COMPUTABILIDADE E COMPLEXIDADE DE ALGORITMOS
Aula 1

Profa. Kátia Alves Bezerra

kbezerra@cruzeirodosul.edu.br

Agradecimentos: Prof. Fabio Cosme

INTRODUÇÃO

- **Definições**

- Algoritmos são projetados para resolver problemas:
  - **Problema:** encontrar a melhor rota, em termos de tempo de entrega, dos produtos das Casas Ceará em Ermelino Matarazzo;
  - **Solução:** algoritmo para descoberta da melhor rota tendo como entrada os locais de entrega.

INTRODUÇÃO

- **Definições**

- Tudo pode ser resolvido por um algoritmo?

INTRODUÇÃO

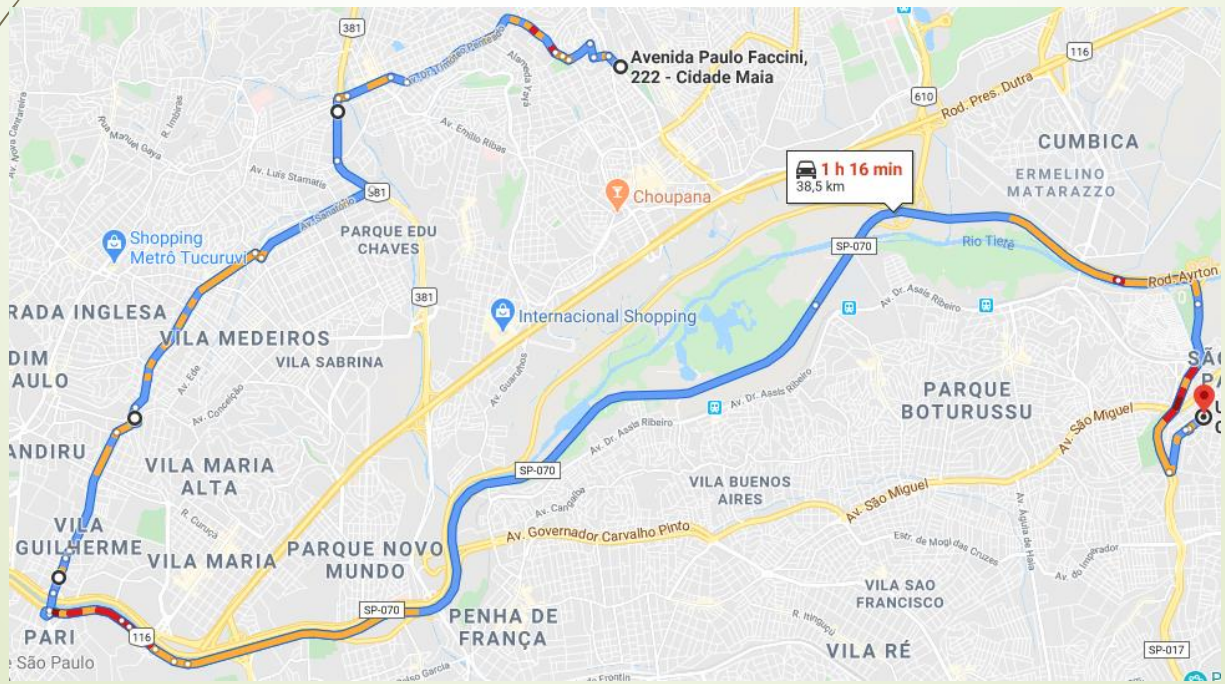
- **Definições**

- Tudo o que é resolvido por um algoritmo é aceitável?

INTRODUÇÃO

- **Definições**

- Tudo o que é resolvido por um algoritmo é aceitável?



INTRODUÇÃO

- **Definições**

- Segundo **Dijkstra**, um algoritmo corresponde a uma **descrição de um padrão de comportamento**, expresso em termos de um **conjunto finito de ações**.

- Segundo **Terada**, um algoritmo é, em geral, uma **descrição passo a passo de como um problema é solucionável**. A descrição deve ser finita, e os passos devem ser **bem definidos**, sem ambiguidades, e executáveis computacionalmente.

INTRODUÇÃO

- **Exemplos**

- Considere os algoritmos a seguir.
  - Ambos recebem um valor n que é um inteiro positivo;
  - Se receberem um mesmo valor de n, ambos devolvem o mesmo valor (x);

```
1
2   function somaQuadradosA(n)
3   {
4       x = 0;
5       for(j=1; j<=n;j++)
6           x = x+(j*j);
7       return x
8   }
9
```

```
10  function somaQuadradosB(n)
11  {
12      x = n*(n+1)*(2*n+1);
13      x = x/6;
14      return x;
15  }
16
```

INTRODUÇÃO

- **Exemplos**

- Quantas operações aritméticas o primeiro algoritmo faz? Quantas operações aritméticas o segundo algoritmo faz? Qual dos dois algoritmos é mais eficiente?

```
1
2   function somaQuadradosA(n)
3   {
4       x = 0;
5       for(j=1; j<=n;j++)
6           x = x+(j*j);
7       return x
8   }
9
```

```
10  function somaQuadradosB(n)
11  {
12      x = n*(n+1)*(2*n+1);
13      x = x/6;
14      return x;
15  }
16
```

INTRODUÇÃO

- **Exemplos**

- Problema **vs** Instância

Preciso urgentemente rearranjar um vetor A[1...n] em ordem crescente:

Valores de entrada: 85 99 64 24 97 25 89 33

INTRODUÇÃO

- **Exemplos**

- Problema **vs** Instância

Preciso urgentemente **rearranjar um vetor** A[1...n] em ordem crescente:

Valores de entrada: 85 99 64 24 97 25 89 33

COMPUTABILIDADE E COMPLEXIDADE DE ALGORITMOS

- **OBRIGADO !!**