Algoritmos e Programação
Operações em Vetores

Prof. Dr. Joaquim Assunção

CENTRO DE TECNOLOGIA
UFSM
2025

Universidade Federal de Santa Maria
SEDES SAPIENTIÆ
1960
CT
CENTRO DE TECNOLOGIA

Operações

- **Busca**
- **Inserção**
- **Ordenação**
- **Remoção**

Busca

- Dado o seguinte vetor

int x[] ← {222, 555, 111, 333, 444, 666, 556, 888, 777, 987, 654}

Faça um algoritmo que receba um vetor e retorne o índice do vetor dado um número qualquer. Adote uma convenção para caso não haja retorno.

Em C++

- Dado o seguinte vetor

int x[] ← {222, 555, 111, 333, 444, 666, 556, 888, 777, 987, 654}

Faça um algoritmo que receba um vetor e retorne o índice do vetor dado um número qualquer. Adote uma convenção para caso não haja retorno.

```cpp
int busca(int num, int x[], int n);

int main() {
    int x[] = {222, 555, 111, 333, 444, 666, 556, 888, 777, 987, 654};
    int n = sizeof(x) / sizeof(x[0]); // number of elements
    busca(444, x, n);
}

int busca(int num, int x[], int n) {
    int k = n - 1;
    while (k >= 0 && x[k] != num) {
        k = k - 1;
    }
    cout << k;
    return k; // return the index (or -1 if not found)
}
```

Inserção

- Dado o seguinte vetor

int x ← {222, 555, 111, 333, 444, 666, 555, 888, 777, 987, 654, -1}

Faça um algoritmo para inserir um valor V na posição P. Todos os valores a frente devem ser deslocados, exceto o -1 que será apagado.

```
//Inserção
void insere(int v, int p);
int x[] = {222, 555, 111, 333, 444, 666, 556, 888, 777, 987, 654, -1};
int tamX = sizeof(x) / sizeof(x[0]); // number of elements


int main() {
    //Complete o código
}


void insere(int v, int p) {
    int k = tamX - 1;
    while (k >= p) {
        x[k] = x[k-1];
        //Complete essa linha
    }
    x[k] = v;
    for (int i = 0; i < tamX; i++){
        cout << x[i] << endl;
    }
}
```

Remoção

- Dado o seguinte vetor

int x ← {222, 555, 111, 333, 444, 666, 555, 888, 777, 987, 654}

Faça um algoritmo que remova o valor da posição P e faça com que todos os demais (a frente) voltem uma posição.

→ Programe em C++

Bogosort

- Também conhecido como Casesort, Permutation Sort, …
- Extremamente ineficiente
- Baseado na ordenação aleatória dos elementos
  - Não é utilizado na prática

Ordenação por Troca

```
para (i <- 0; i < (tam -1); i++)
     para (j <- i+1; j < tam; j++)
          se (vetor[i] > vetor[j])
               aux <- vetor[i];
               vetor[i] <- vetor[j];
               vetor[j] <- aux;
          }
     }
}
```

Índide	Valor
0	2
1	7
2	5
3	2
4	4

→ **Programe em C++**

Exercícios

- Elabore um programa que leia 9 inteiros e imprima-os em ordem crescente.
- Construir um algoritmo que leia um vetor A com 15 elementos numéricos inteiros. Construa um vetor B de mesmo tipo, em que cada elemento seja o fatorial do elemento correspondente armazenado em A. Apresentar os valores do vetor B em ordem crescente.