PONTEIROS
ALOCAÇÃO DE MEMÓRIA

Introdução à Computação I

Ponteiros

- Apontadores
- Uma variável que aponta para outra variável
- Armazena um endereço de memória
- Recurso poderoso
- Na linguagem C é usado para:
  - Estruturas de dados dinâmicas
  - **Passagem de parâmetros em funções (referência)**
  - Alternativa para acessar vetores
  - Códigos mais eficientes

Ponteiros

- Declaração
  - tipo *identificador;
- Operadores
  - &var1: obtém o endereço
  - *var2: obtém o conteúdo

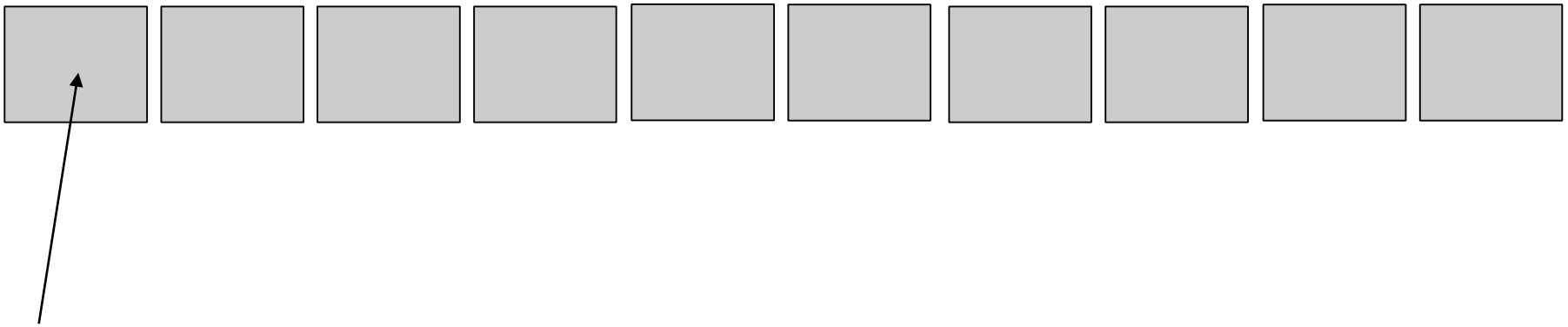
Ponteiros

```
int main()
{
  int a = 10;
  int *b=NULL;

  printf("conteudo de a: %d\n", a);
  printf("endereco de a: %d\n", &a);

  b = &a;
  printf("conteudo de b: %d\n", *b);
  printf("endereco de b: %d\n", b);

  *b = 15;
  printf("conteudo de a: %d\n", a);
  printf("conteudo de b: %d\n", *b);
  printf("conteudo de a: %d\n", *(&a));

  getch();
}
```

Ponteiros e Vetores

Em C um vetor é um apontador (ponteiro) para a sua primeira posição (índice 0)

int v[10];



v ou &v[0] *(em C int v[ ] e int *v são sinônimos)*

Identificador do vetor é um ponteiro

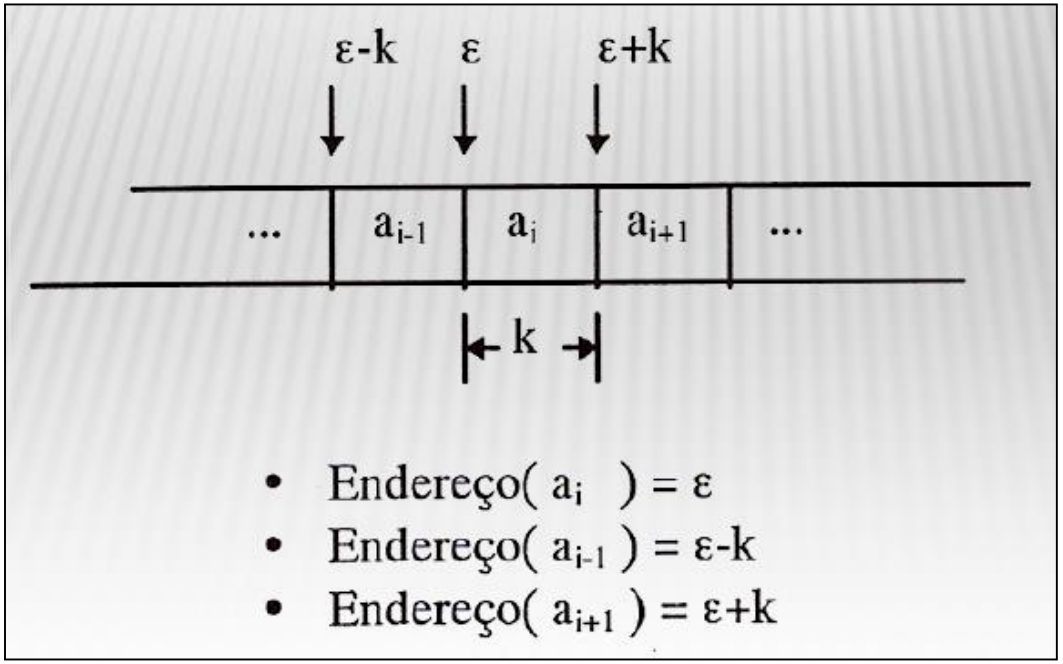
a[0] e *a são iguais (acessam a mesma posição), logo: a[i] é igual a *(a + i)

Alocação de Memória

- **Sequencial**
- Encadeada

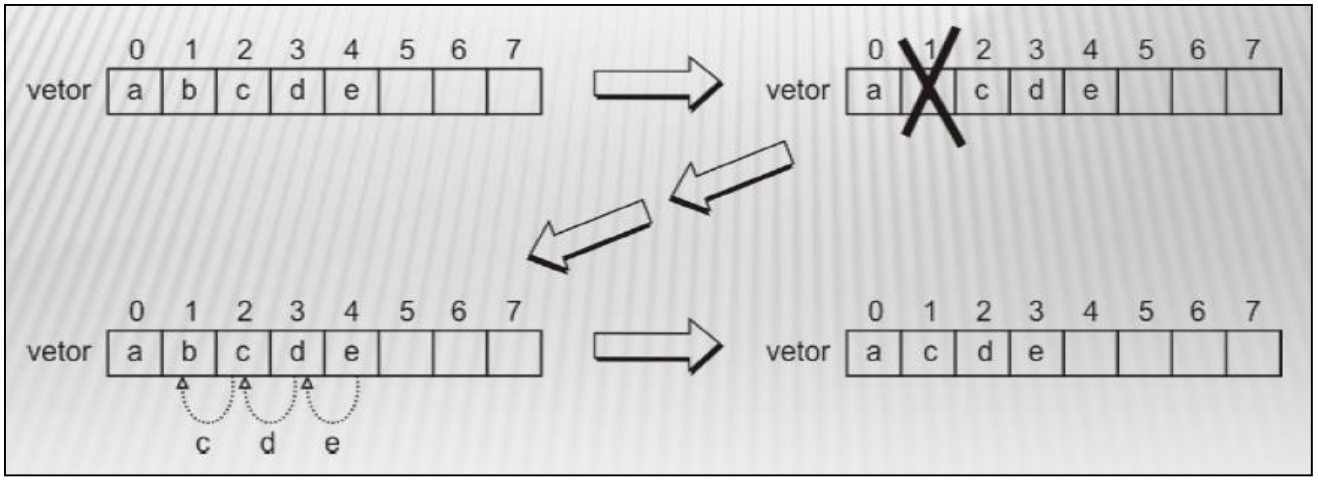
Alocação de Memória Sequencial

- Células de memória consecutivas
- Se cada célula tem um endereço único ε e utiliza k bytes, tem-se que:



Alocação de Memória Sequencial

- Vantagem: possibilidade de acessar diretamente por meio de índices um elemento qualquer de uma estrutura
- Desvantagem: quando precisamos inserir ou apagar elementos no meio do vetor, o vetor deve ser rearranjado



Alocação de Memória Sequencial

- Sequencial
  - Estática
    - A quantidade total de memória utilizada pelos dados **é** previamente conhecida e definida
  - Dinâmica
    - A quantidade total de memória utilizada pelos dados **não é** previamente conhecida e definida
  - A diferença da alocação sequencial dinâmica em relação à estática está somente na possibilidade da dinâmica poder redimensionar o uso da memória, se necessário

Alocação de Memória Sequencial

- Sequencial
  - Estática
  - Dinâmica

Sequencial Estática

```
int main()
 {
    const int tam = 5;
    int i;
    int vetor[tam];

    for(i=0;i<tam;i++)
     vetor[i]=i+1;

    for(i=0;i<tam;i++)
     printf("%d ",vetor[i]);

    getch();
}
```

Alocação de Memória Sequencial

- Sequencial
  - Estática
  - Dinâmica

Sequencial Dinâmica

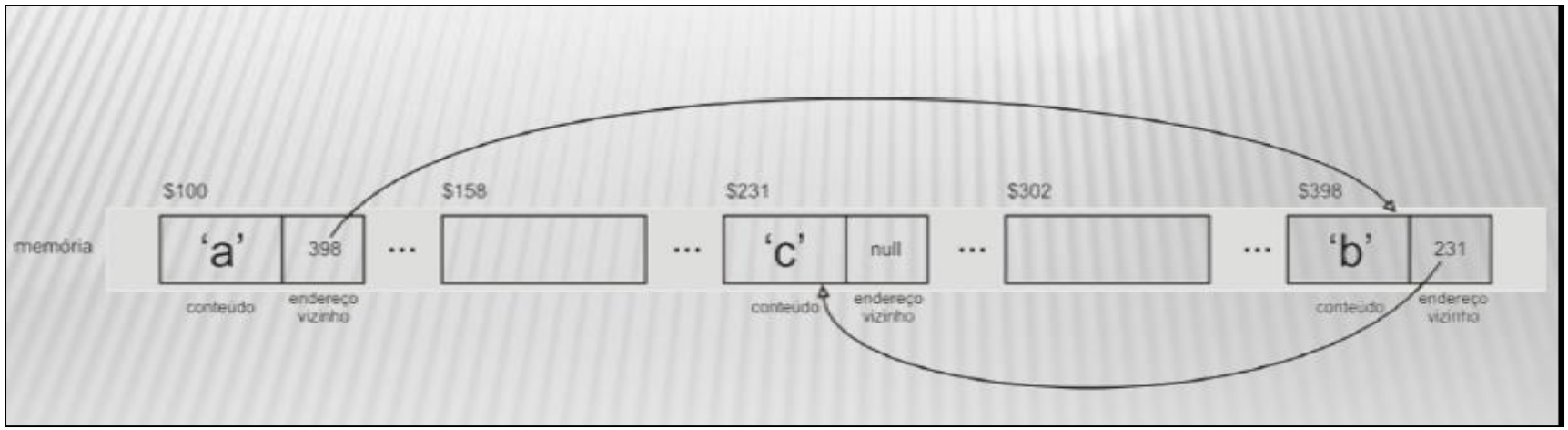
```
int main()
 {
    int N, i;
    int *vetor

    printf("Entre com o valor do vetor: ");
    scanf("%d",&N);
    vetor = (int *)malloc(sizeof(int)*N);

    for(i=0;i<N;i++)
     vetor[i]=i+1;

    for(i=0;i<N;i++)
     printf("%d ",vetor[i]); //como vetor

    printf("\n\n");
    for(i=0;i<N;i++,vetor++)
     printf("%d ",*vetor);     //como ponteiro

    free(vetor); // libera memória alocada

    getch();
}
```

Alocação de Memória

- Sequencial
- **Encadeada**

Alocação de Memória Encadeada

- A alocação encadeada de memória propicia a possibilidade de alocar células distribuídas e não consecutivas na memória
- Em cada célula armazena-se o elemento (conteúdo) e o endereço de memória de seu vizinho (anterior e/ou próximo)



Alocação de Memória Encadeada

- Vantagem: possibilidade de utilizar somente a quantidade de memória necessária em determinado instante
- Desvantagem: ao contrário da alocação sequencial, não podemos acessar uma posição diretamente por seu índice. É necessário "procurar" a posição desejada sempre