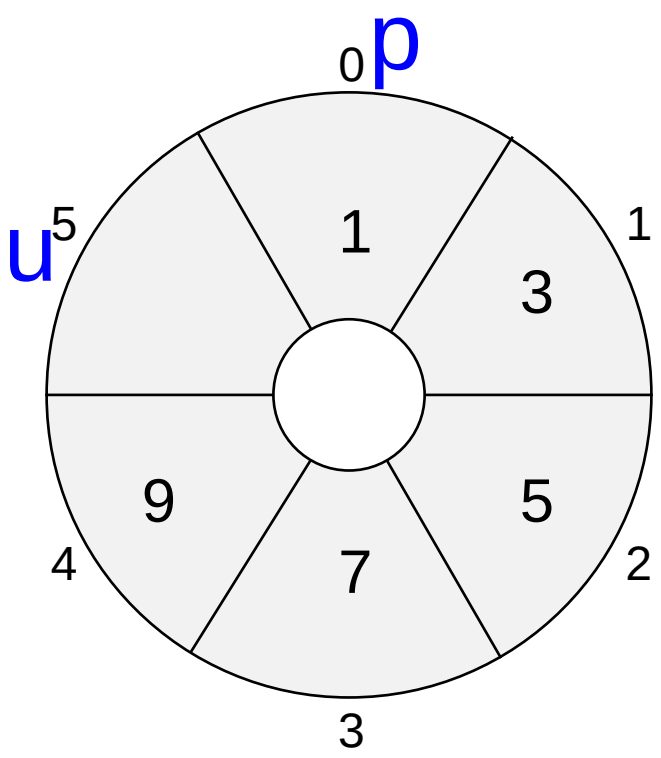


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

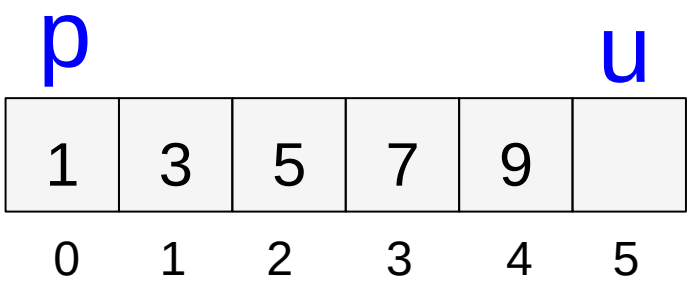


Algoritmo em C

```
//Inserir(2)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```



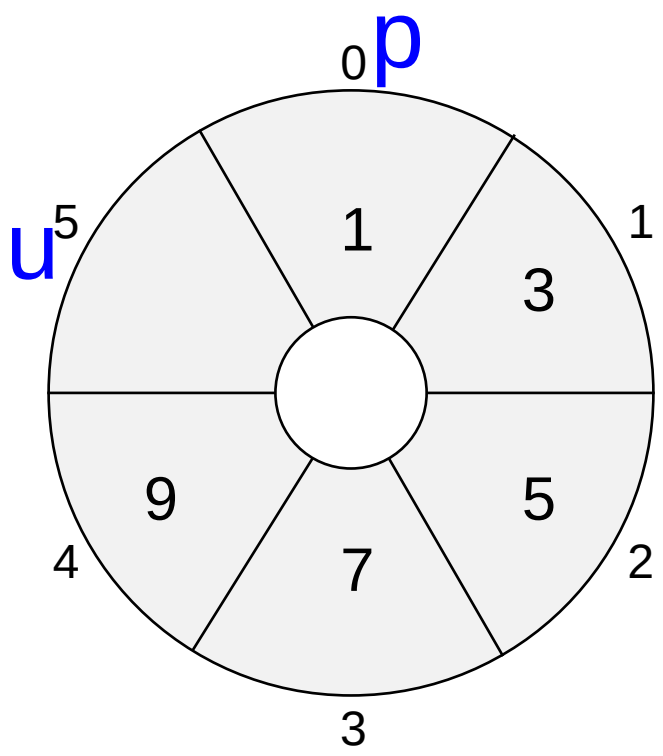
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



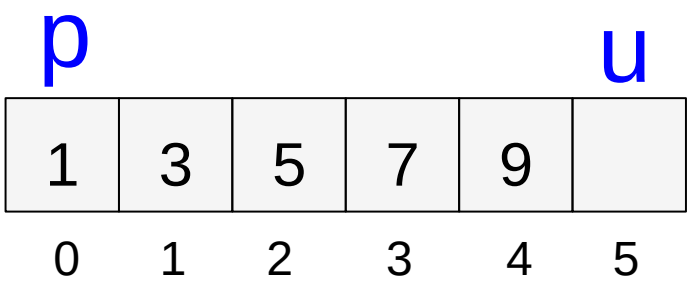
Algoritmo em C

```
//Inserir(2)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
        exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

true: 5 + 1 % 6 == 0



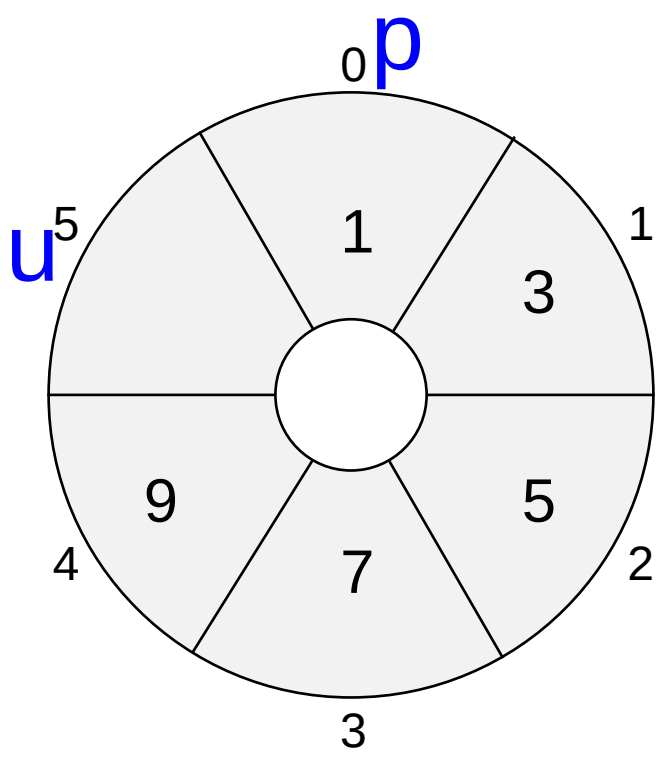
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em C

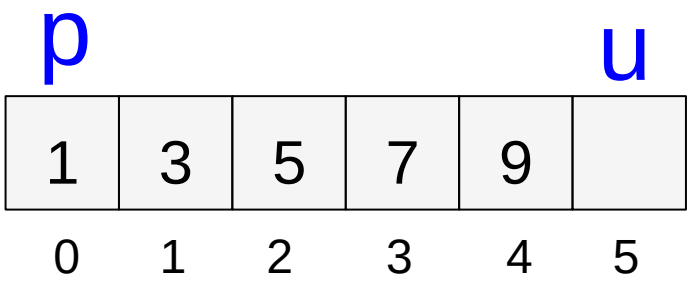
```
//Inserir(2)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
        exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

true: 5 + 1 % 6 == 0



Como nossa fila tem tamanho cinco, não conseguimos alocar mais elementos

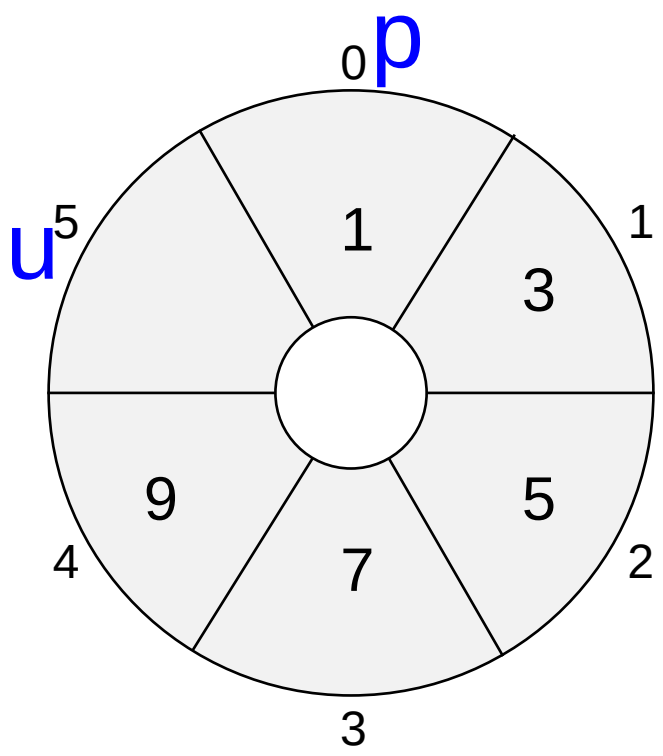
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em C

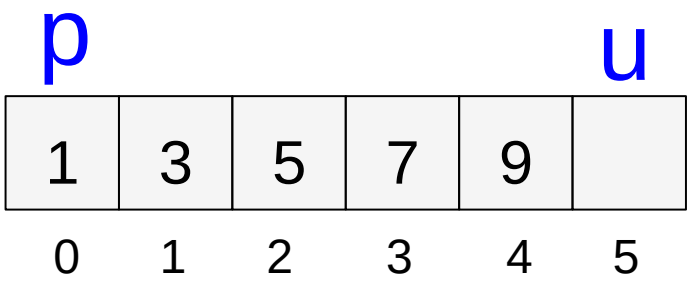
```
//Inserir(2)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
        exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

true: 5 + 1 % 6 == 0



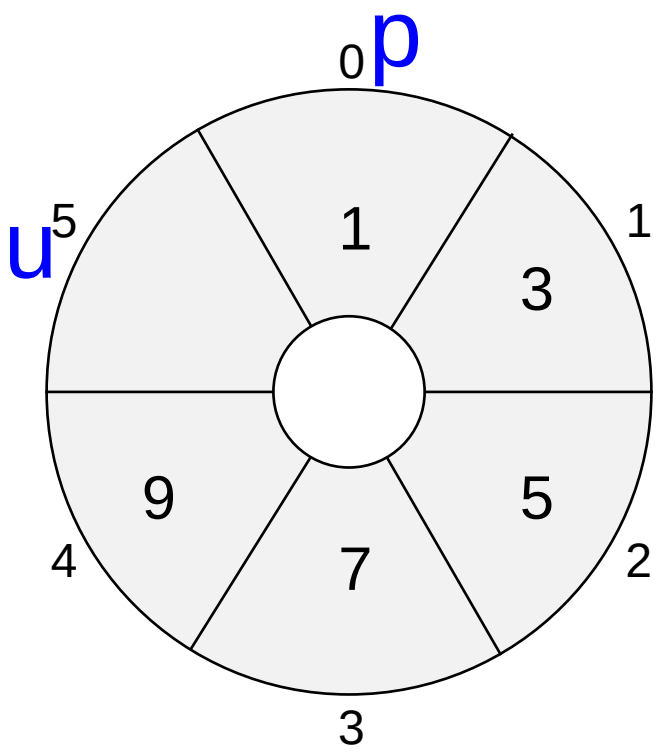
Vamos para a próxima operação, um remover

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

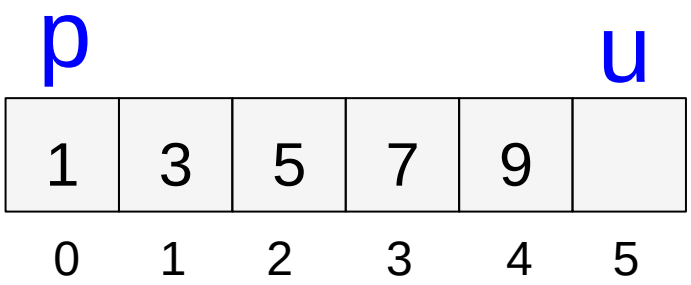


Algoritmo em C

```
//Remover()

int remover() {

   if (primeiro == ultimo)
      exit(1);

   int resp = array[primeiro];
   primeiro = (primeiro + 1) % MAXTAM;
   return resp;
}
```



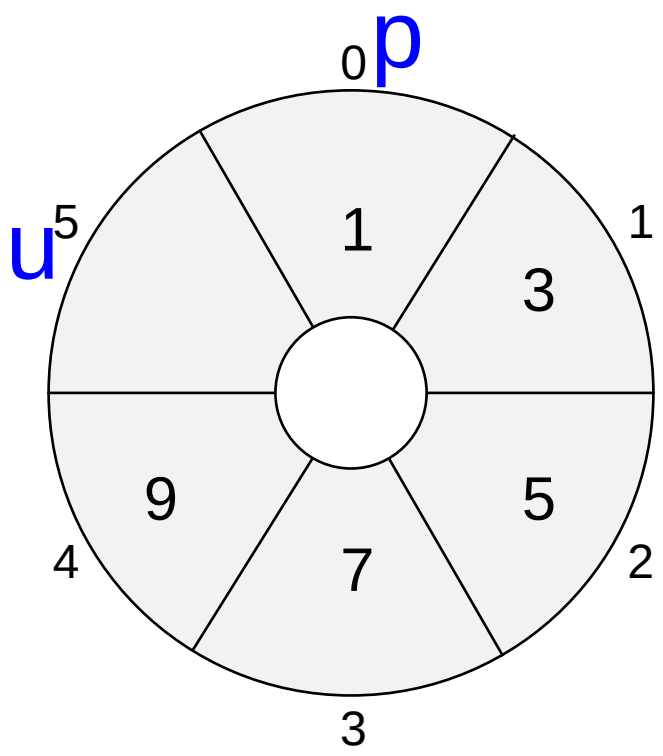
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



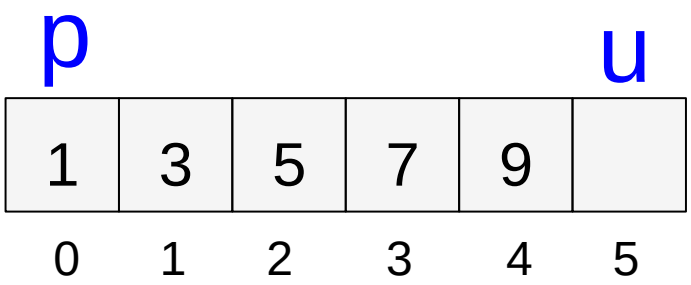
Algoritmo em C

```
//Remover()

 int remover() {

    if (primeiro == ultimo)
       exit(1);

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % MAXTAM;
    return resp;
 }
```

false: 0 == 5

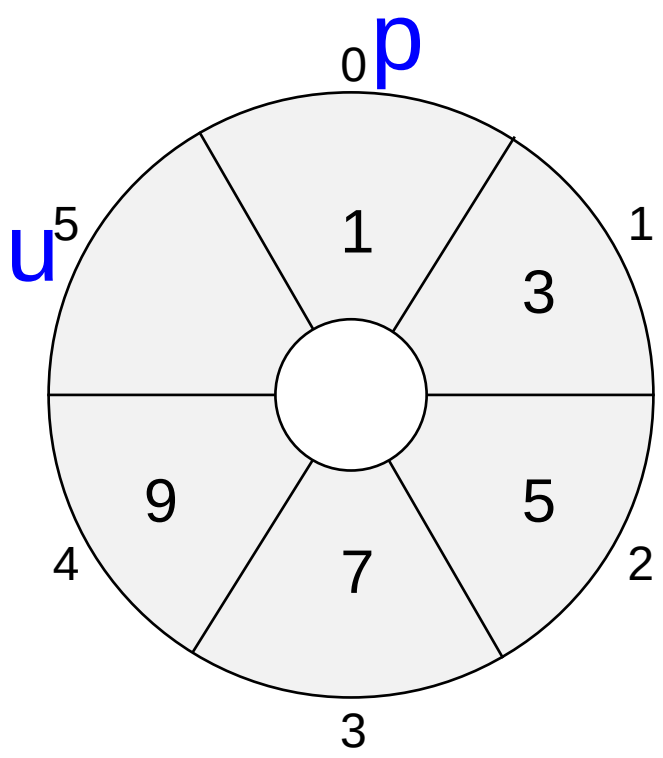


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

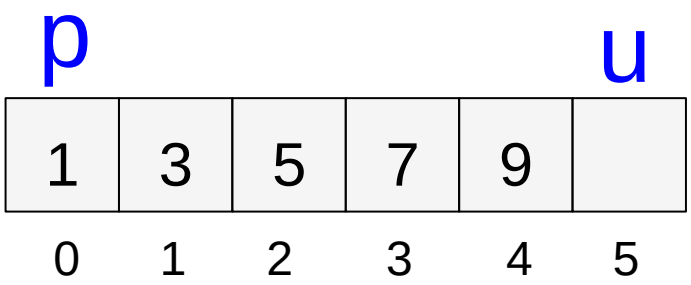


Algoritmo em C

```
//Remover()                                          resp | 1 |

 int remover() {

    if (primeiro == ultimo)
       exit(1);

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % MAXTAM;
    return resp;
 }
```



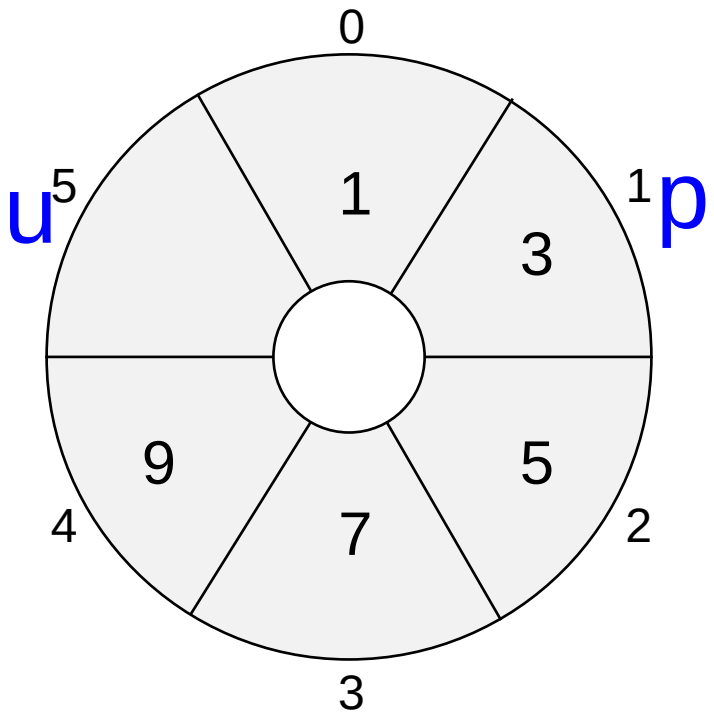
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



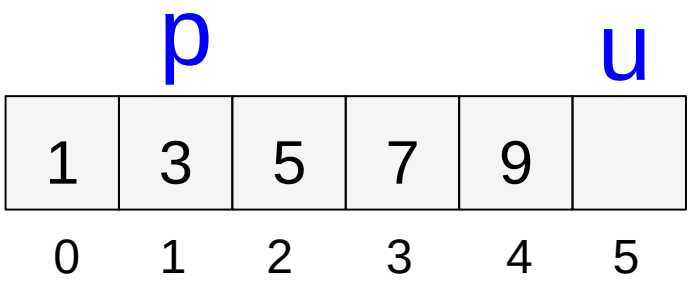
Algoritmo em C

```
//Remover()

 int remover() {

    if (primeiro == ultimo)
       exit(1);

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % MAXTAM;
    return resp;
 }
```

resp | 1

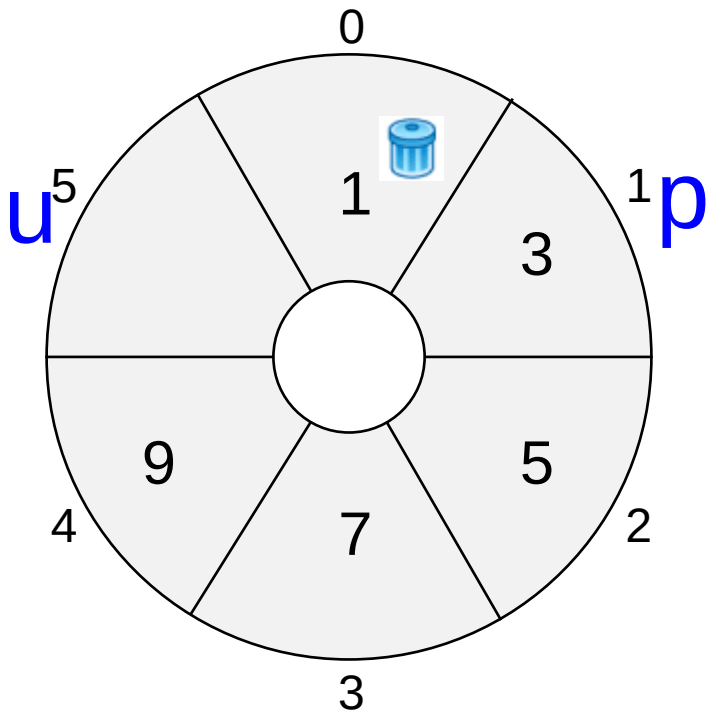


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

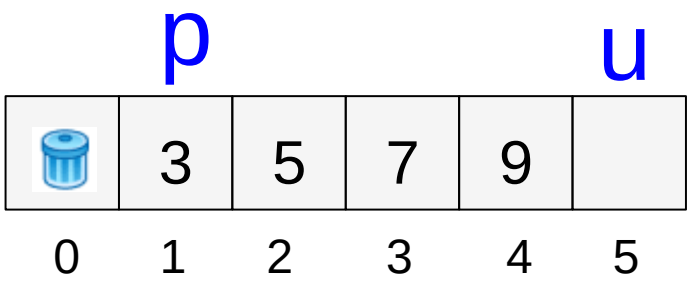


Algoritmo em C

```
//Remover()                                          resp | 1 |

 int remover() {

    if (primeiro == ultimo)
       exit(1);

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % MAXTAM;
    return resp;
 }
```

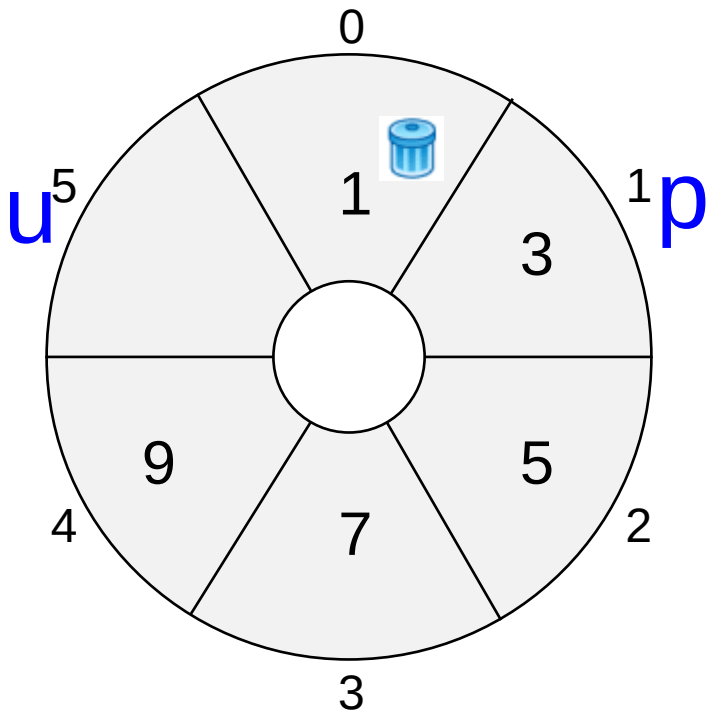


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

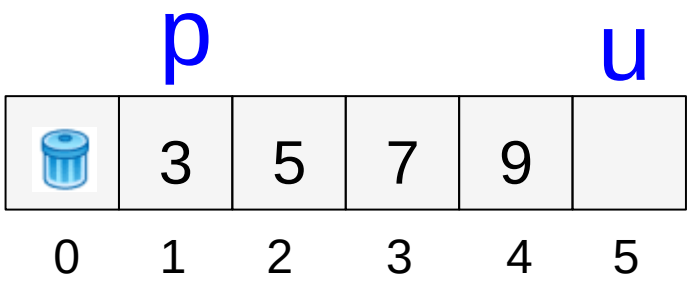


Algoritmo em C

```
//Remover()

 int remover() {

    if (primeiro == ultimo)
       exit(1);

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % MAXTAM;
    return resp;
 }
```

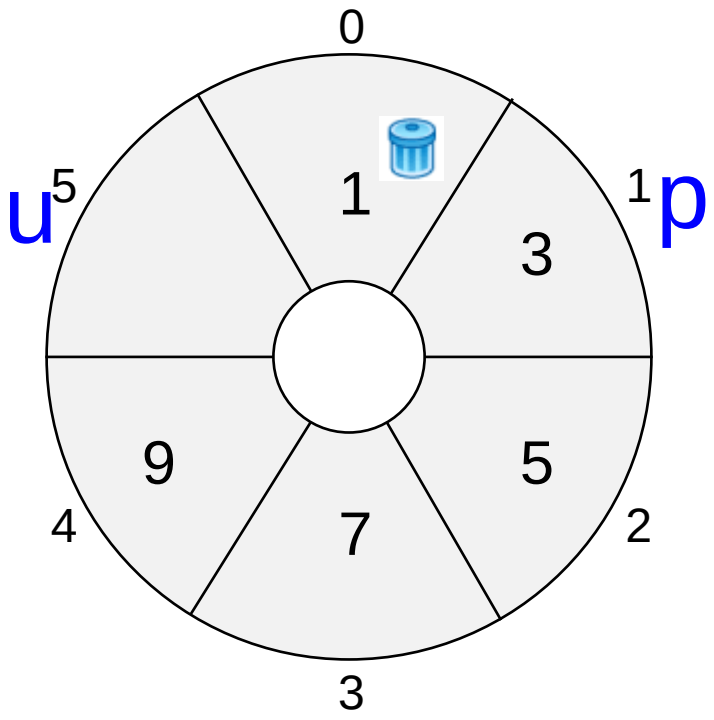


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

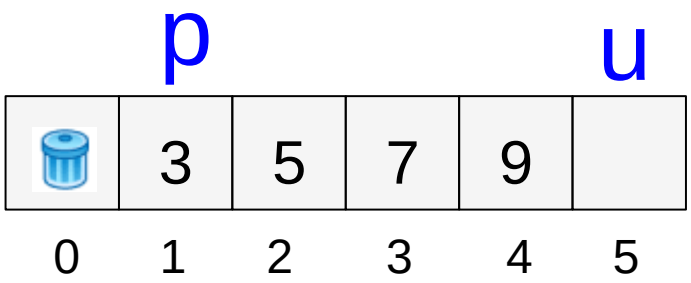


Algoritmo em C

```
//Remover()

int remover() {

   if (primeiro == ultimo)
      exit(1);

   int resp = array[primeiro];
   primeiro = (primeiro + 1) % MAXTAM;
   return resp;
}
```



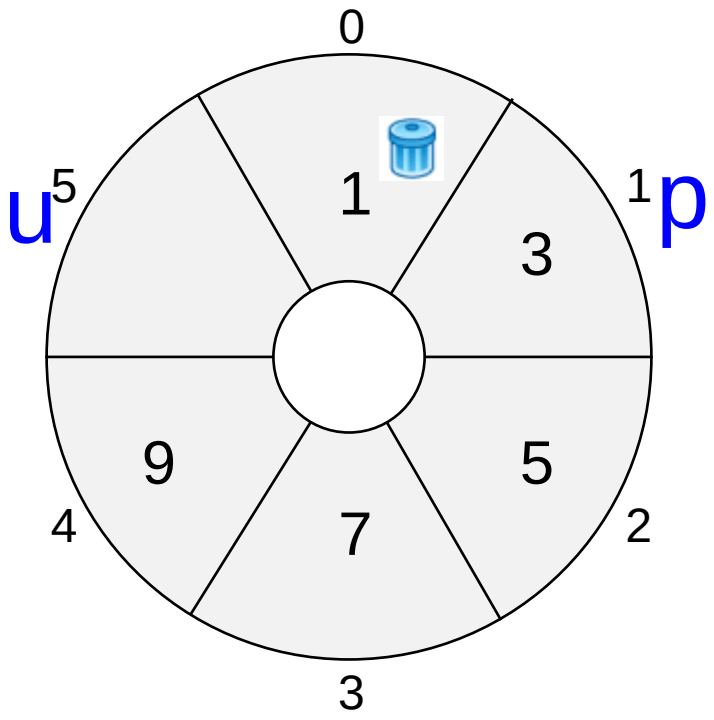
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



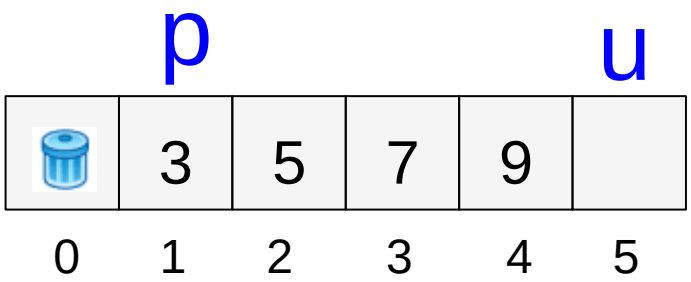
Algoritmo em C

```
//Remover()

int remover() {

    if (primeiro == ultimo)
        exit(1);

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % MAXTAM;
    return resp;
}
```

false: 1 == 5

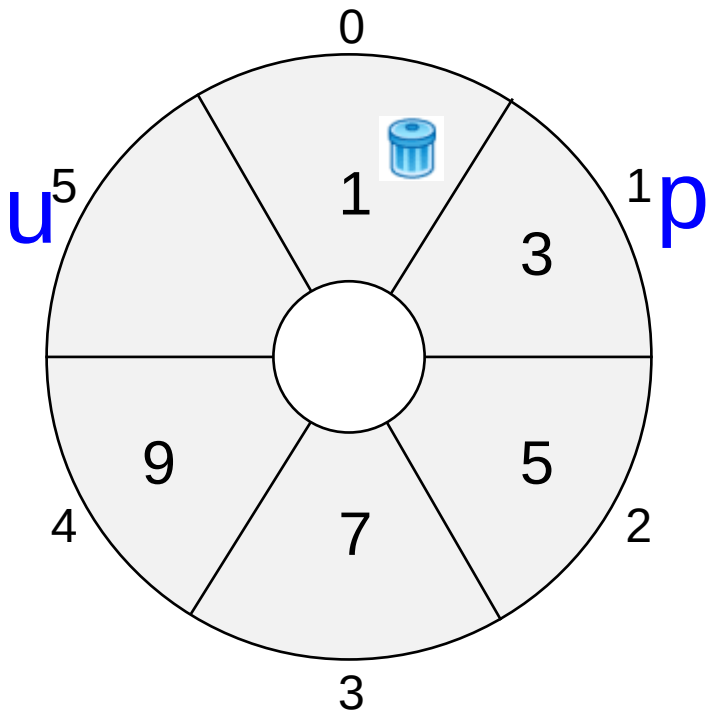


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

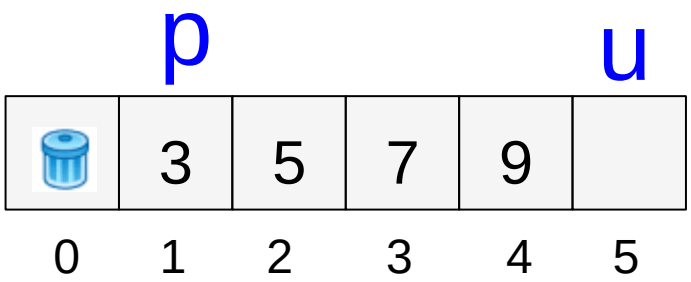


Algoritmo em C

```
//Remover()                                          resp | 3 |

 int remover() {

    if (primeiro == ultimo)
       exit(1);

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % MAXTAM;
    return resp;
 }
```



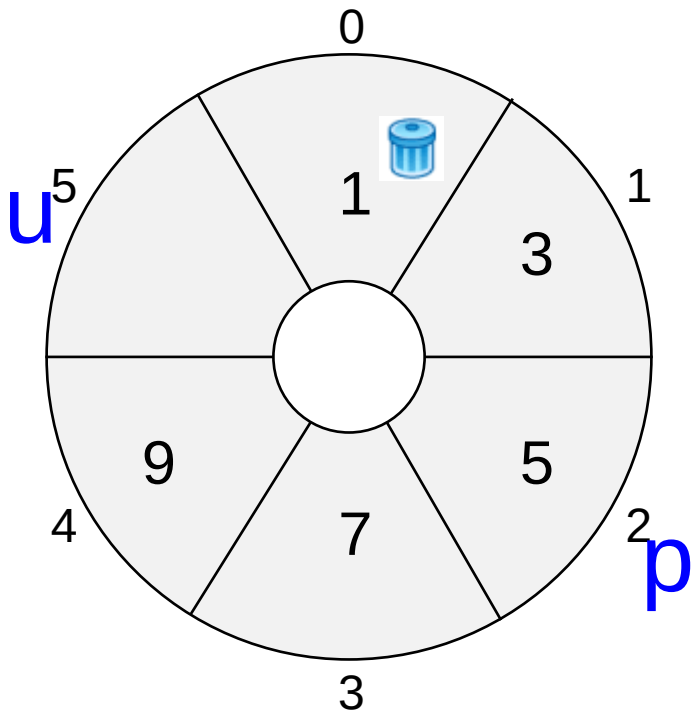
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



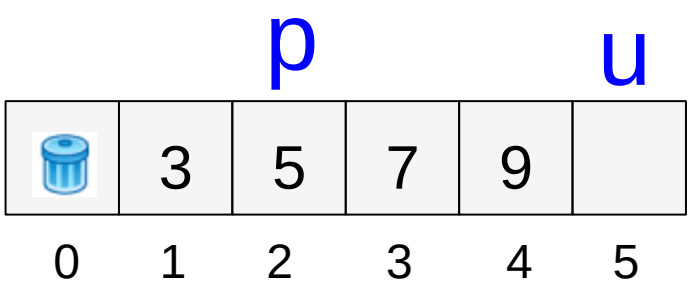
Algoritmo em C

```
//Remover()

 int remover() {

    if (primeiro == ultimo)
       exit(1);

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % MAXTAM;
    return resp;
 }
```

resp | 3

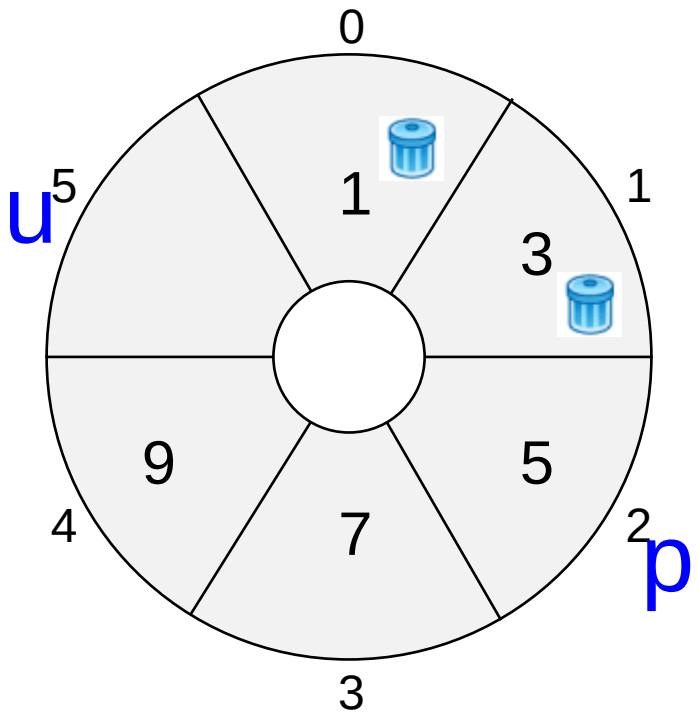


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

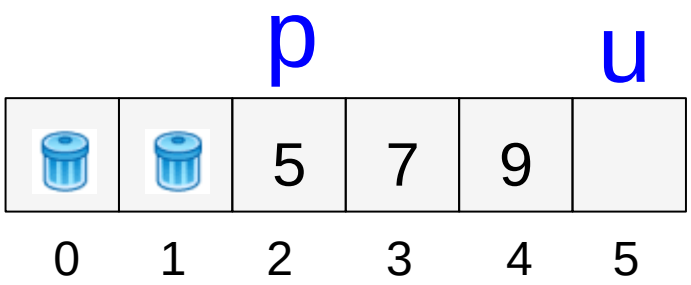


Algoritmo em C

```
//Remover()                                          resp | 3 |

 int remover() {

    if (primeiro == ultimo)
       exit(1);

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % MAXTAM;
    return resp;
 }
```

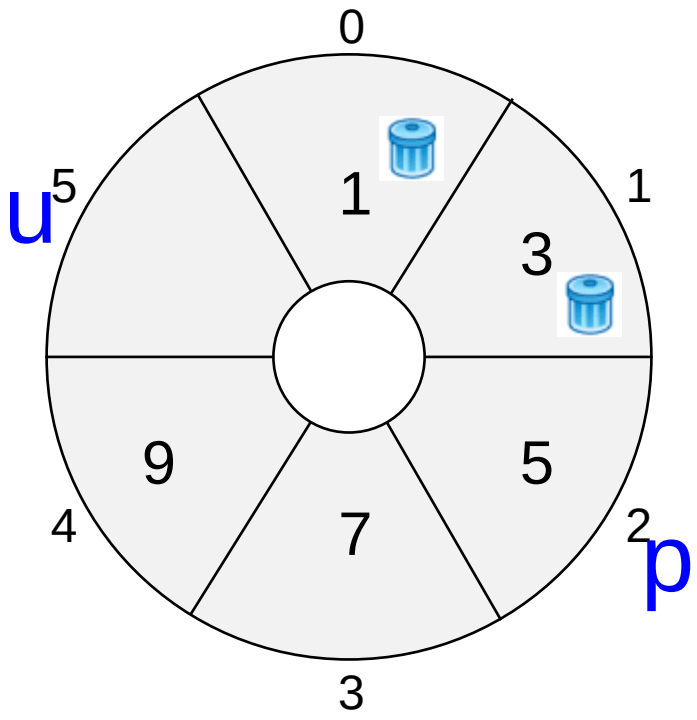


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

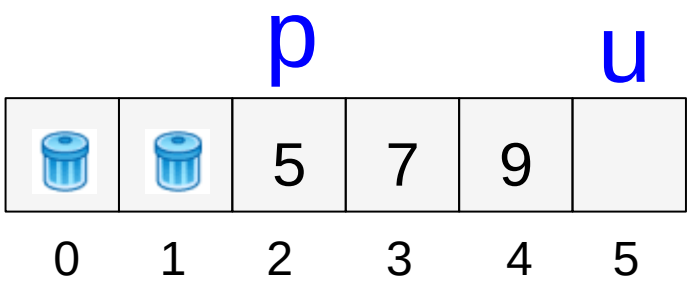


Algoritmo em C

```
//Remover()

 int remover() {

    if (primeiro == ultimo)
       exit(1);

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % MAXTAM;
    return resp;
 }
```

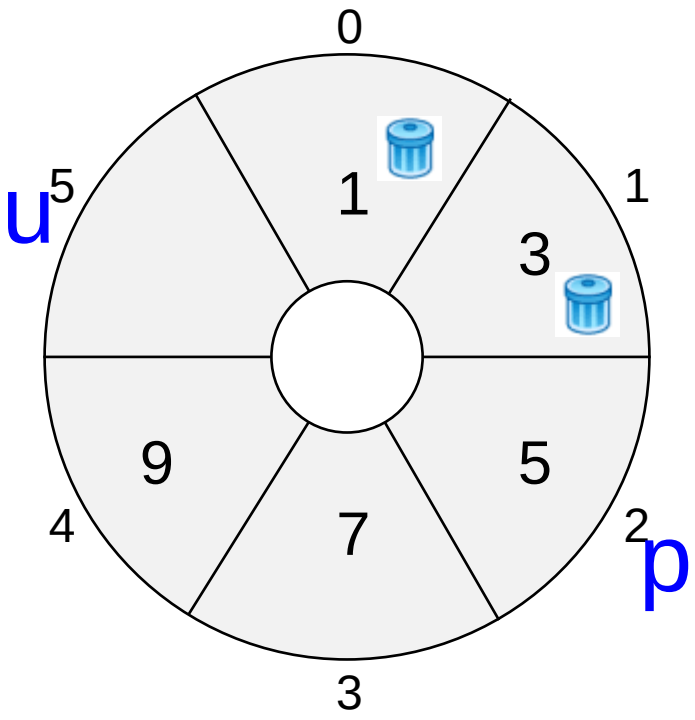


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

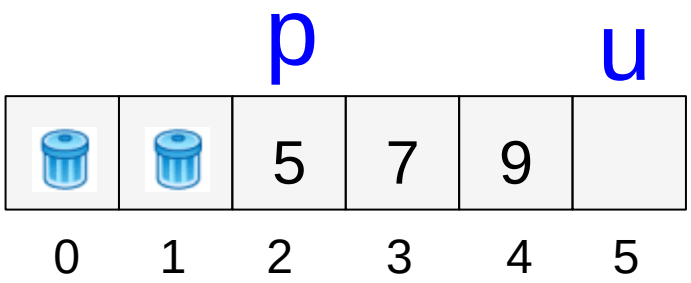


Algoritmo em C

```
//Inserir(4)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```



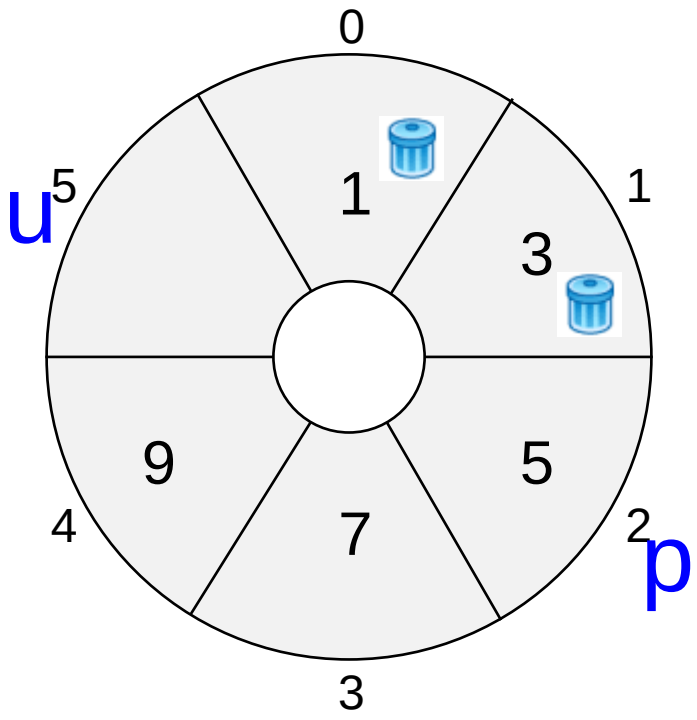
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



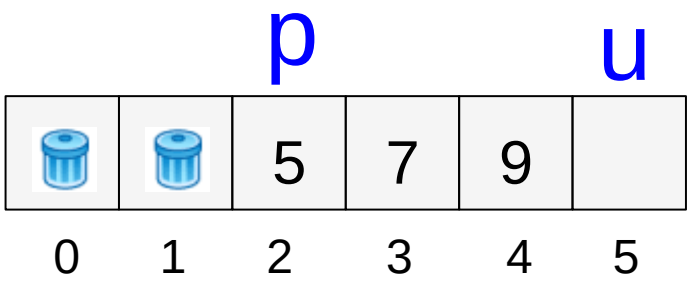
Algoritmo em C

```
//Inserir(4)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
        exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

false: 5 + 1 % 6 == 2

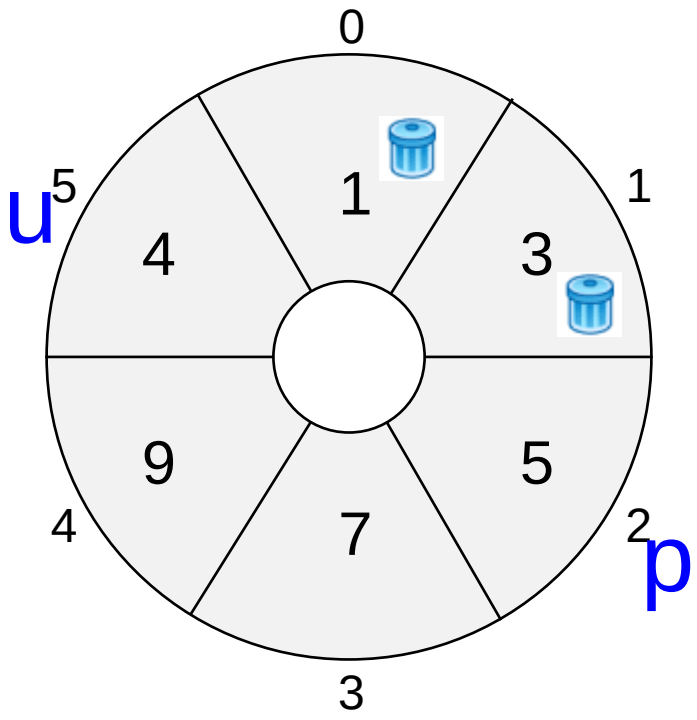


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

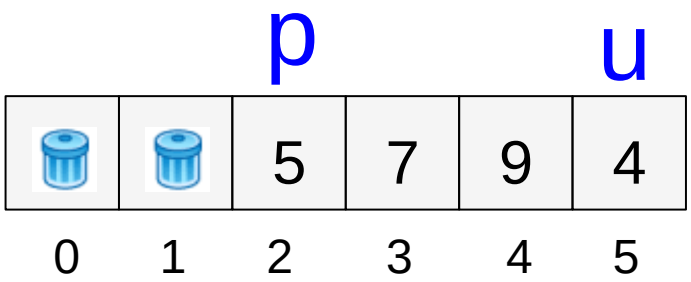


Algoritmo em C

```
//Inserir(4)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

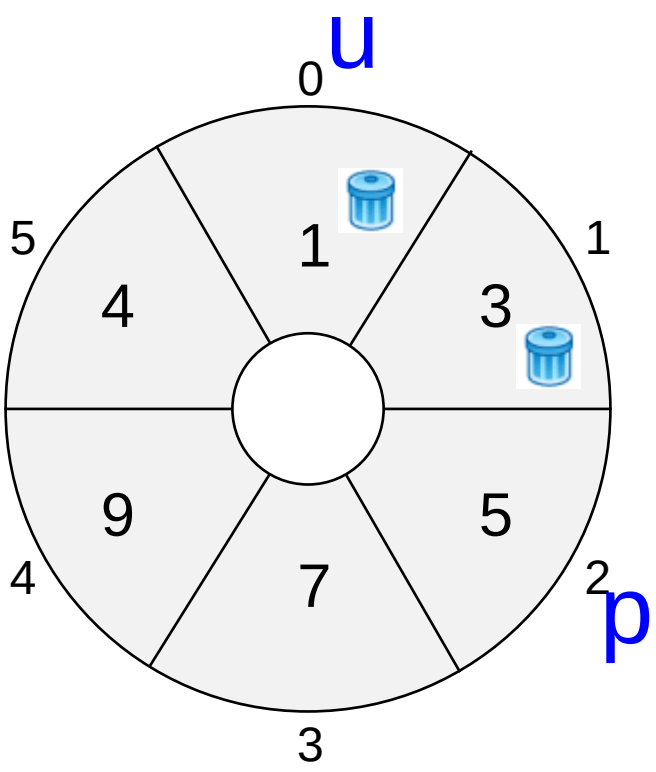


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

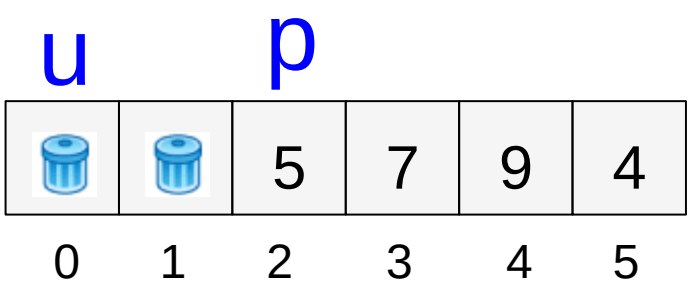


Algoritmo em C

```
//Inserir(4)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

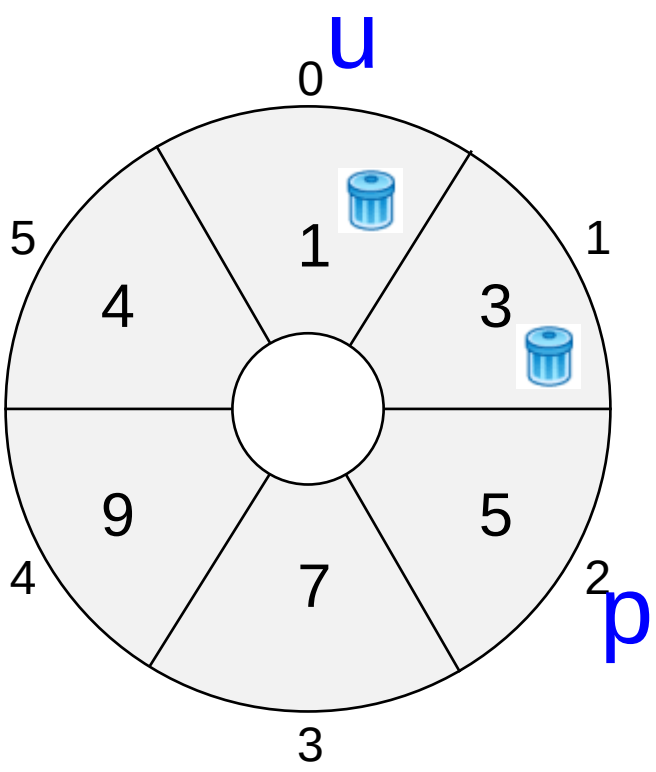


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

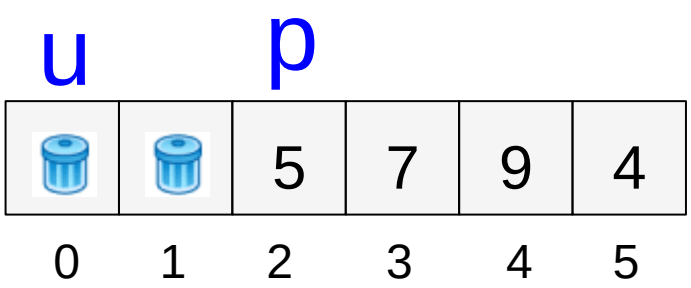


Algoritmo em C

```
//Inserir(4)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

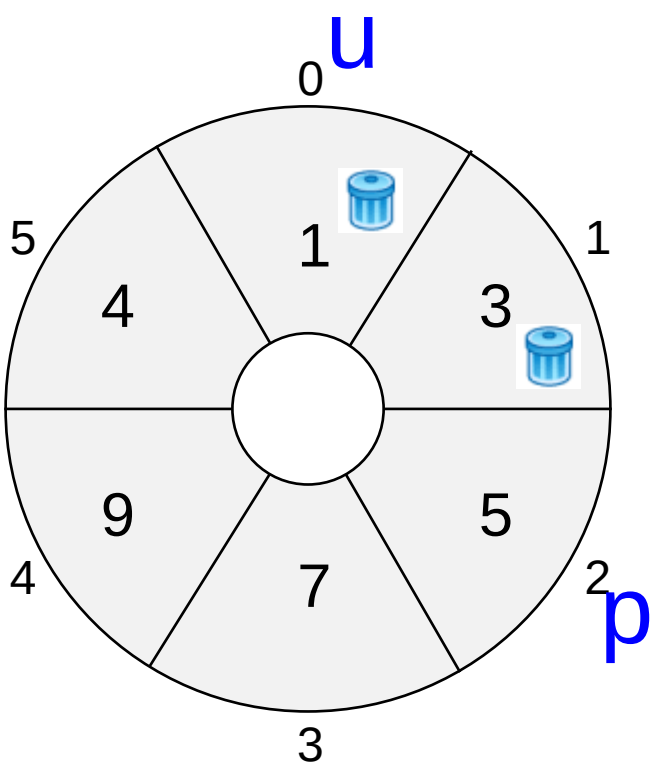


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

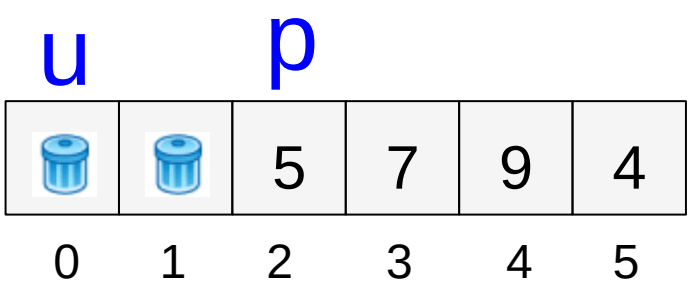


Algoritmo em C

```
//Inserir(6)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```



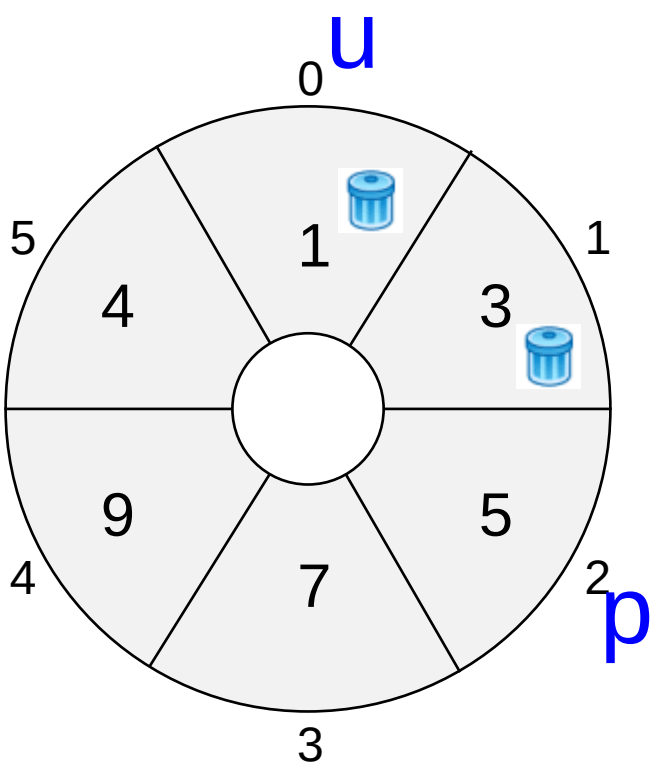
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



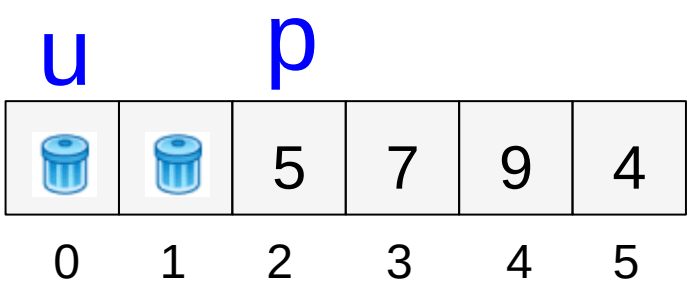
Algoritmo em C

```
//Inserir(6)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
        exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

false: 0 + 1 % 6 == 2

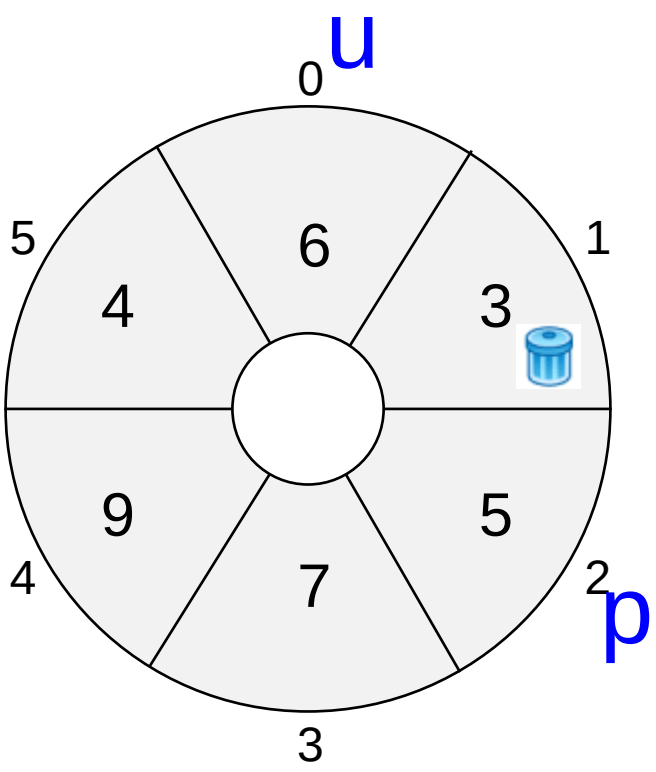


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

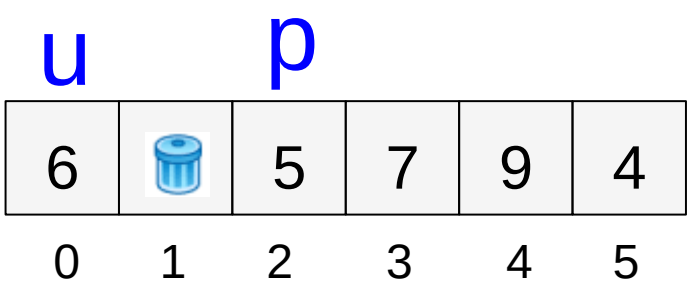


Algoritmo em C

```
//Inserir(6)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

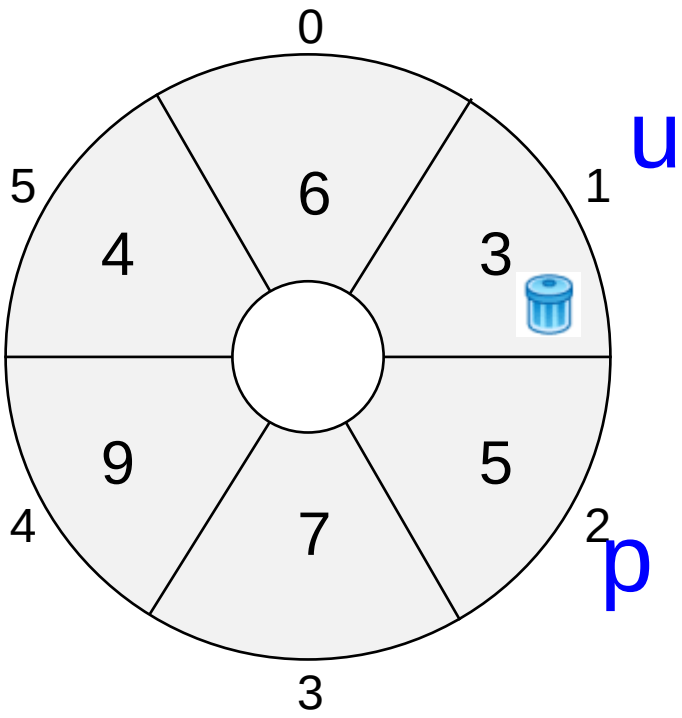


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

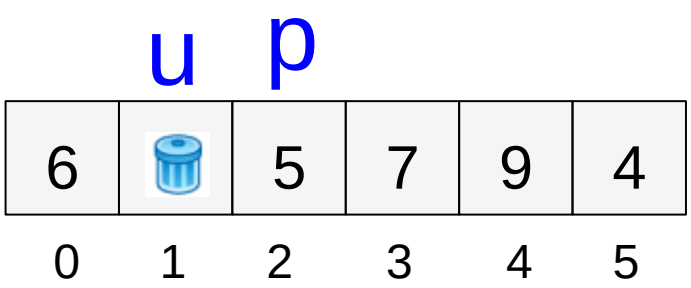


Algoritmo em C

```
//Inserir(6)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

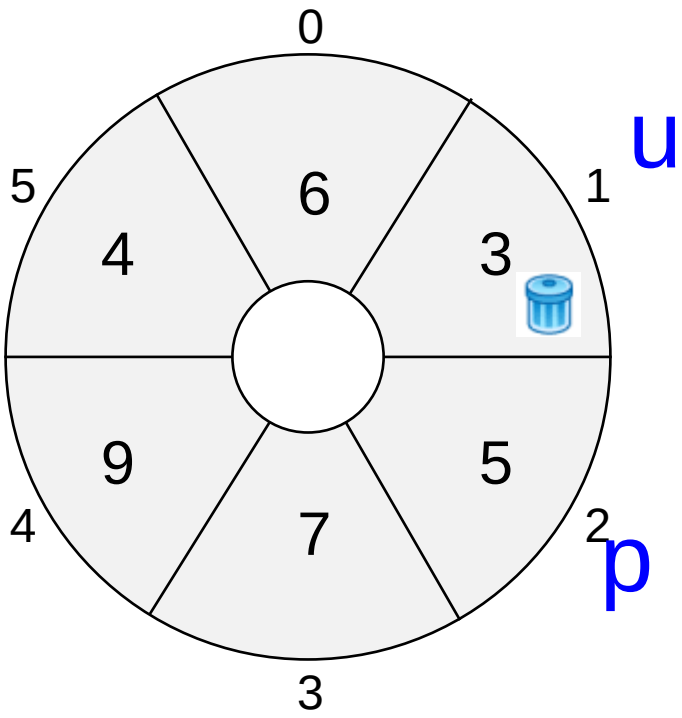


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

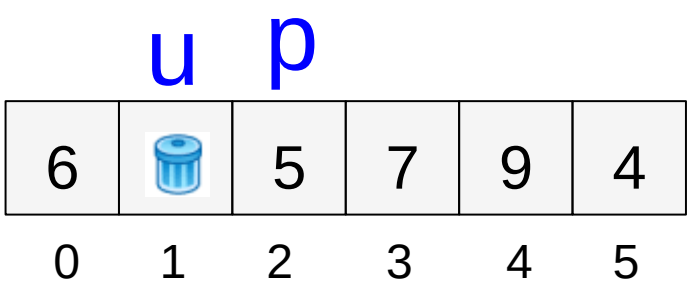


Algoritmo em C

```
//Inserir(6)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

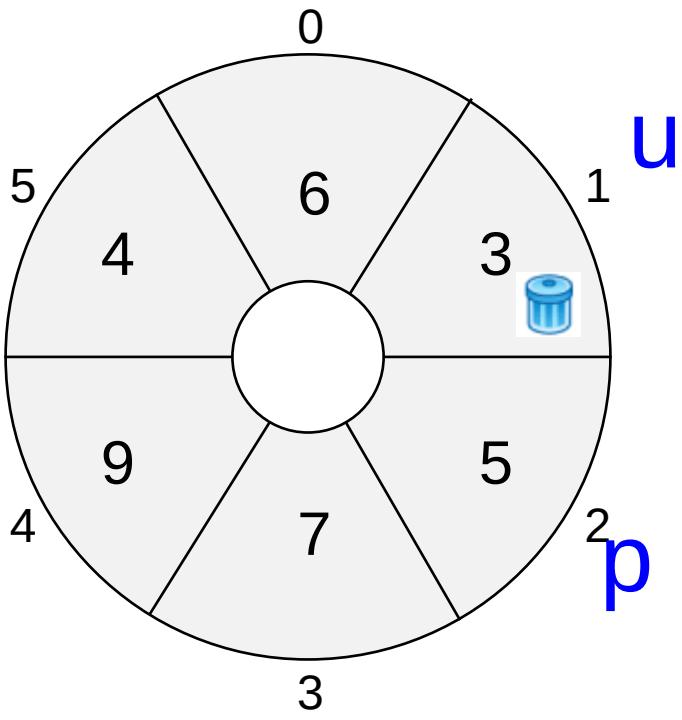


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

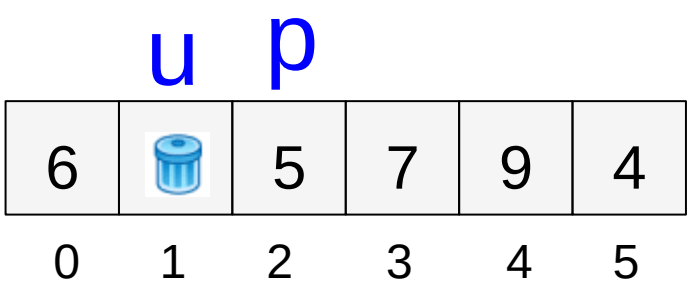


Algoritmo em C

```
//Remover()

int remover() {

   if (primeiro == ultimo)
      exit(1);

   int resp = array[primeiro];
   primeiro = (primeiro + 1) % MAXTAM;
   return resp;
}
```



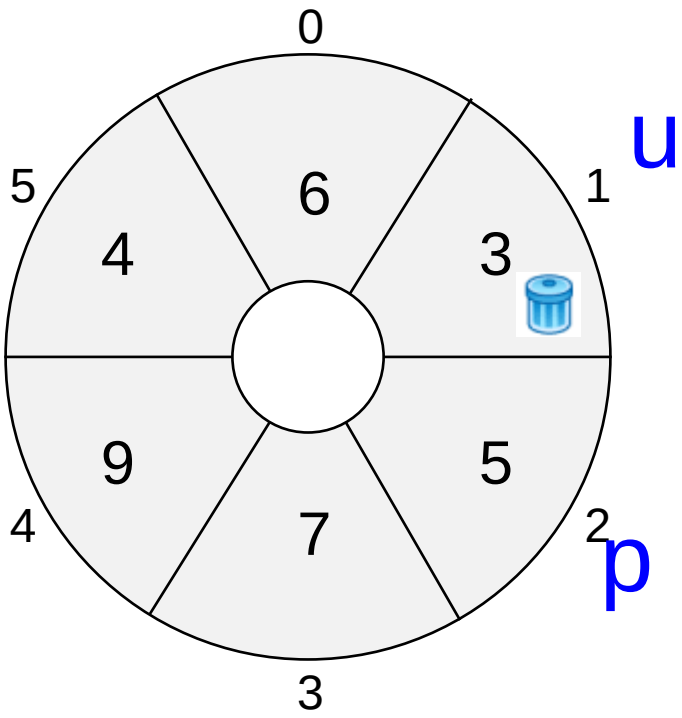
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



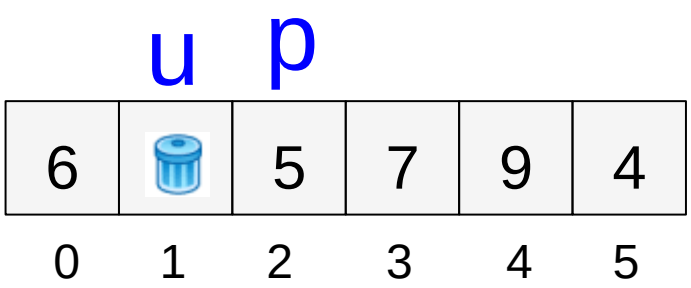
Algoritmo em C

```
//Remover()

int remover() {

    if (primeiro == ultimo)
        exit(1);

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % MAXTAM;
    return resp;
}
```

false: 2 == 1



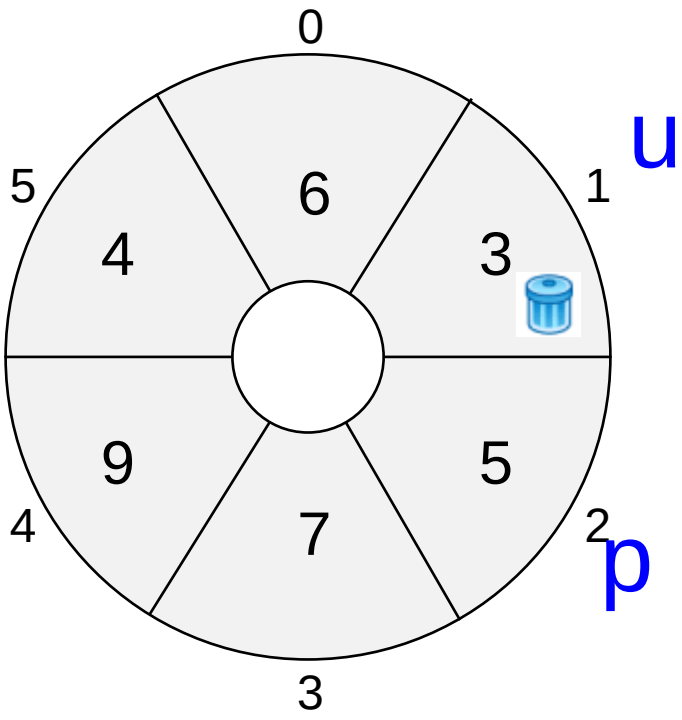
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



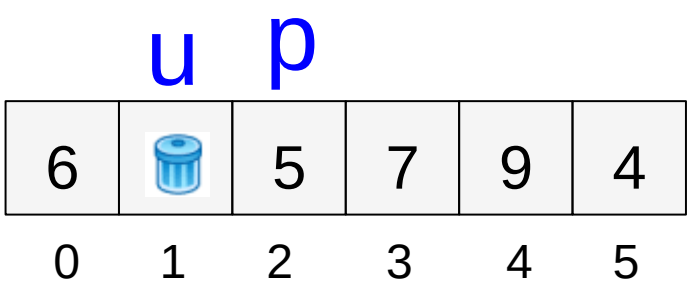
Algoritmo em C

```
//Remover()

 int remover() {

    if (primeiro == ultimo)
       exit(1);

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % MAXTAM;
    return resp;
 }
```

resp | 5



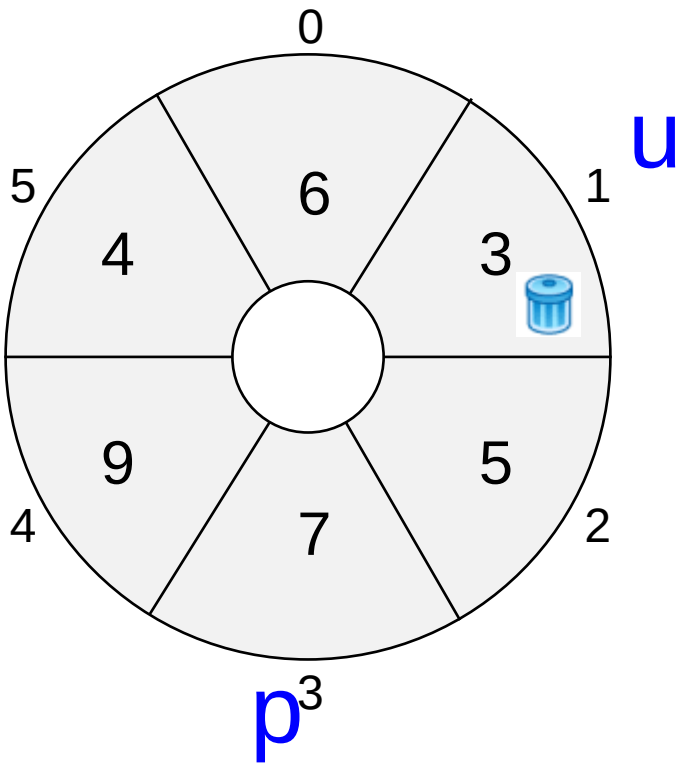
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



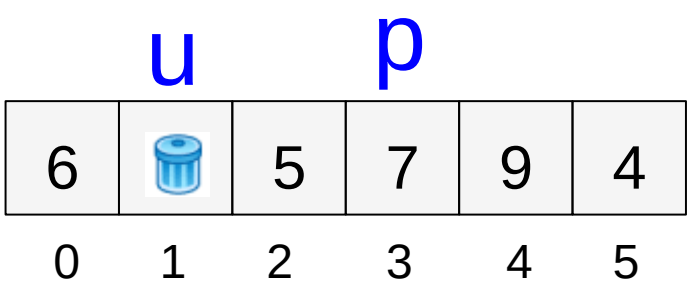
Algoritmo em C

```
//Remover()

 int remover() {

    if (primeiro == ultimo)
       exit(1);

    int resp = array[primeiro];
    primeiro = (primeiro + 1) % MAXTAM;
    return resp;
 }
```

resp | 5



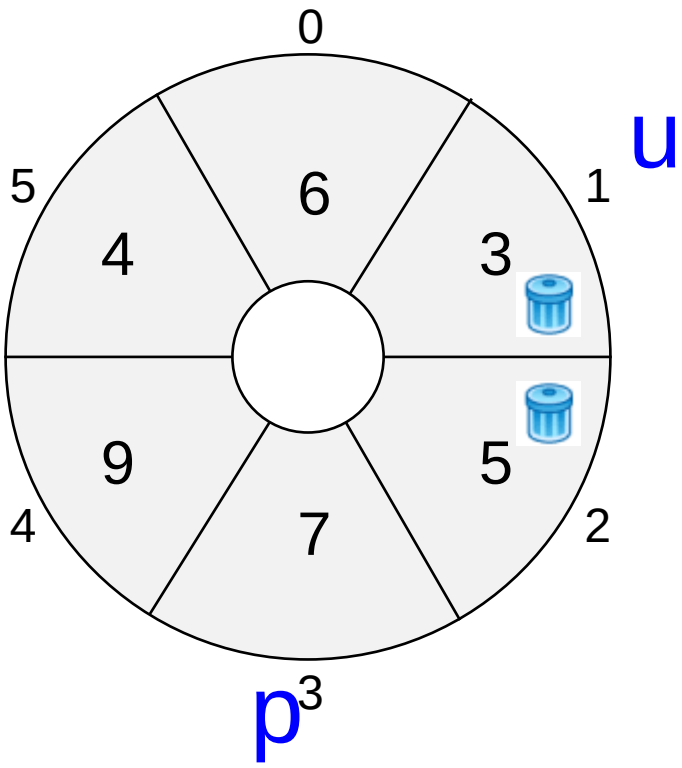
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



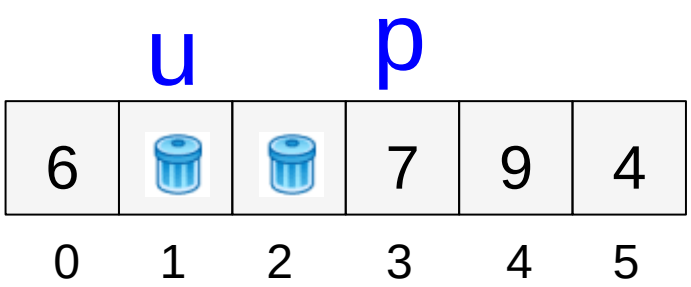
Algoritmo em C

```
//Remover()

int remover() {

   if (primeiro == ultimo)
      exit(1);

   int resp = array[primeiro];
   primeiro = (primeiro + 1) % MAXTAM;
   return resp;
}
```

resp 5

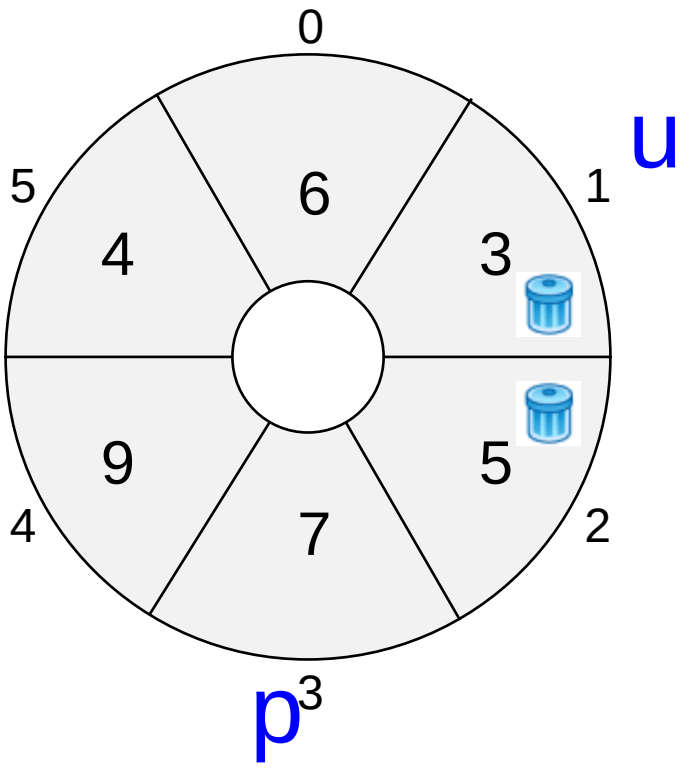


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

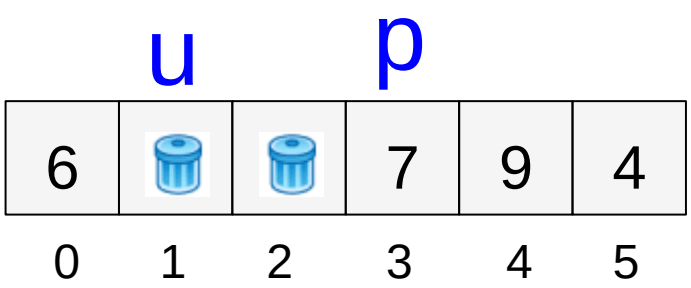


Algoritmo em C

```
//Remover()

int remover() {

   if (primeiro == ultimo)
      exit(1);

   int resp = array[primeiro];
   primeiro = (primeiro + 1) % MAXTAM;
   return resp;
}
```

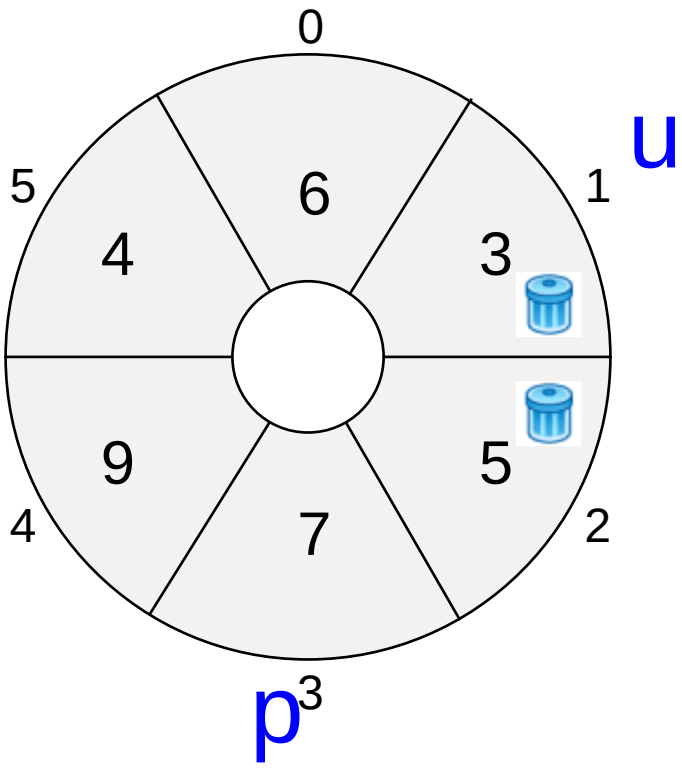


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

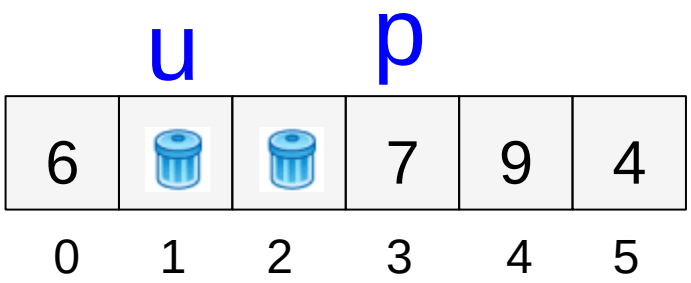


Algoritmo em C

```
//Inserir(8)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```



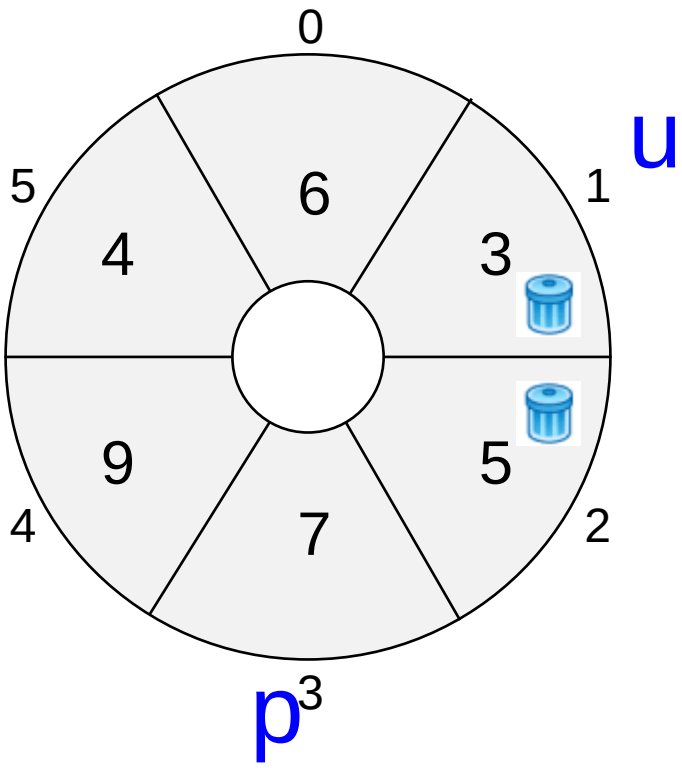
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



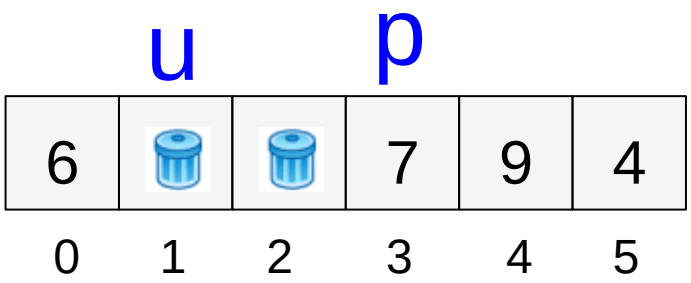
Algoritmo em C

```
//Inserir(8)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
        exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

false: 1 + 1 % 6 == 3

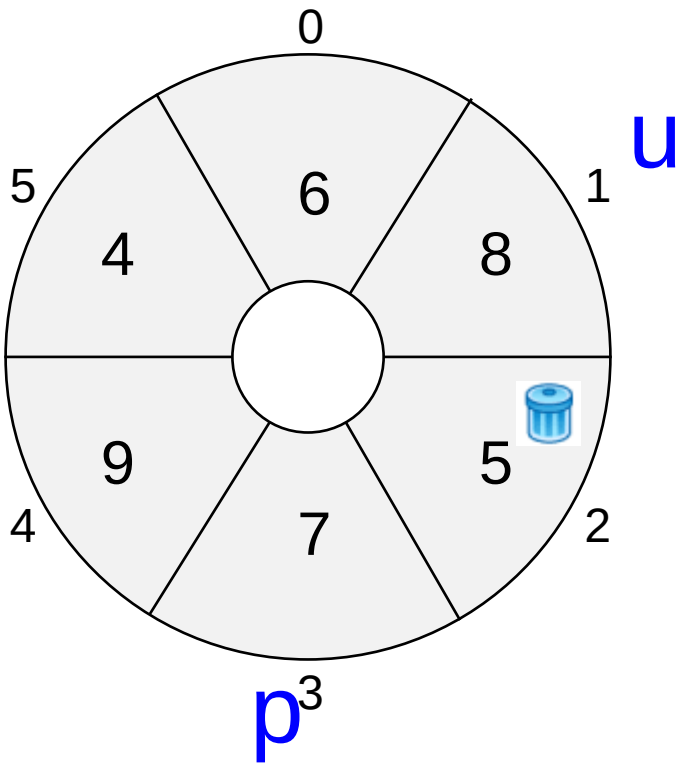


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

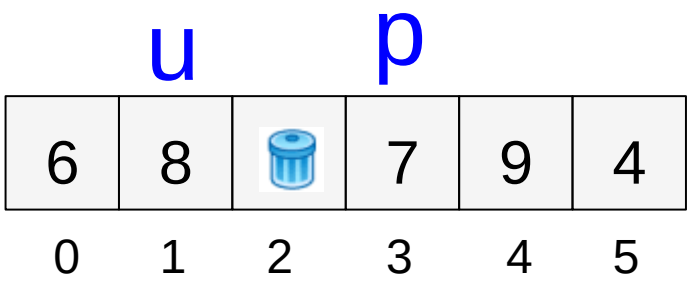


Algoritmo em C

```
//Inserir(8)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

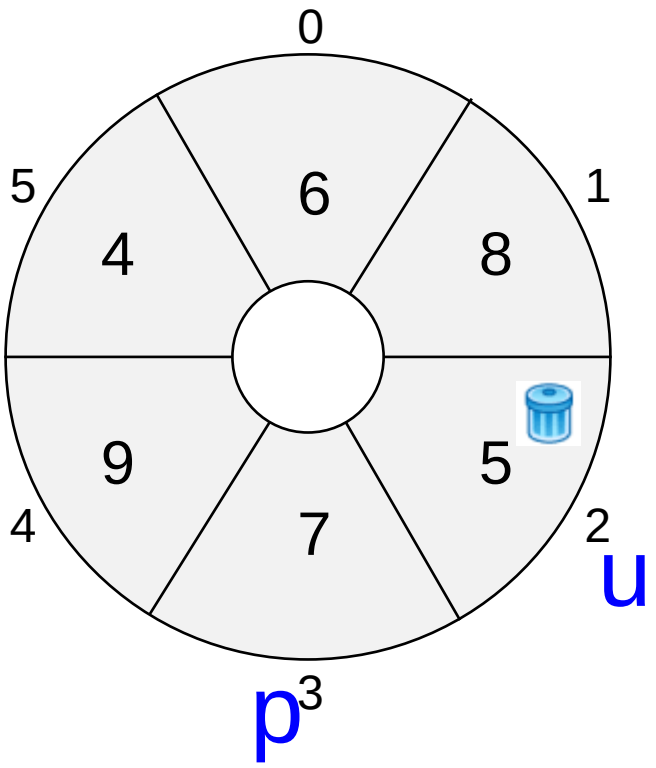


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

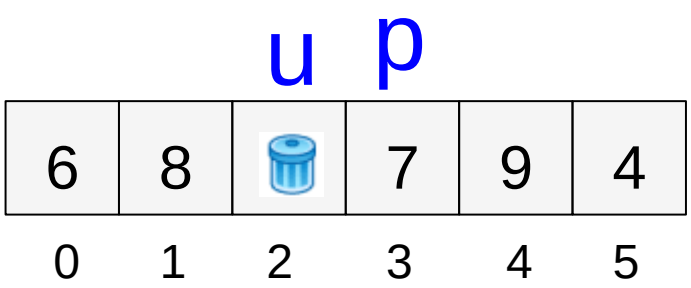


Algoritmo em C

```
//Inserir(8)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

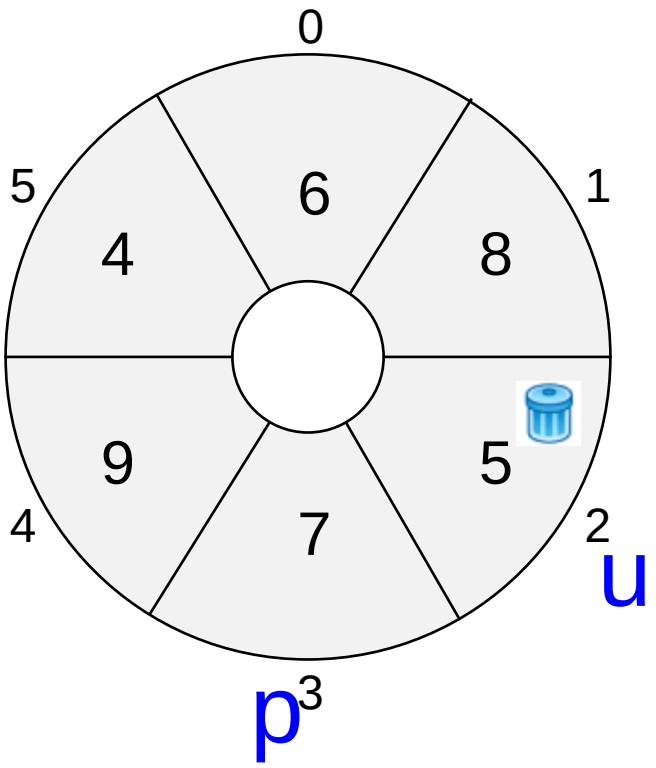


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()

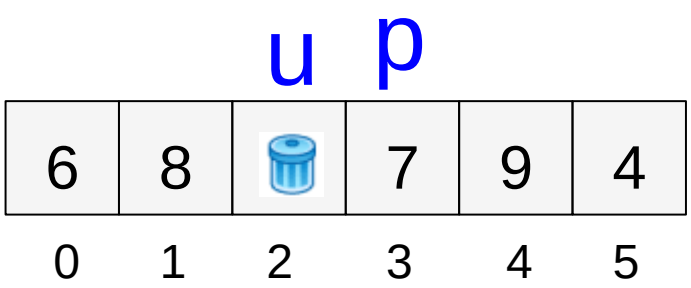


Algoritmo em C

```
//Inserir(8)

void inserir(int x) {

    if (((ultimo + 1) % MAXTAM) == primeiro)
       exit(1);

    array[ultimo] = x;
    ultimo = (ultimo + 1) % MAXTAM;
}
```

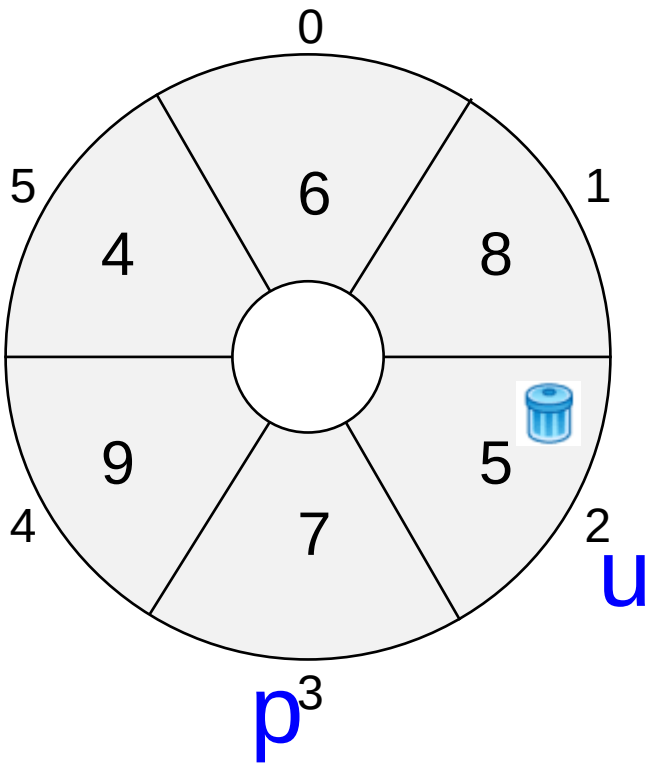


Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



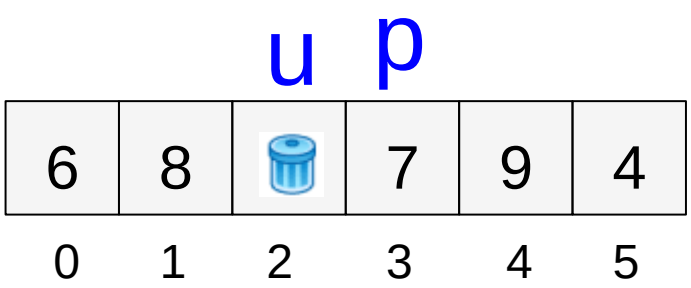
Algoritmo em C

```
void mostrar (){
   int i = primeiro;
   printf("[");

  while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
  }

  printf("]");
}
```



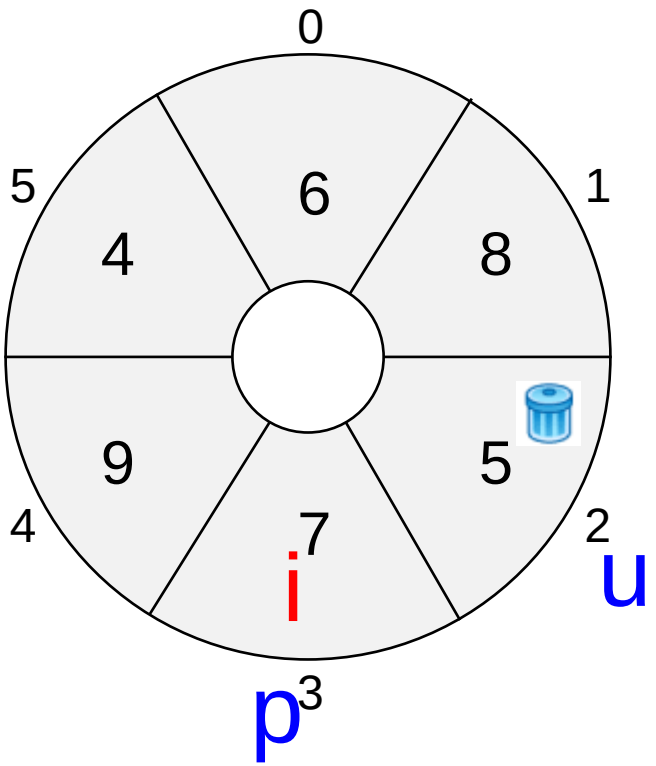
Tela:

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



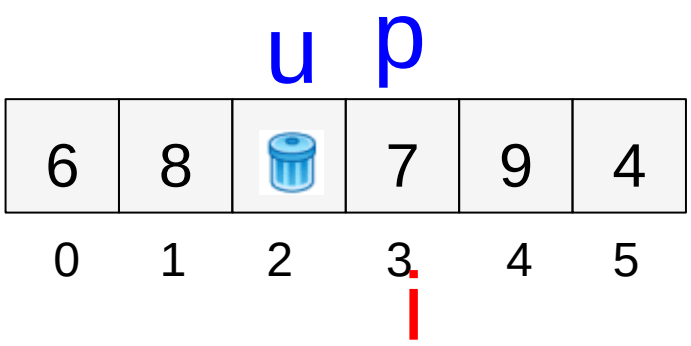
Algoritmo em C

```
void mostrar (){
   int i = primeiro;
   printf("[");

  while (i != ultimo) {
     printf(array[i] + " ");
     i = ((i + 1) % MAXTAM)
  }

  printf("]");
}
```



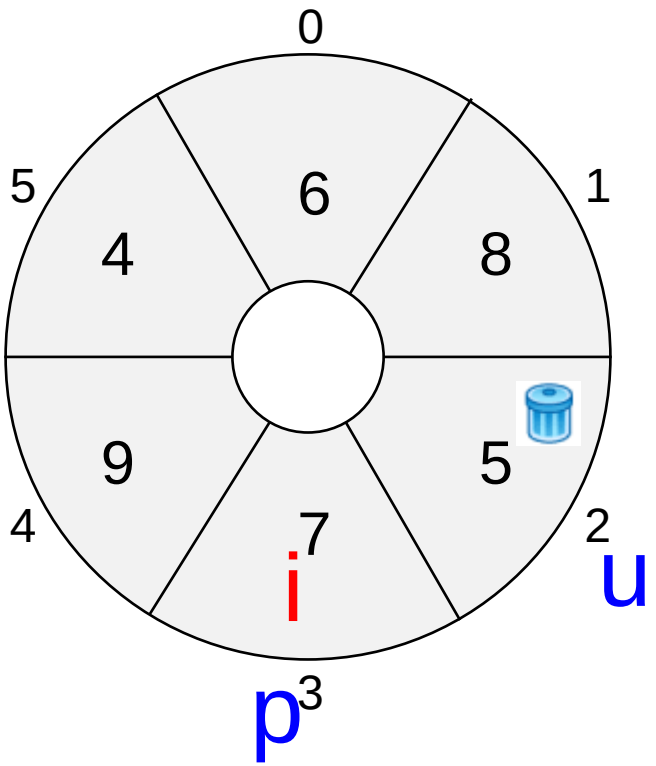
Tela:

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



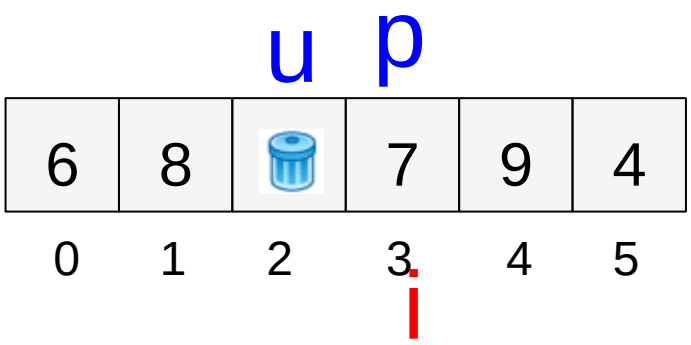
Algoritmo em C

```
void mostrar (){
   int i = primeiro;
   printf("[");

  while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
  }

  printf("]");
}
```



Tela: [

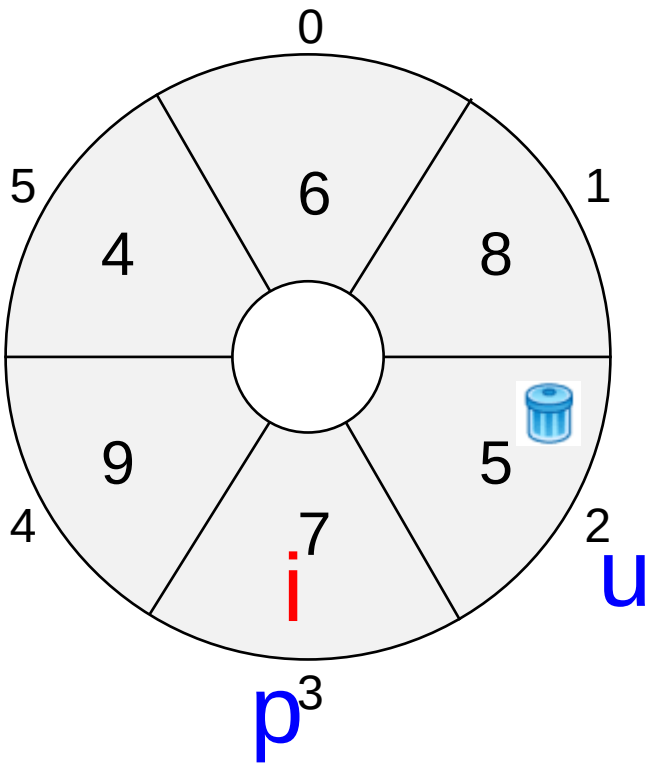
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em C

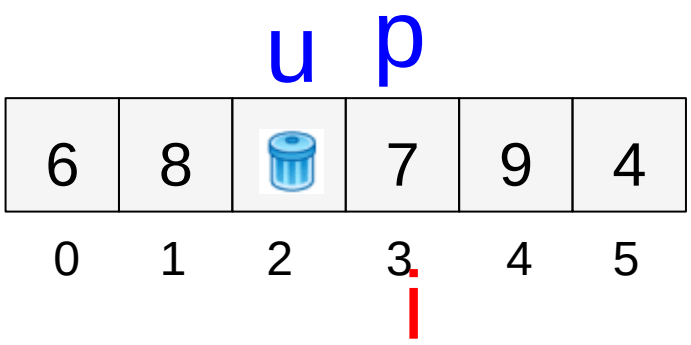
```
void mostrar (){
   int i = primeiro;
   printf("[");

   while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
   }

   printf("]");
}
```

true: 3 != 2



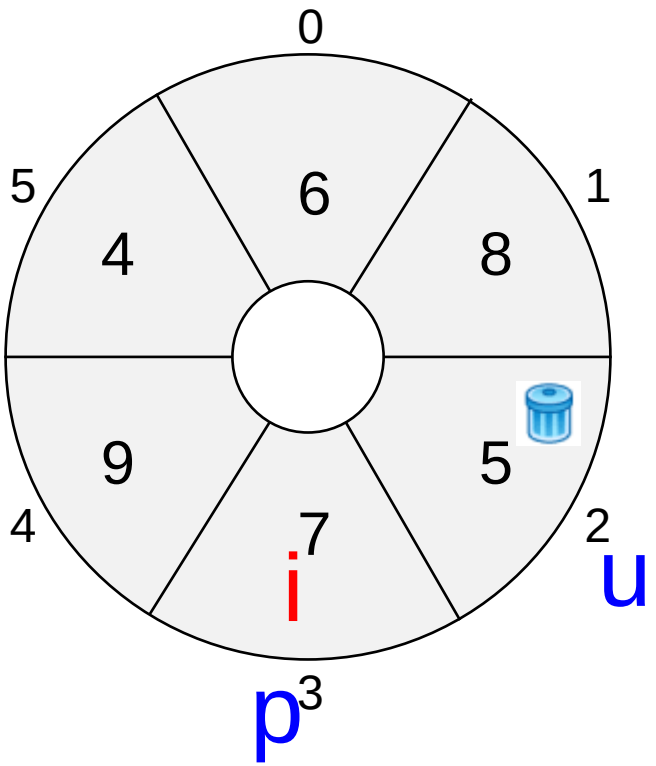
Tela: [

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



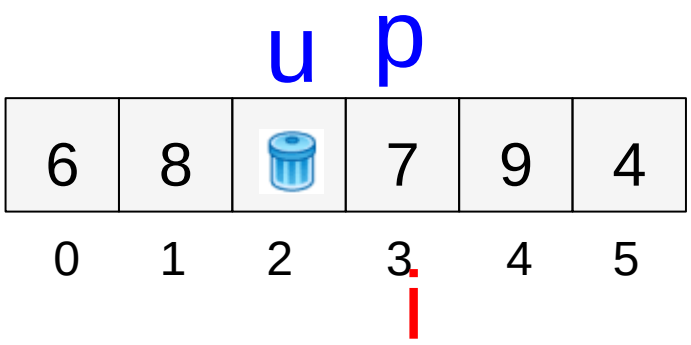
Algoritmo em C

```
void mostrar (){
   int i = primeiro;
   printf("[");

  while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
  }

  printf("]");
}
```



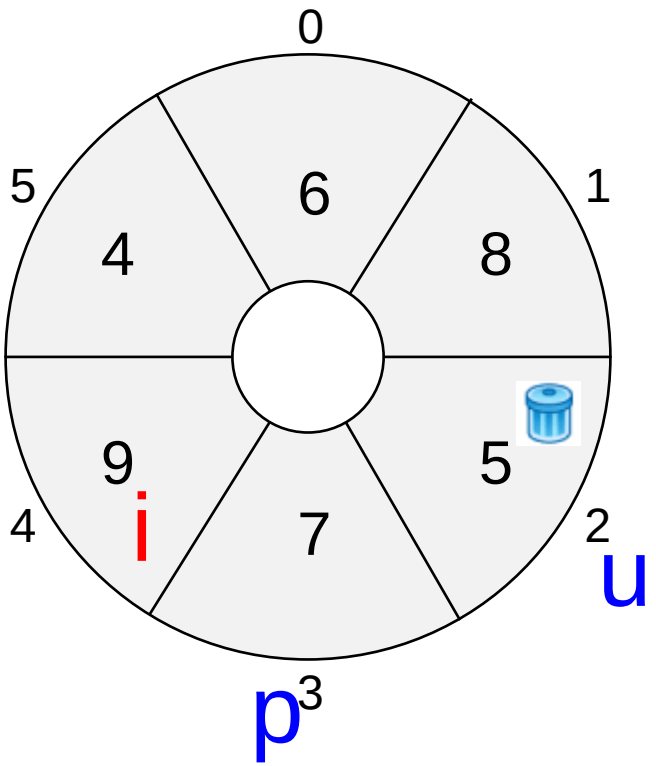
Tela: [7

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



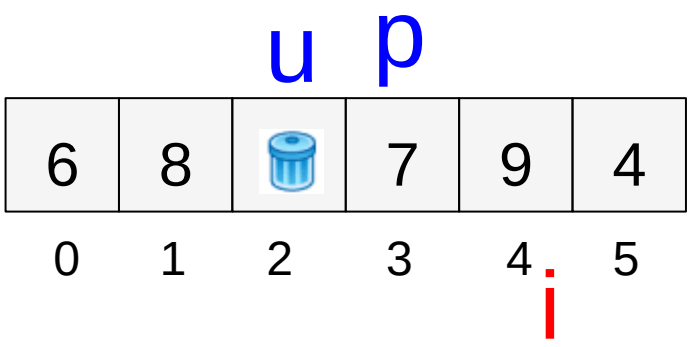
Algoritmo em C

```
void mostrar (){
   int i = primeiro;
   printf("[");

  while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
  }

  printf("]");
}
```



Tela: [7

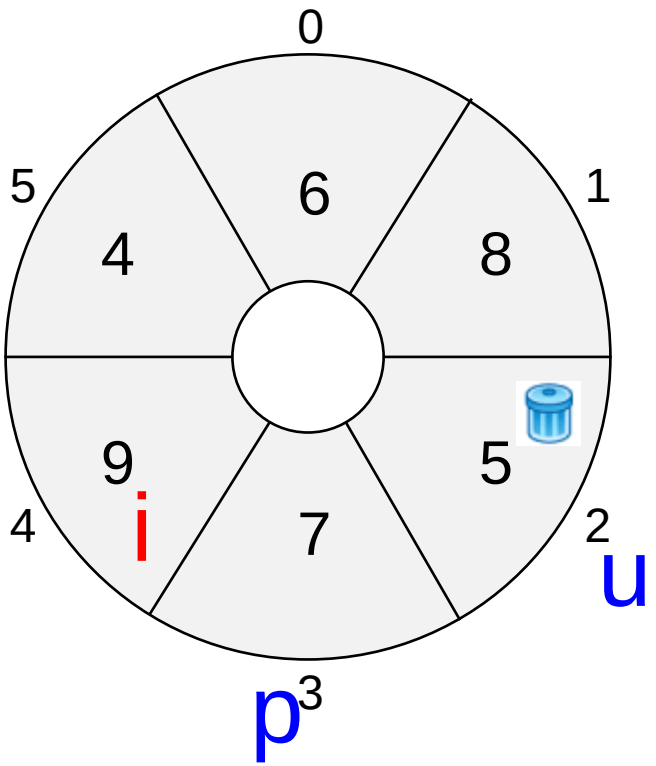
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em C

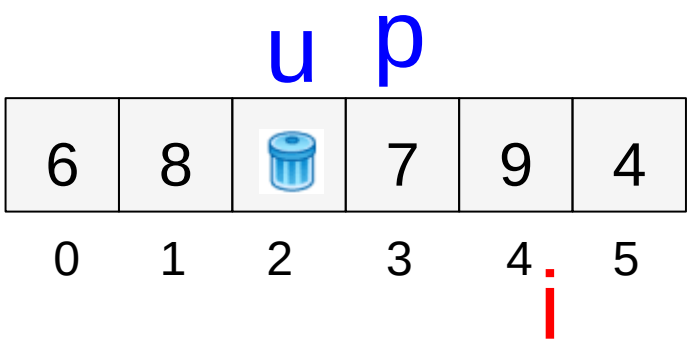
```
void mostrar (){
   int i = primeiro;
   printf("[");

   while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
   }

   printf("]");
}
```

true: 4 != 2



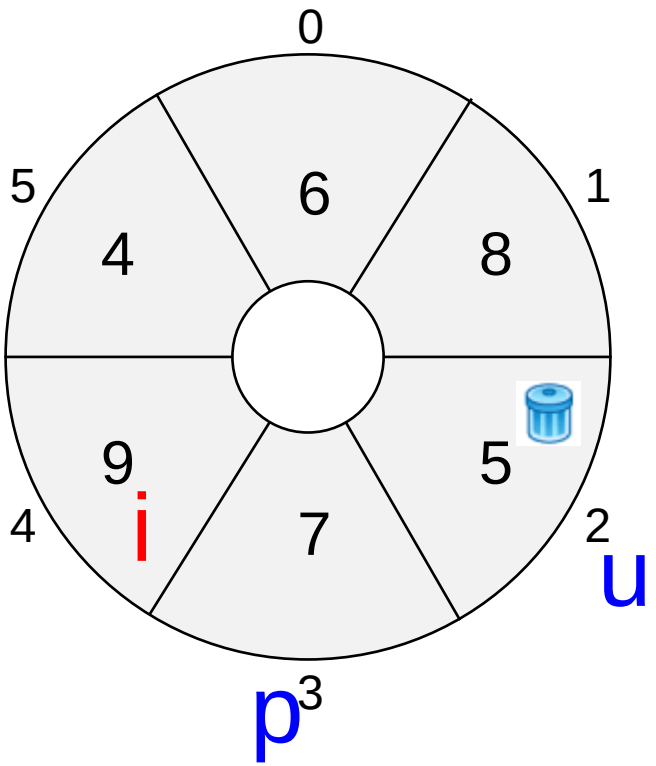
Tela: [7

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



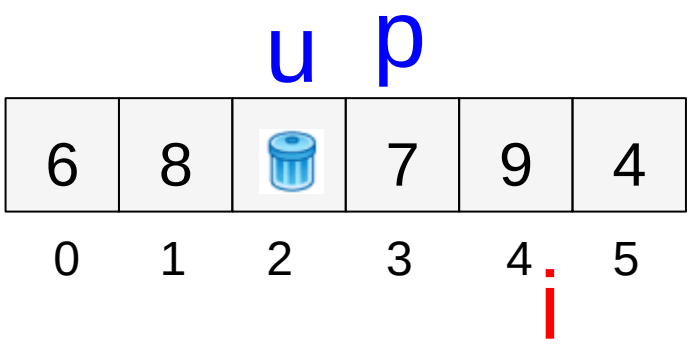
Algoritmo em C

```
void mostrar (){
   int i = primeiro;
   printf("[");

  while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
  }

  printf("]");
}
```



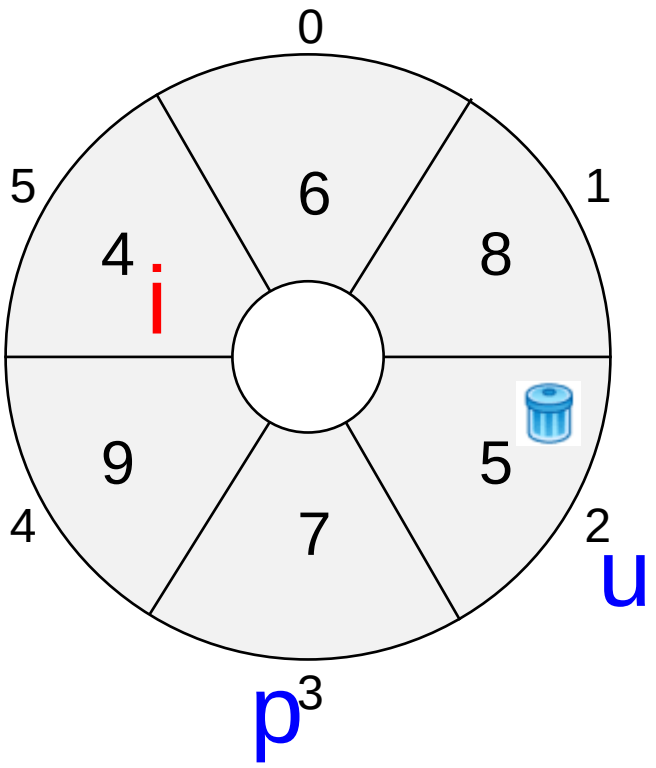
Tela: [7 9

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



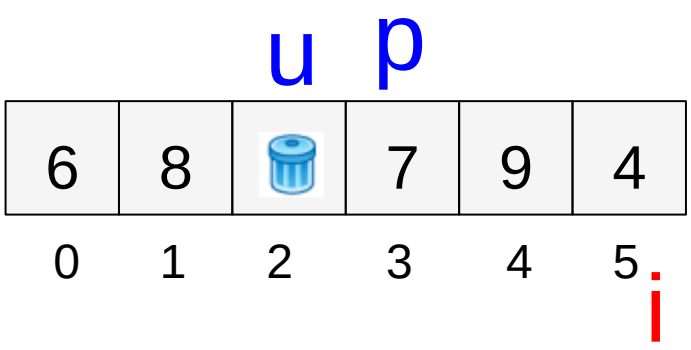
Algoritmo em C

```
void mostrar (){
   int i = primeiro;
   printf("[");

  while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
  }

  printf("]");
}
```



Tela: [7 9

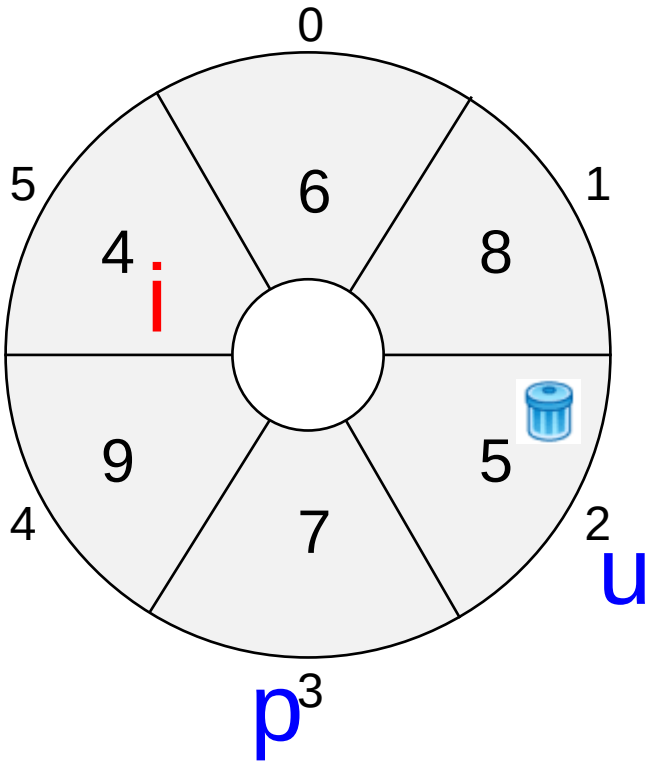
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em C

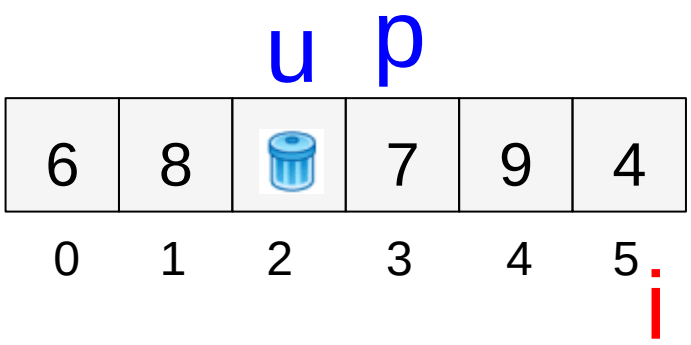
```
void mostrar (){
   int i = primeiro;
   printf("[");

   while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
   }

   printf("]");
}
```

true: 5 != 2



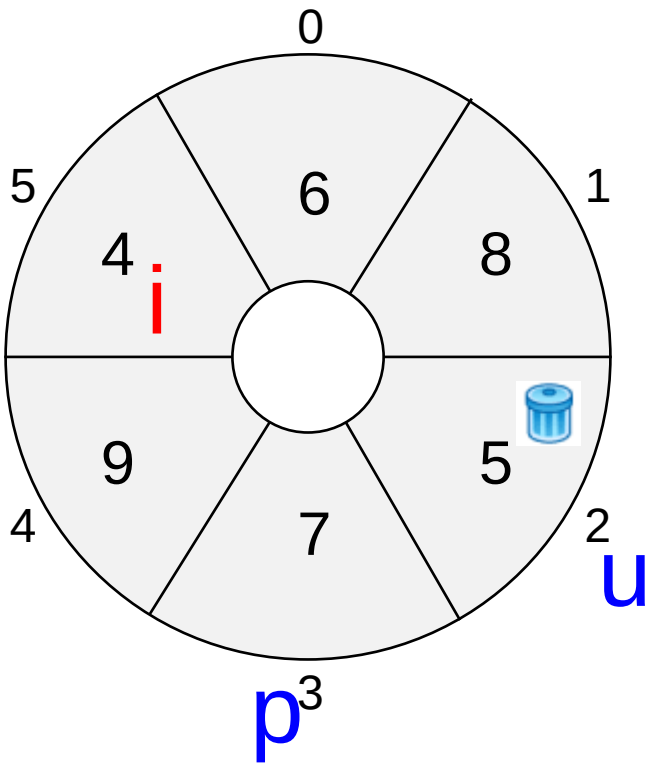
Tela: [7 9

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



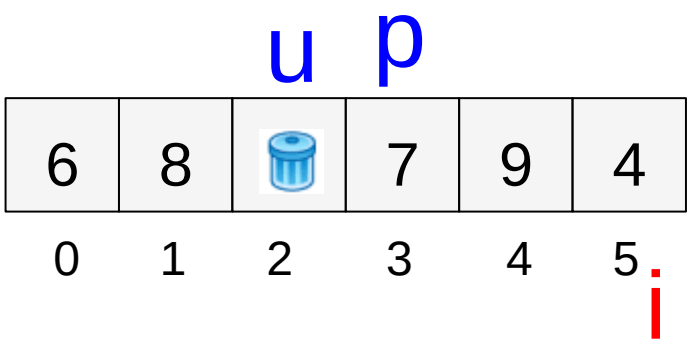
Algoritmo em C

```
void mostrar (){
   int i = primeiro;
   printf("[");

  while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
  }

  printf("]");
}
```



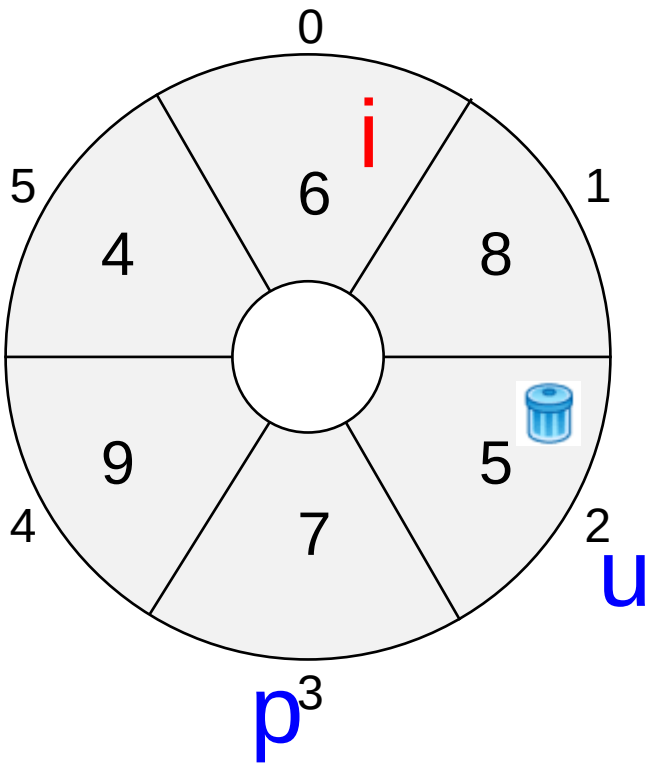
Tela: [7 9 4

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



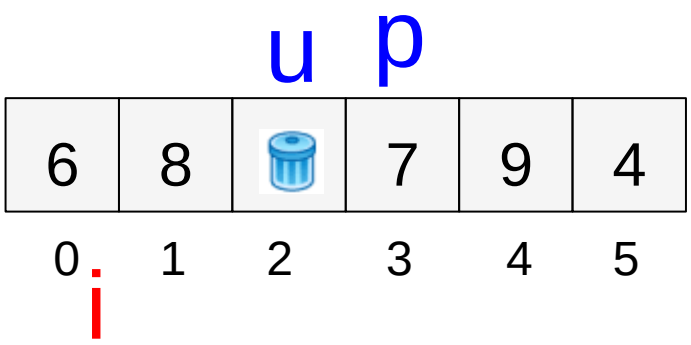
Algoritmo em C

```
void mostrar (){
   int i = primeiro;
   printf("[");

  while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
  }

  printf("]");
}
```



Tela: [7 9 4

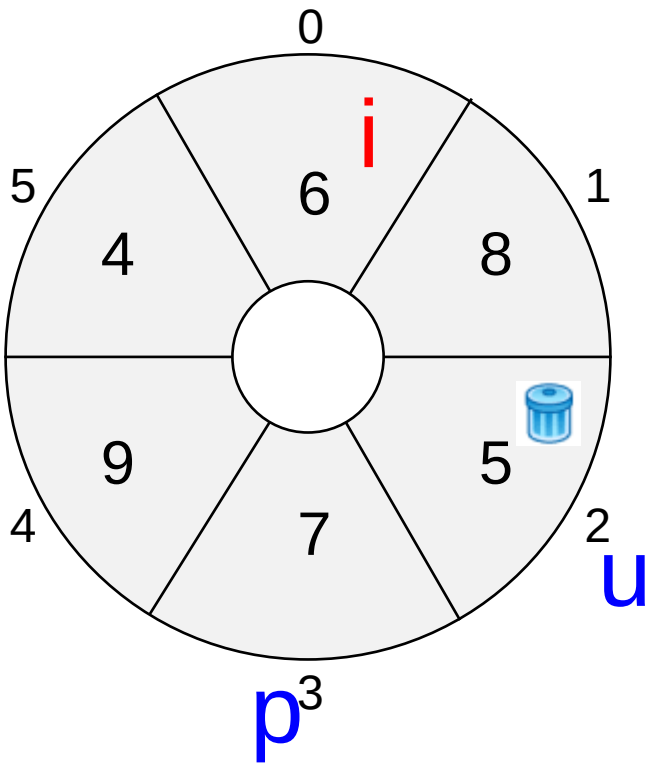
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em C

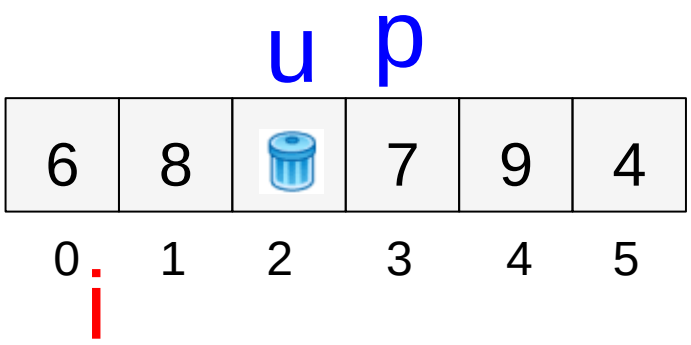
```
void mostrar (){
   int i = primeiro;
   printf("[");

   while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
   }

   printf("]");
}
```

true: 0 != 2



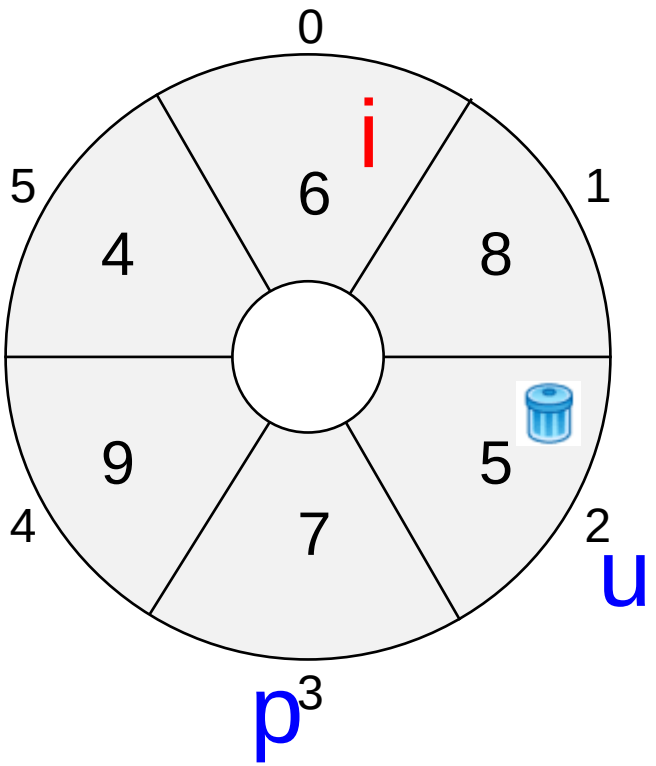
Tela: [7 9 4

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



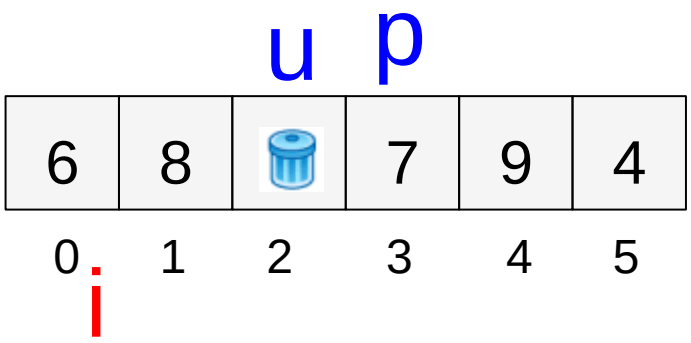
Algoritmo em C

```
void mostrar (){
   int i = primeiro;
   printf("[");

  while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
  }

  printf("]");
}
```



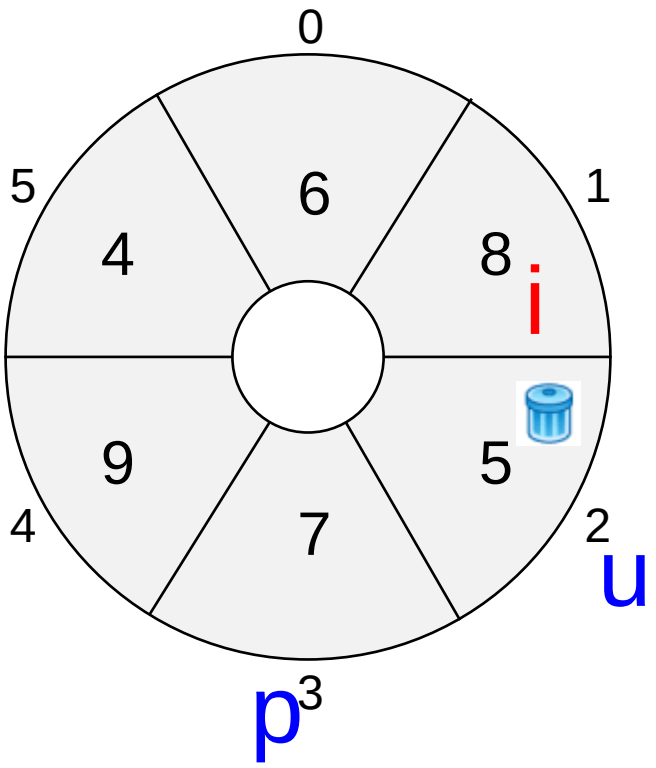
Tela: [7 9 4 6

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



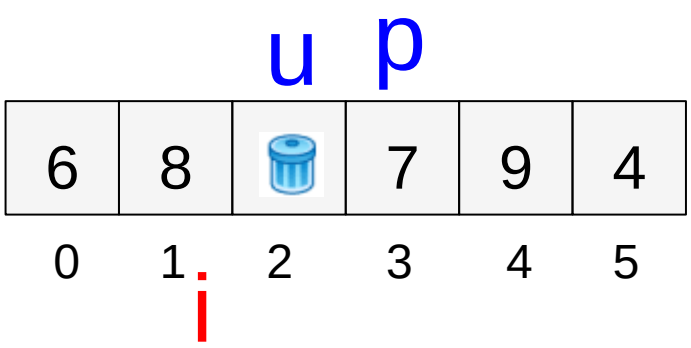
Algoritmo em C

```
void mostrar (){
   int i = primeiro;
   printf("[");

  while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
  }

  printf("]");
}
```



Tela: [7 9 4 6

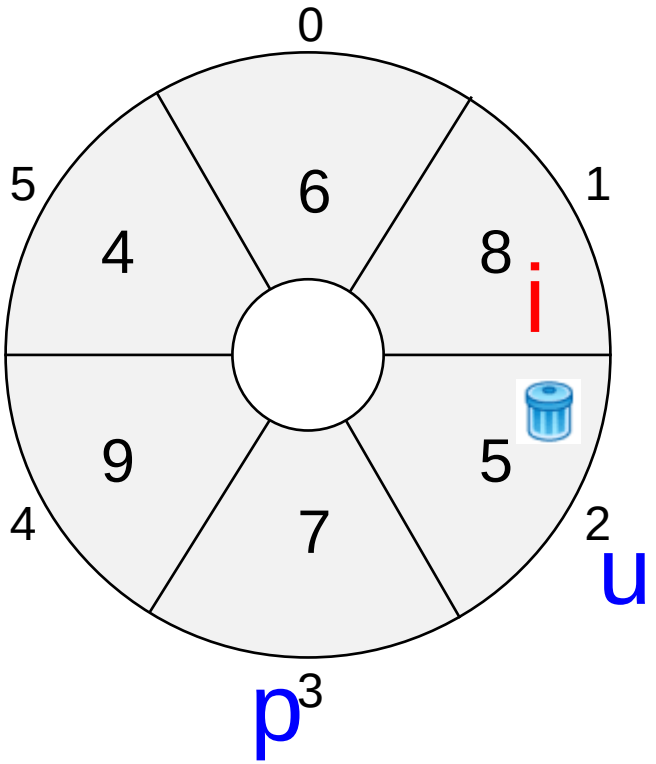
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em C

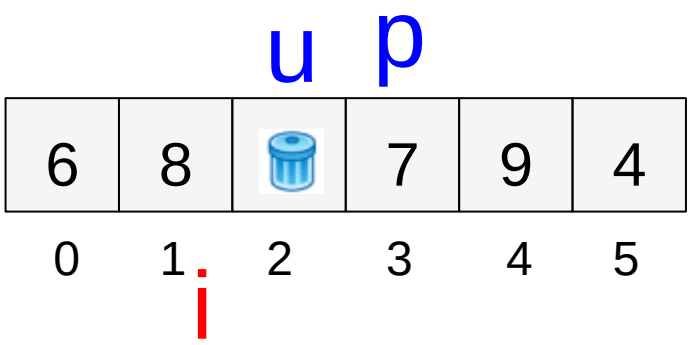
```
void mostrar (){
   int i = primeiro;
   printf("[");

   while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
   }

   printf("]");
}
```

true: 1 != 2



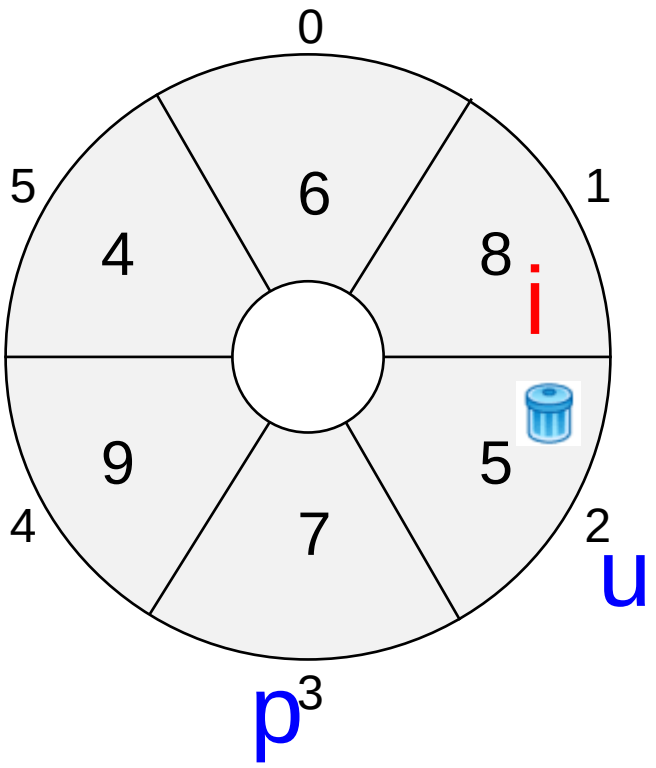
Tela: [7 9 4 6

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



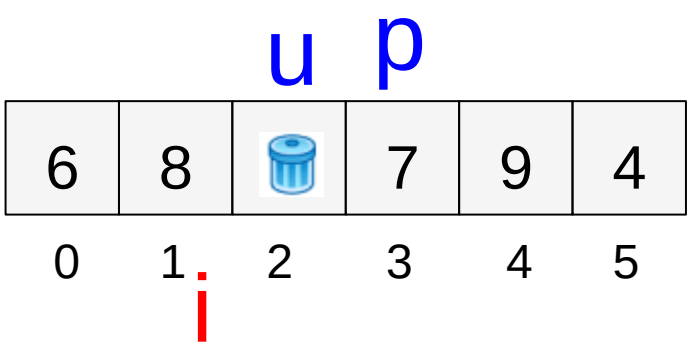
Algoritmo em C

```
void mostrar (){
   int i = primeiro;
   printf("[");

  while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
  }

  printf("]");
}
```



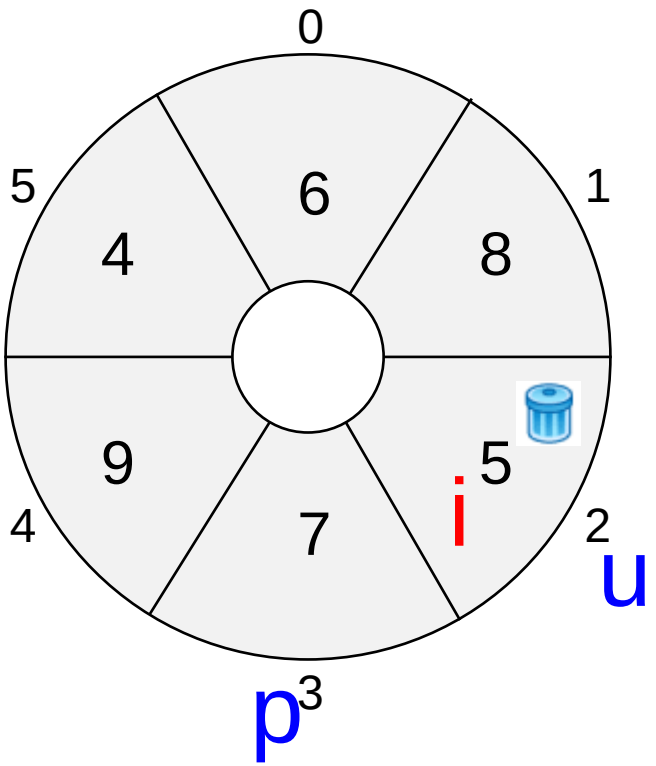
Tela: [7 9 4 6 8

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



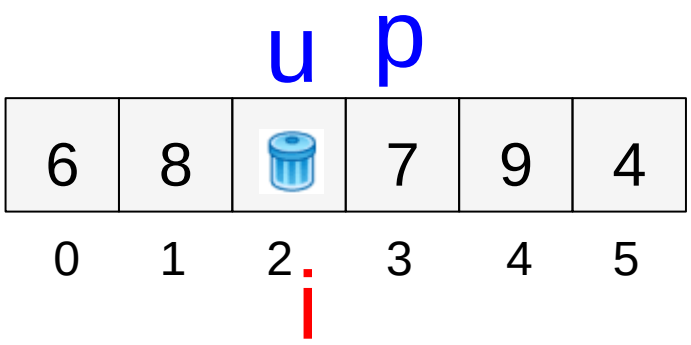
Algoritmo em C

```
void mostrar (){
   int i = primeiro;
   printf("[");

  while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
  }

  printf("]");
}
```



Tela: [7 9 4 6 8

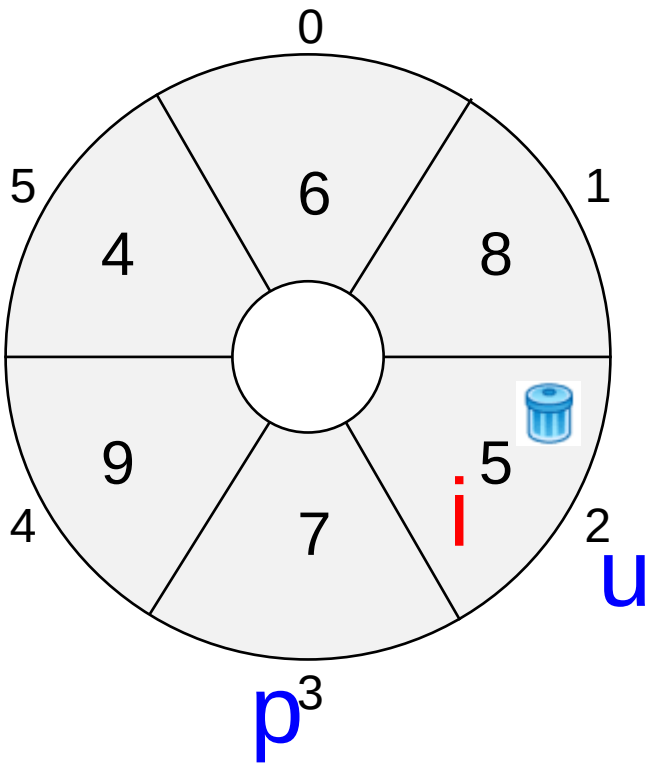
Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Algoritmo em C

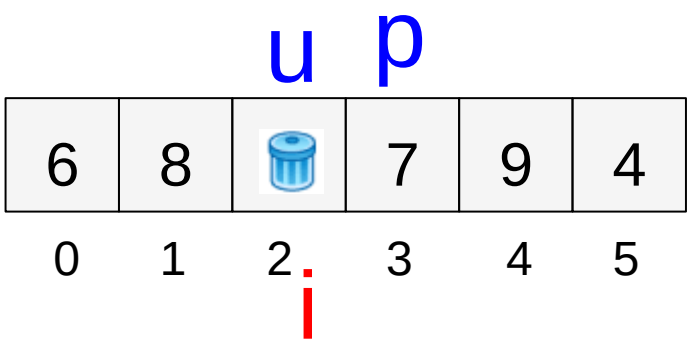
```
void mostrar (){
   int i = primeiro;
   printf("[");

   while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
   }

   printf("]");
}
```

false: 2 != 2



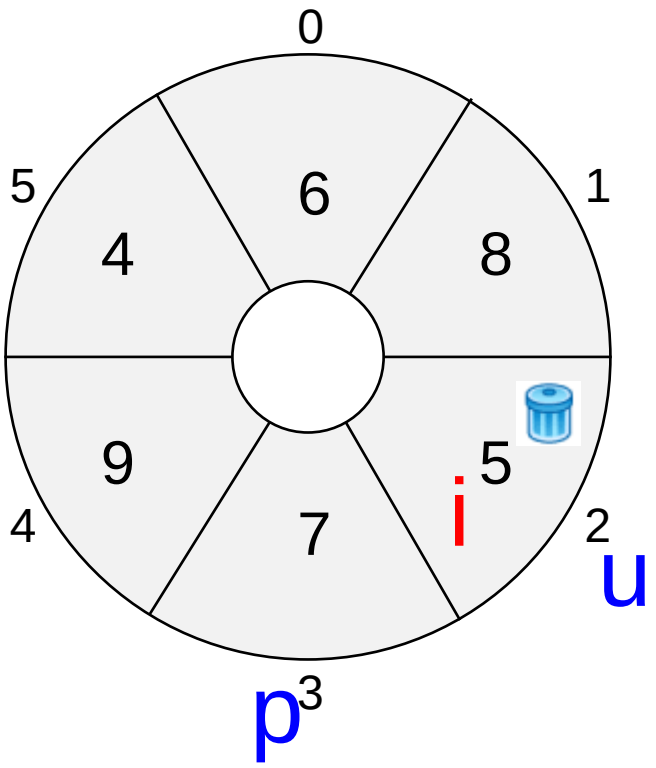
Tela: [7 9 4 6 8

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



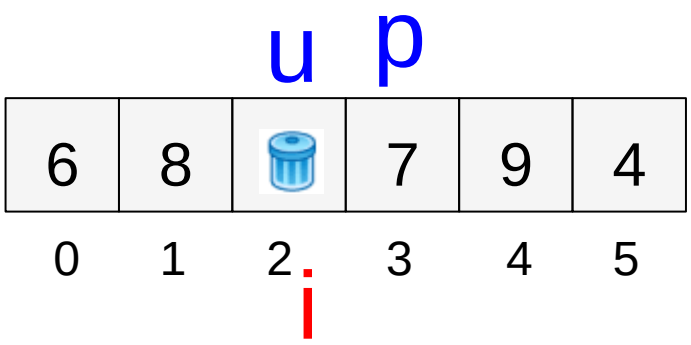
Algoritmo em C

```
void mostrar (){
   int i = primeiro;
   printf("[");

  while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
  }

  printf("]");
}
```



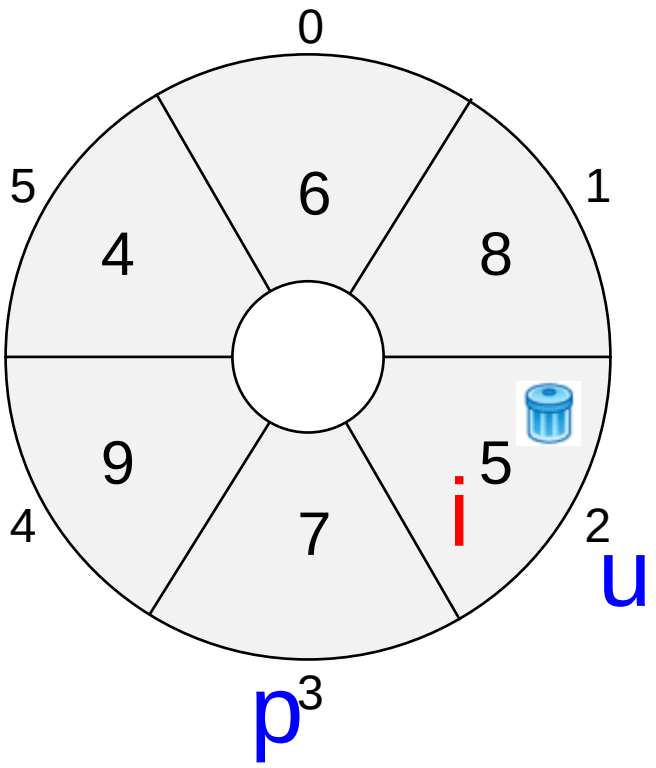
Tela: [7 9 4 6 8]

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



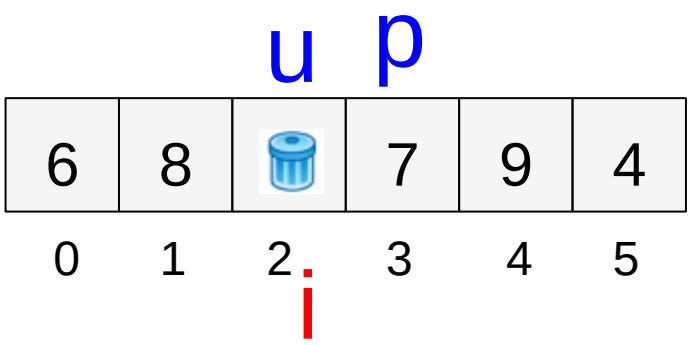
Algoritmo em C

```
void mostrar (){
   int i = primeiro;
   printf("[");

  while (i != ultimo) {
      printf(array[i] + " ");
      i = ((i + 1) % MAXTAM)
  }

  printf("]");
}
```



Tela: [7 9 4 6 8]

Vamos criar uma fila com tamanho cinco e efetuar as operações I(1), I(3), I(5), I(7), I(9), I(2), R(), R(), I(4), I(6), R(), I(8), M()



Exercício

- Implemente o método isVazio()
- Implemente o método mostrar de forma recursiva
- Implemente o método pesquisar