Algoritmos e Programação II

Aula 6

UNIVALI

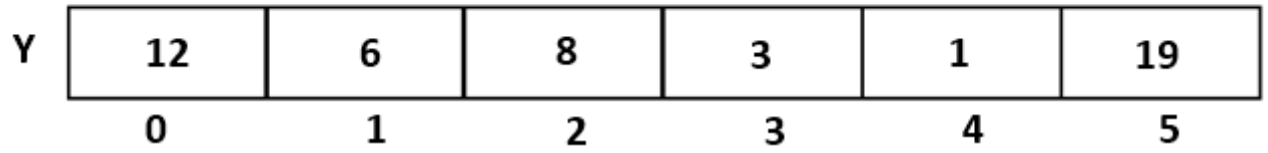
Escola Politécnica

Vetores

- Vetores, ou arrays, são uma estrutura que armazena uma sequência de dados, todos do mesmo tipo, em posições consecutivas da memória.

- São identificados por um nome, onde as posições de memória são identificadas por índices (valores inteiros).

- Vetores permitem armazenar e manipular coleções de valores relacionados de forma eficiente.



Declaração de vetores

- Sintaxe:
  - tipo identificador [nº de posições de memória];

- tipo = int, float, char, ..., são os tipos dos dados que o vetor armazena.
- identificador = nome do vetor.

- nº de posições de memória = tamanho do vetor = número de elementos que o vetor pode armazenar.

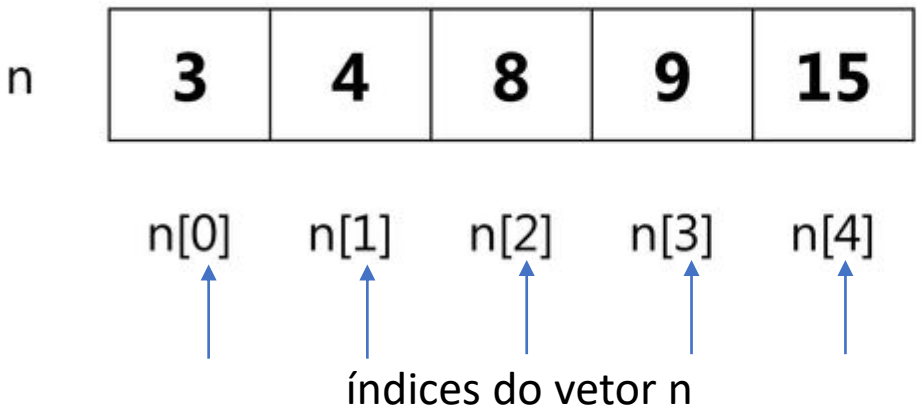
```c
int vetor[5];
```

```c
double notas[50];
```

```c
char palavra[20];
```

Alocação estática

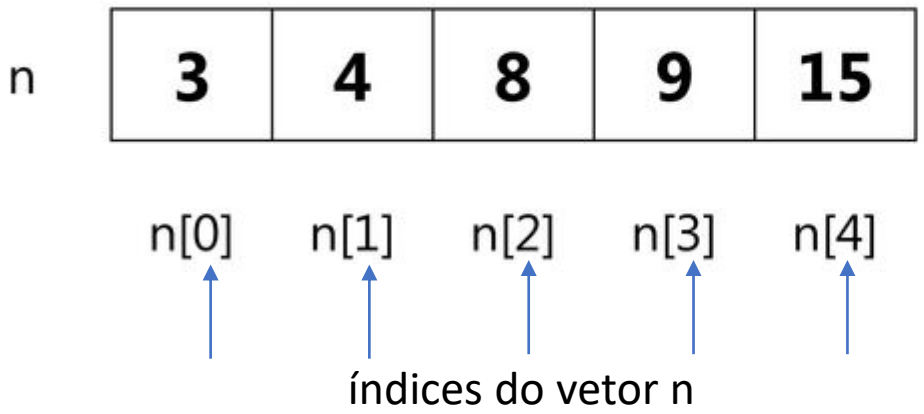
```
43  int main() {
44
45      int n[5];
46      n[0] = 3;
47      n[1] = 4;
48      n[2] = 8;
49      n[3] = 9;
50      n[4] = 15;
51
52  }
```



UNIVALI

Escola Politécnica

```
43  int main() {
44
45      int n[5] = {3,4,8,9,15};
46
47      int v[ ] = {3,4,8,9,15};
48
49  }
50
```



UNIVALI

Escola Politécnica

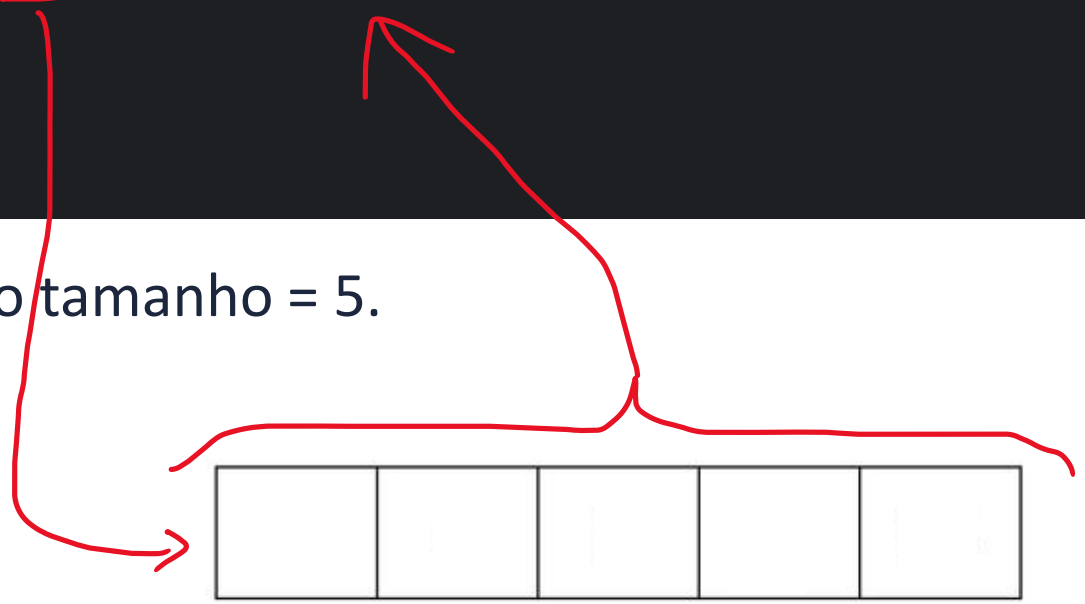
Alocação dinâmica

- Reserva de espaço na memória do computador durante a execução do programa.
- Isso é útil quando não sabemos o tamanho exato de uma estrutura de dados.
- Precisamos usar o comando new e ponteiros:
  - tipo *ponteiro = new tipo[tamanho];
- tipo = int, float, char, ..., são os tipos dos dados que o vetor armazena.
- *ponteiro = variável que armazena o endereço de memória do vetor alocado dinamicamente = nome do vetor.
- Precisamos de ponteiro pois não sabemos do endereço do vetor antes da execução. Durante a execução será decidido a região de memória onde residirá o vetor. Só nesse momento saberemos o endereço. O ponteiro serve para estarmos preparados para armazenar esse endereço;

```
43  int main() {
44
45      int* vetor; // Declarando um ponteiro para um vetor de inteiros
46      int tamanho;
47      cin >> tamanho;
48      vetor = new int[tamanho]; // Alocando dinamicamente um vetor de inteiros
49
50  }
51
```

```
43  int main() {
44
45      int* vetor; // Declarando um ponteiro para um vetor de inteiros
46      int tamanho;
47      cin >> tamanho;
48      vetor = new int[tamanho]; // Alocando dinamicamente um vetor de inteiros
49
50  }
51
```

- Supondo tamanho = 5.



- Inserindo e lendo valores

```
43  int main() {
44
45      int* vetor; // Declarando um ponteiro para um vetor de inteiros
46      int tamanho;
47      cin >> tamanho;
48      vetor = new int[tamanho]; // Alocando dinamicamente um vetor de inteiros
49
50      for(int i =0; i<tamanho; i++){
51          cin >> vetor[i];
52      }
53      for(int i =0; i<tamanho; i++){
54          cout << vetor[i] << endl;
55      }
```

Passando vetores como argumento

```
42  void minhaFuncao(int vetor[], int tamanho) {
43      for (int i = 0; i < tamanho; i++) {
44          cout << vetor[i] << endl;
45      }
46  }
47
48  int main() {
49
50      int *vetor;
51      int tamanho;
52      cin >> tamanho;
53      vetor = new int[tamanho];
54      for(int i = 0; i<tamanho; i++) {
55          cin >> vetor[i];
56      }
57      minhaFuncao(vetor,tamanho);
58  }
```

Se precisarmos modificar o vetor:

```
42  void criar(int *vetor, int tamanho){
43
44      cout << "preencha " << endl;
45      for (int i = 0; i < tamanho; ++i) {
46          cin >> vetor[i];
47      }
48  }
49
50
51  int main() {
52      int vetor[] = {1,2,3,4,5};
53
54      int tamanho;
55      cin >> tamanho;
56      int *vetord = new int[tamanho];
57      criar(vetord,tamanho);
58  }
```

UNIVALI Escola Politécnica

Calculando o tamanho de vetores

```
43  int main() {
44
45      int vetor[] = {1,2,3,4,5};
46
47      cout << sizeof(int) << " bytes" << endl;
48
49      cout << sizeof(vetor) << " bytes" << endl;
50
51      // quantos números inteiros cabem em um vetor de 20 bytes?
52      cout << sizeof (vetor)/sizeof(vetor[0]) << endl;
53
54  }
55
```

```
untitled ×
⋮
C:\Users\jhona\CLionProjects\untitled\cmake-build-debug\untitled.exe
4 bytes
20 bytes
5
```

UNIVALI Escola Politécnica

Exercícios

Crie funções em C++ para (os vetores devem ser passados como argumento):

1. Somar os elementos de um vetor.
2. Calcular a Média de Elementos de um vetor
3. Verificar se um elemento existe em um vetor
4. Contar um determinado valor no vetor
5. Determinar o maior e menor valor de um vetor
6. Verificar quantos valores distintos existem em um vetor
7. Substituir valores ímpares por pares aleatórios em um vetor
8. Copiar os valores de um vetor par um novo vetor
9. Determinar a quantidade de positivos em um vetor
10. Determinar quantos valores são divisíveis por n em um vetor
11. Inverter os valores de um vetor

Dúvidas?!

UNIVALI

Escola Politécnica