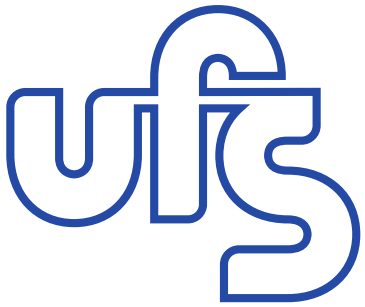
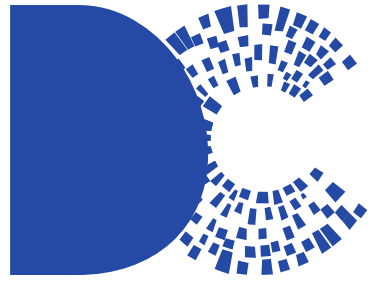


UNIVERSIDADE FEDERAL DE SERGIPE



DEPARTAMENTO DE COMPUTAÇÃO

# Introdução de algoritmos e técnicas de projeto

## Projeto e Análise de Algoritmos

Bruno Prado

Departamento de Computação / UFS

# Introdução

- O que é um algoritmo?
  - Passos não ambíguos

# Introdução

- O que é um algoritmo?
  - Passos não ambíguos
  - Entradas e saídas esperadas

# Introdução

- O que é um algoritmo?
  - Passos não ambíguos
  - Entradas e saídas esperadas
  - Resolução dinâmica de um problema em um determinado espaço de tempo

# Introdução

- Exemplo do cotidiano: receita de bolo
  - Passos = instruções da receita (algoritmo)

# Introdução

- Exemplo do cotidiano: receita de bolo
  - Passos = instruções da receita (algoritmo)
  - Entradas = ingredientes

# Introdução

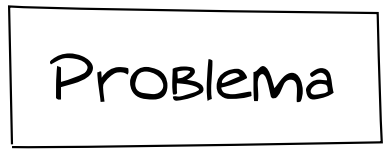
- Exemplo do cotidiano: receita de bolo
  - Passos = instruções da receita (algoritmo)
  - Entradas = ingredientes
  - Saída = bolo

# Introdução

- Exemplo do cotidiano: receita de bolo
  - Passos = instruções da receita (algoritmo)
  - Entradas = ingredientes
  - Saída = bolo
  - Tempo = preparo de 40 minutos

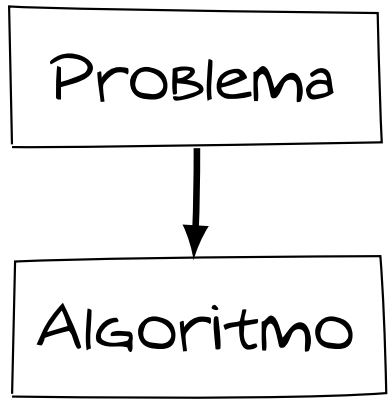
# Introdução

- Algoritmos na Computação



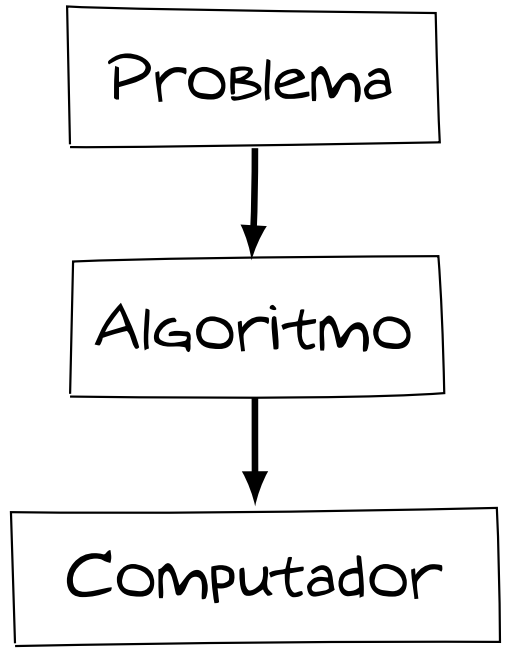
# Introdução

- Algoritmos na Computação



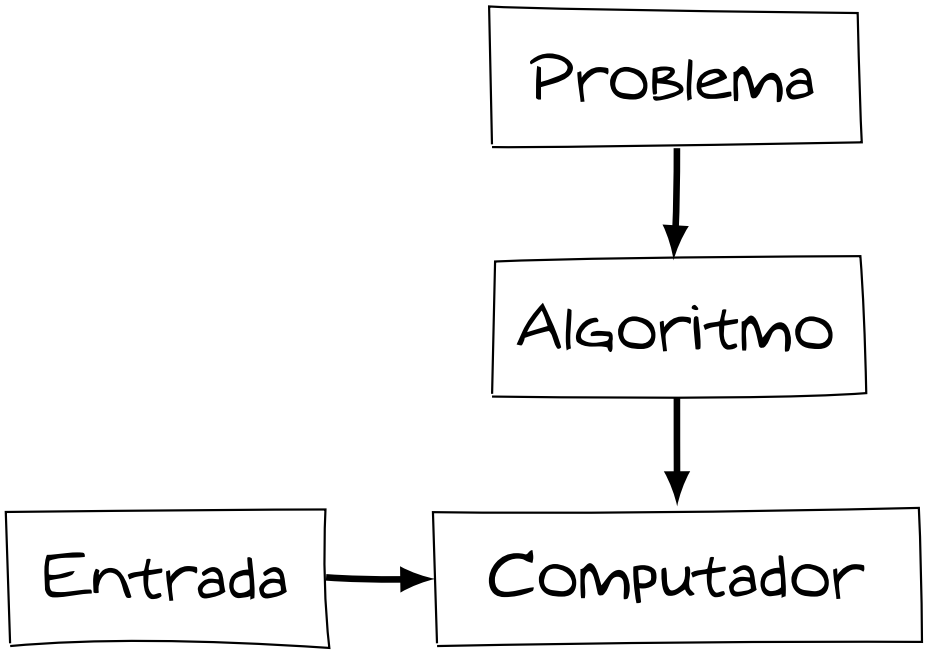
# Introdução

- Algoritmos na Computação



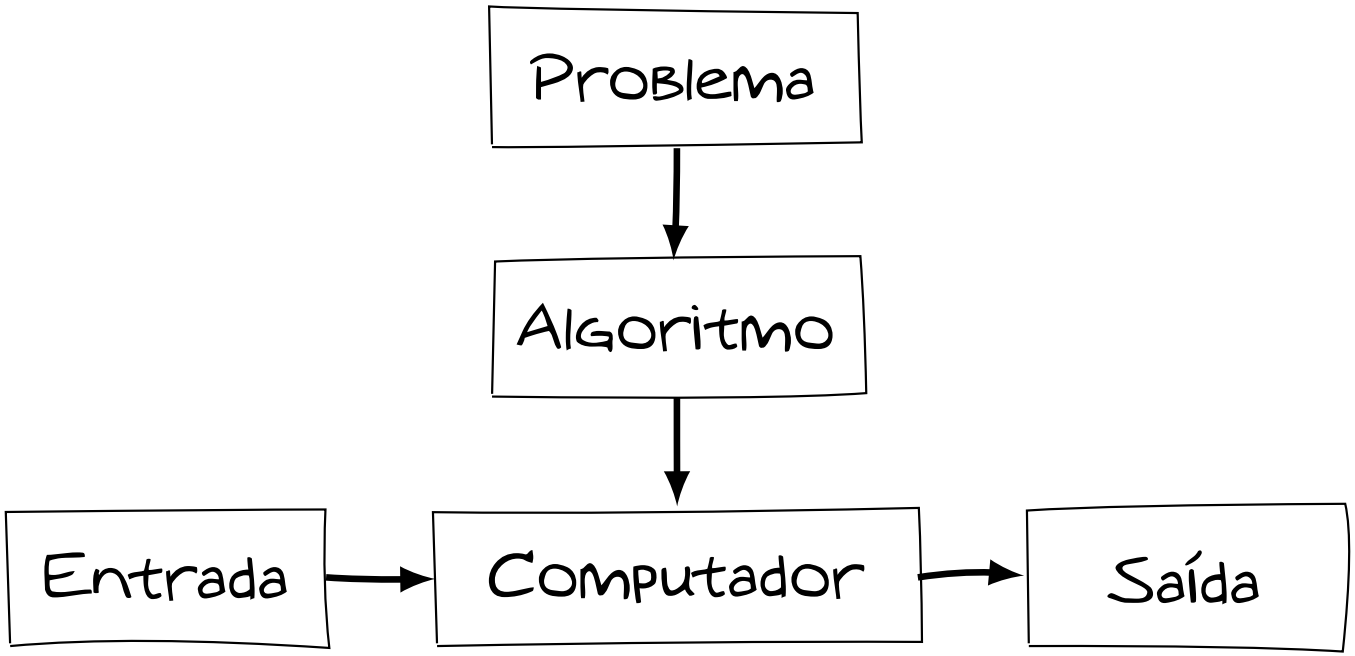
# Introdução

- Algoritmos na Computação



# Introdução

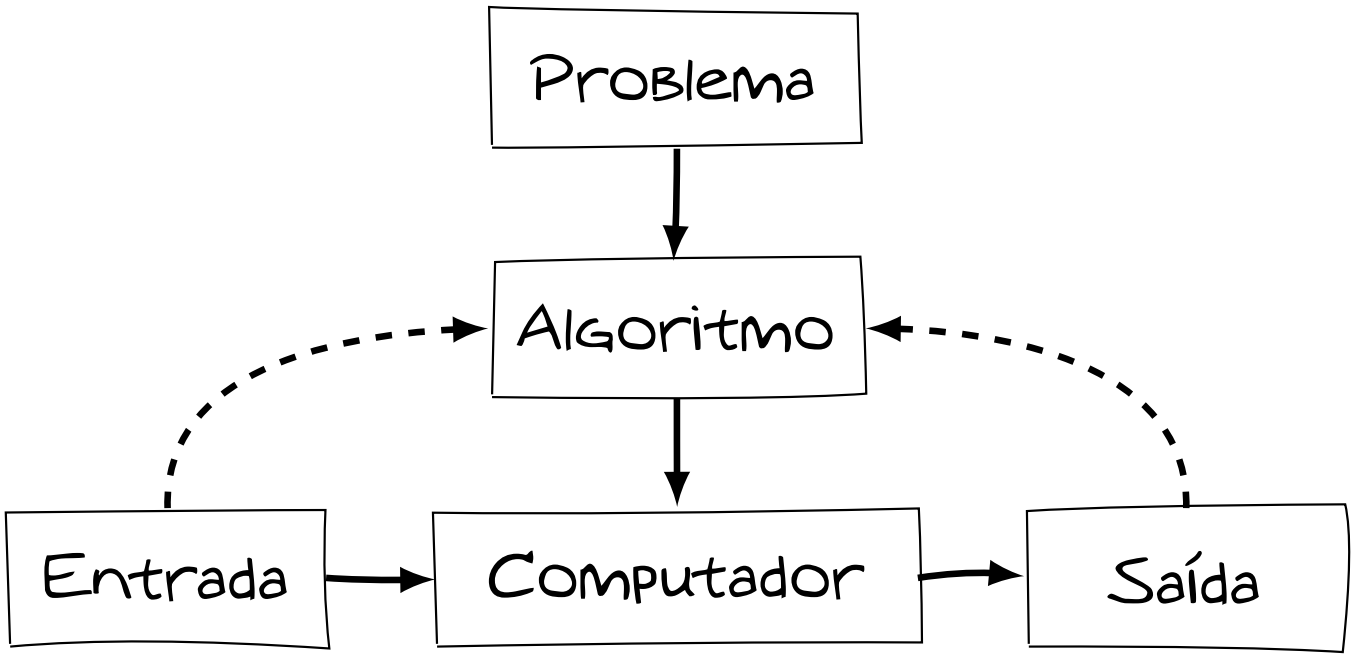
- Algoritmos na Computação




Abordagem tradicional

# Introdução

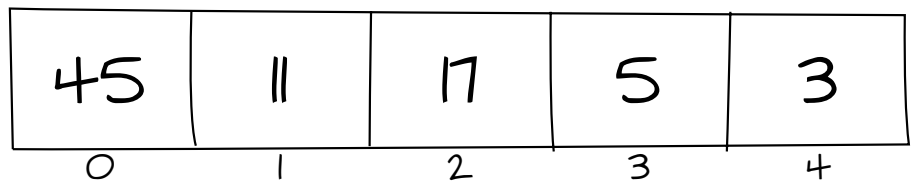
- Algoritmos na Computação

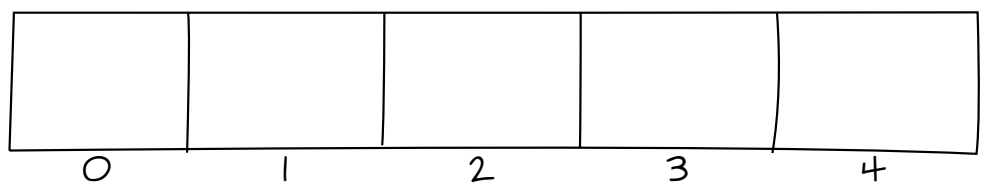



Aprendizado de máquina

# Introdução

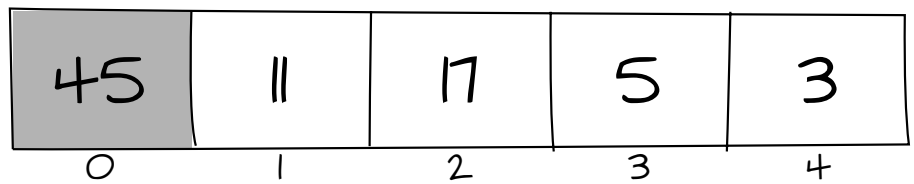
- Problema de ordenação
  - Entrada: sequência de $n$ números $E_1, E_2, \ldots, E_n$
  - Saída: sequência de $n$ números $E'_1, E'_2, \ldots, E'_n$, onde $E'_1 \leq E'_2 \leq \ldots \leq E'_n$
  - Algoritmo de ordenação simples

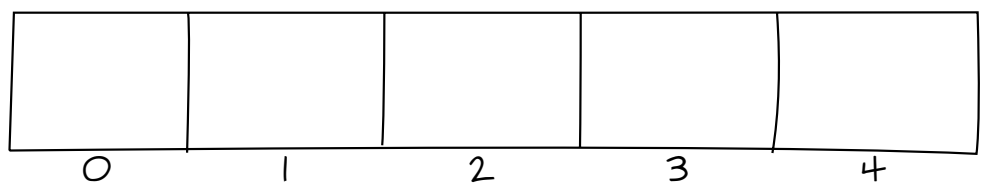


# Introdução

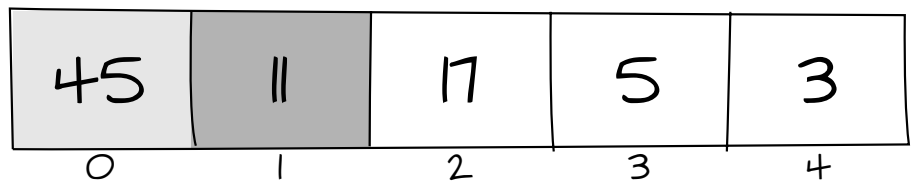
- Problema de ordenação
  - Entrada: sequência de $n$ números $E_1, E_2, \ldots, E_n$
  - Saída: sequência de $n$ números $E'_1, E'_2, \ldots, E'_n$, onde $E'_1 \leq E'_2 \leq \ldots \leq E'_n$
  - Algoritmo de ordenação simples

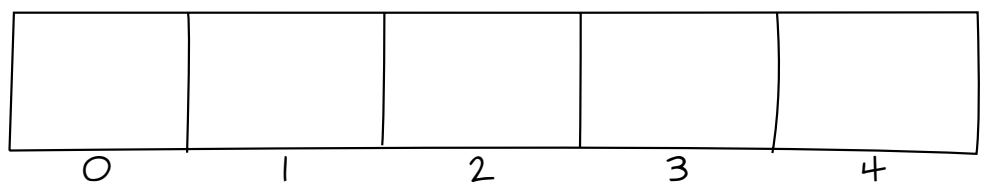


# Introdução

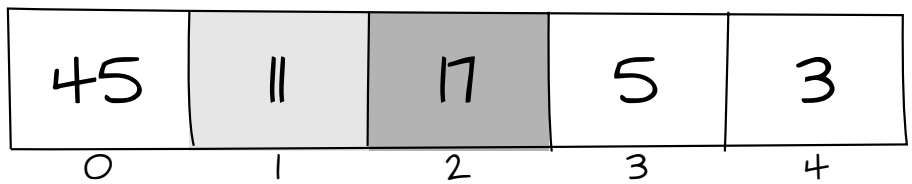
- Problema de ordenação
  - Entrada: sequência de $n$ números $E_1, E_2, \ldots, E_n$
  - Saída: sequência de $n$ números $E'_1, E'_2, \ldots, E'_n$, onde $E'_1 \leq E'_2 \leq \ldots \leq E'_n$
  - Algoritmo de ordenação simples

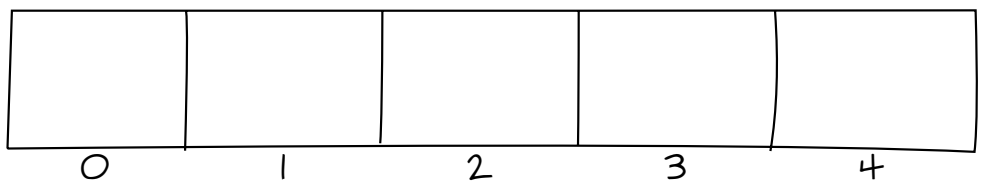


# Introdução

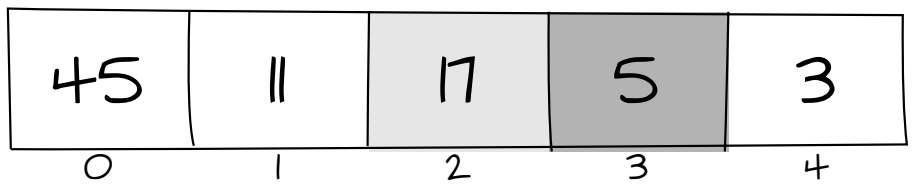
- Problema de ordenação
  - Entrada: sequência de $n$ números $E_1, E_2, \ldots, E_n$
  - Saída: sequência de $n$ números $E'_1, E'_2, \ldots, E'_n$, onde $E'_1 \leq E'_2 \leq \ldots \leq E'_n$
  - Algoritmo de ordenação simples

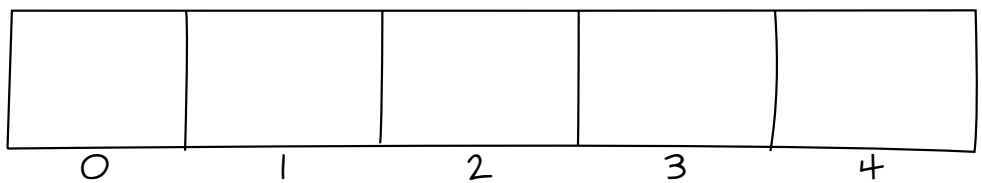


# Introdução

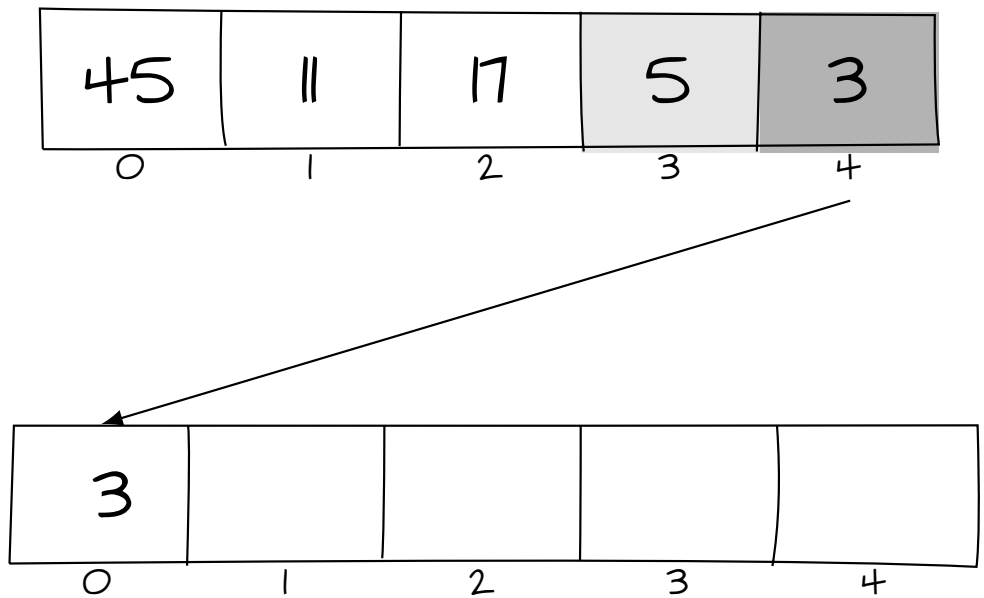
- Problema de ordenação
  - Entrada: sequência de $n$ números $E_1, E_2, \ldots, E_n$
  - Saída: sequência de $n$ números $E'_1, E'_2, \ldots, E'_n$, onde $E'_1 \leq E'_2 \leq \ldots \leq E'_n$
  - Algoritmo de ordenação simples



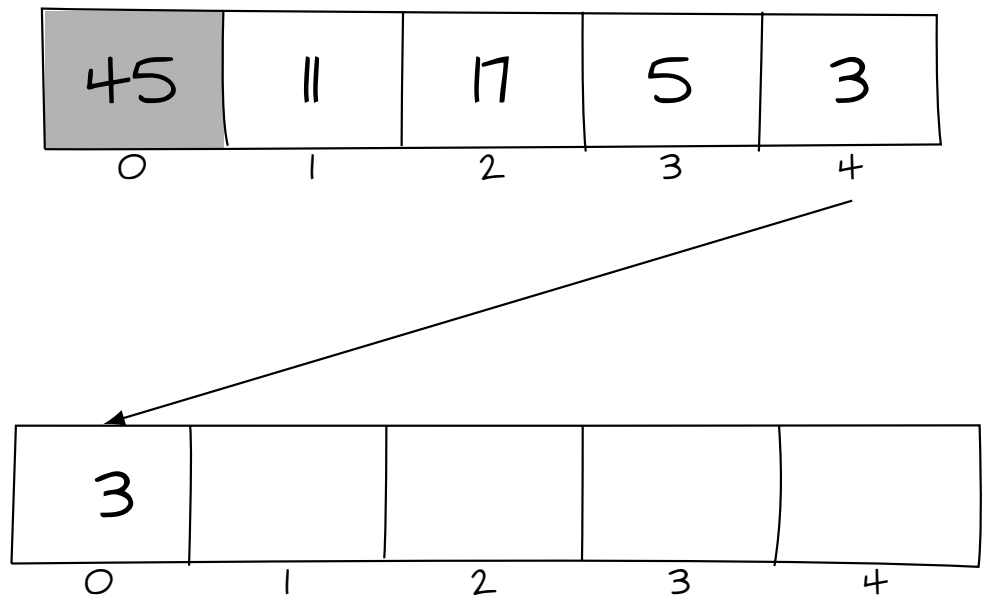
# Introdução

- Problema de ordenação
  - Entrada: sequência de $n$ números $E_1, E_2, \ldots, E_n$
  - Saída: sequência de $n$ números $E'_1, E'_2, \ldots, E'_n$, onde $E'_1 \leq E'_2 \leq \ldots \leq E'_n$
  - Algoritmo de ordenação simples



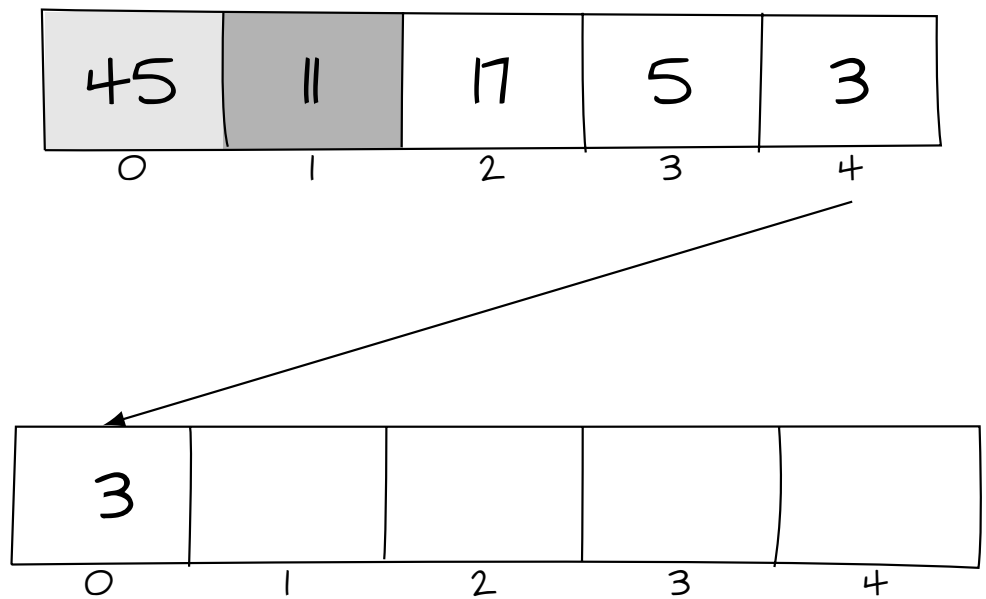
# Introdução

- Problema de ordenação
  - Entrada: sequência de $n$ números $E_1, E_2, \ldots, E_n$
  - Saída: sequência de $n$ números $E'_1, E'_2, \ldots, E'_n$, onde $E'_1 \leq E'_2 \leq \ldots \leq E'_n$
  - Algoritmo de ordenação simples



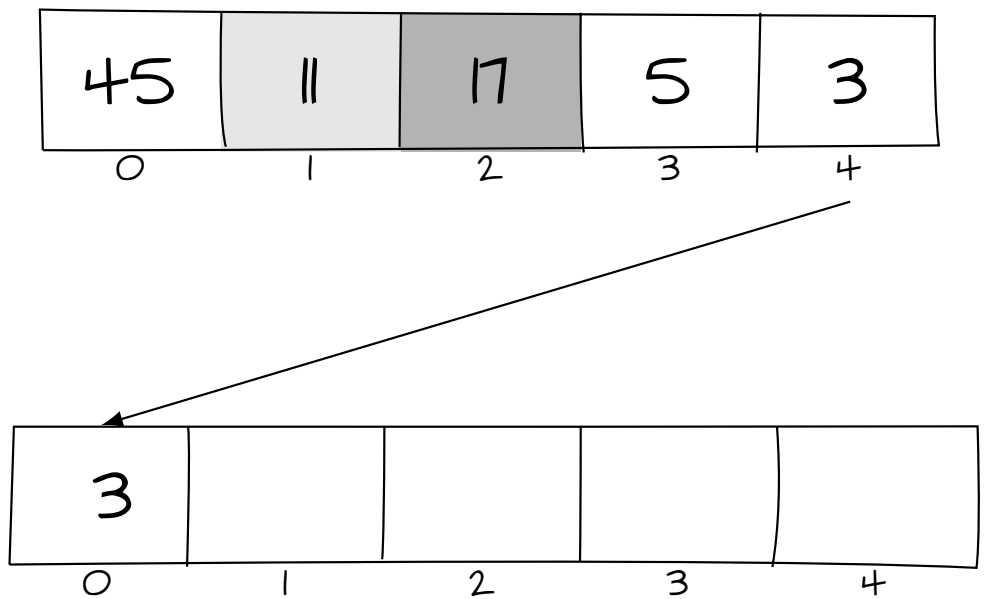
# Introdução

- Problema de ordenação
  - Entrada: sequência de $n$ números $E_1, E_2, \ldots, E_n$
  - Saída: sequência de $n$ números $E'_1, E'_2, \ldots, E'_n$, onde $E'_1 \leq E'_2 \leq \ldots \leq E'_n$
  - Algoritmo de ordenação simples



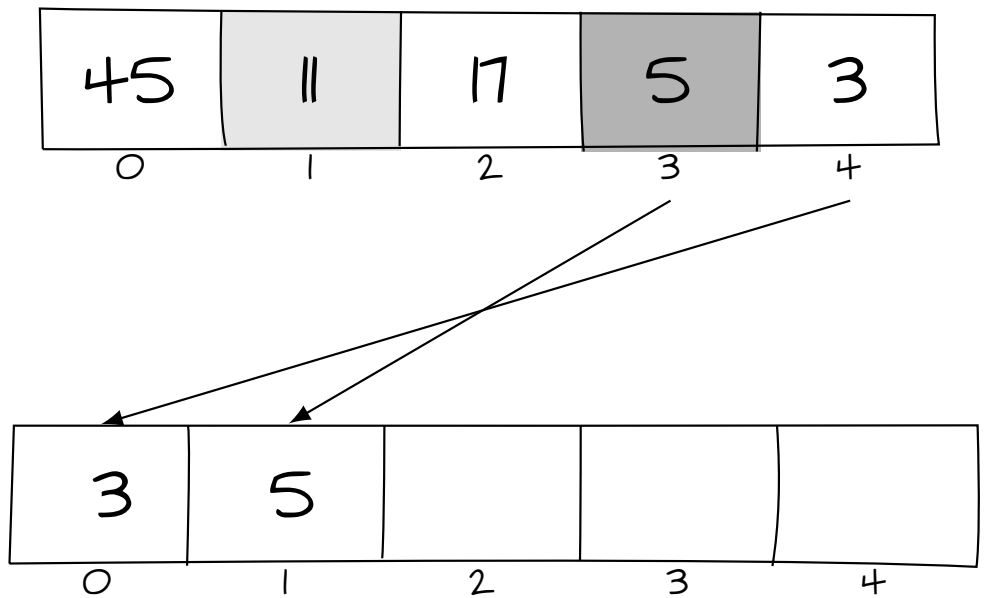
# Introdução

- Problema de ordenação
  - Entrada: sequência de $n$ números $E_1, E_2, \ldots, E_n$
  - Saída: sequência de $n$ números $E'_1, E'_2, \ldots, E'_n$, onde $E'_1 \leq E'_2 \leq \ldots \leq E'_n$
  - Algoritmo de ordenação simples



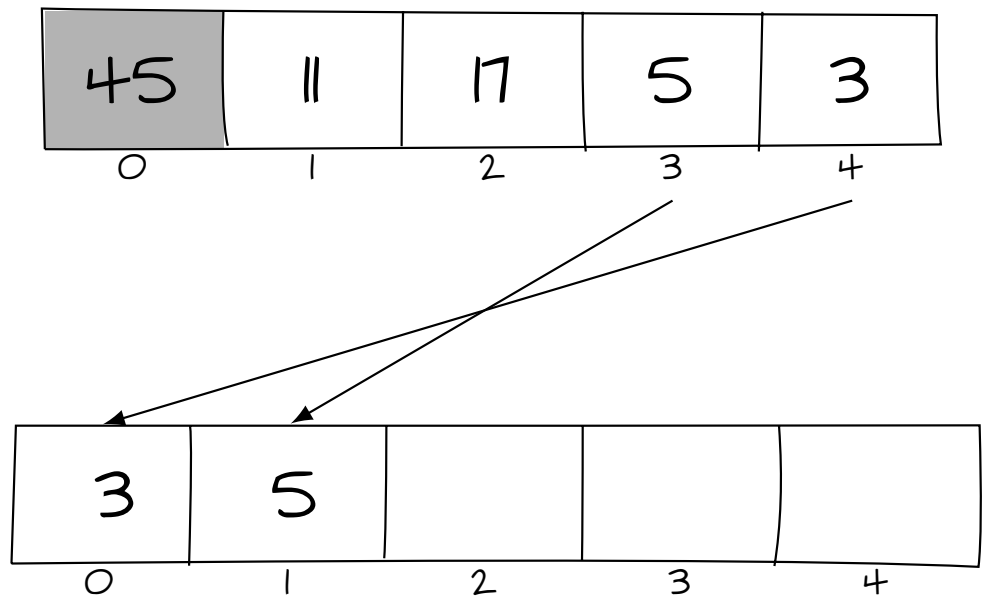
# Introdução

- Problema de ordenação
  - Entrada: sequência de $n$ números $E_1, E_2, \ldots, E_n$
  - Saída: sequência de $n$ números $E'_1, E'_2, \ldots, E'_n$, onde $E'_1 \leq E'_2 \leq \ldots \leq E'_n$
  - Algoritmo de ordenação simples



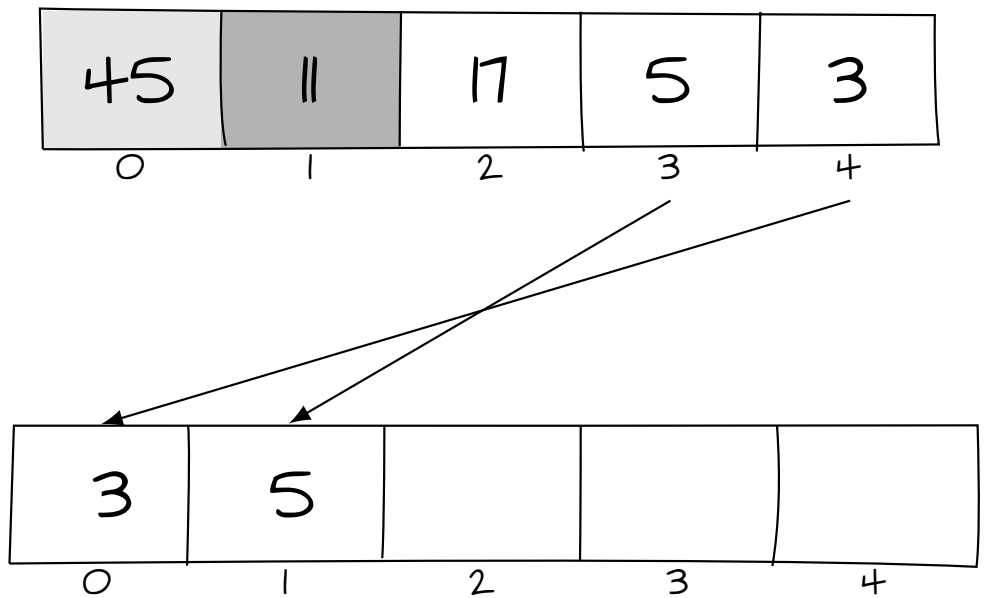
# Introdução

- Problema de ordenação
  - Entrada: sequência de $n$ números $E_1, E_2, \ldots, E_n$
  - Saída: sequência de $n$ números $E'_1, E'_2, \ldots, E'_n$, onde $E'_1 \leq E'_2 \leq \ldots \leq E'_n$
  - Algoritmo de ordenação simples



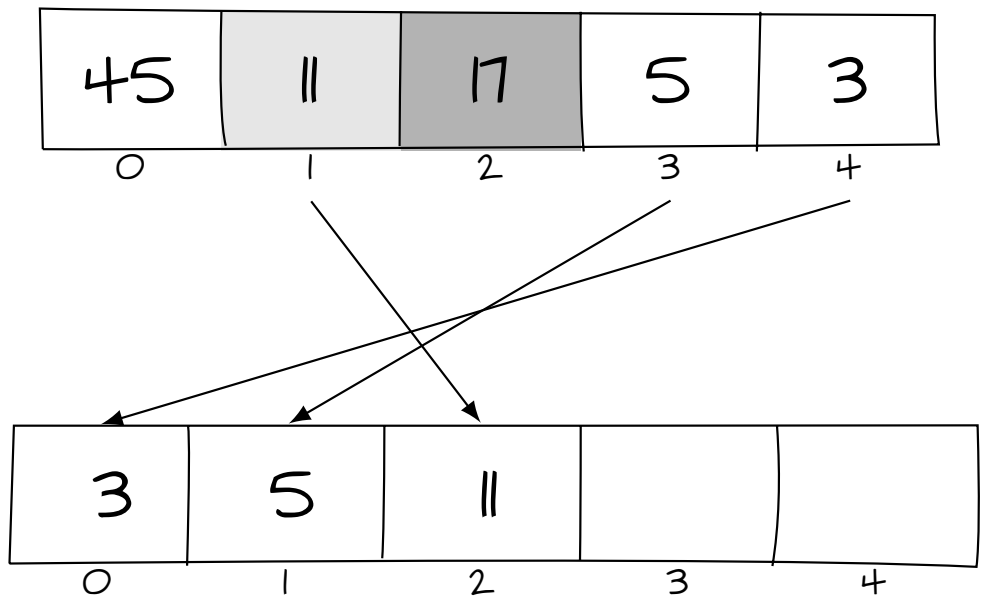
# Introdução

- Problema de ordenação
  - Entrada: sequência de $n$ números $E_1, E_2, \ldots, E_n$
  - Saída: sequência de $n$ números $E'_1, E'_2, \ldots, E'_n$, onde $E'_1 \leq E'_2 \leq \ldots \leq E'_n$
  - Algoritmo de ordenação simples



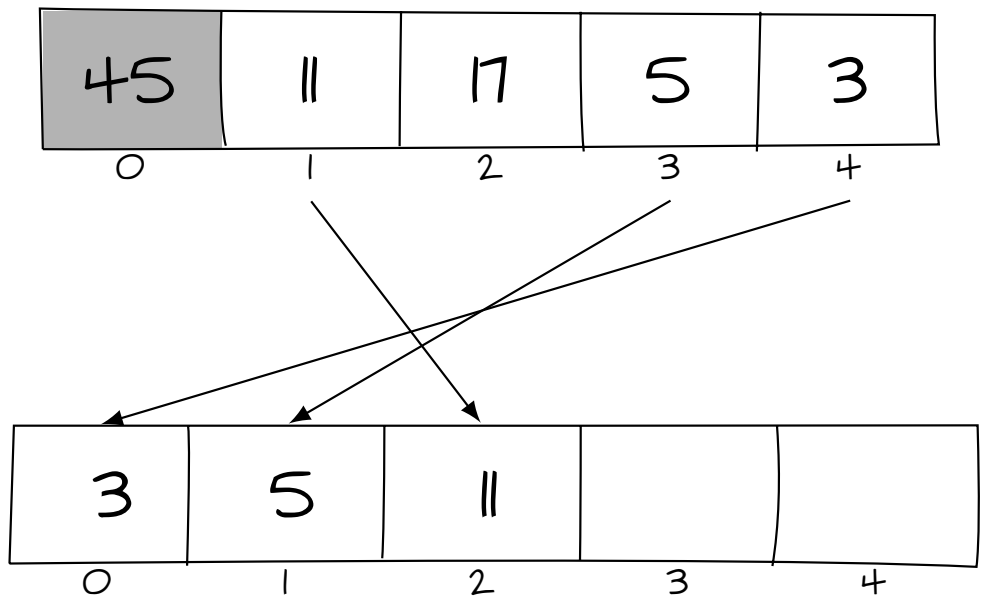
# Introdução

- Problema de ordenação
  - Entrada: sequência de $n$ números $E_1, E_2, \ldots, E_n$
  - Saída: sequência de $n$ números $E'_1, E'_2, \ldots, E'_n$, onde $E'_1 \leq E'_2 \leq \ldots \leq E'_n$
  - Algoritmo de ordenação simples



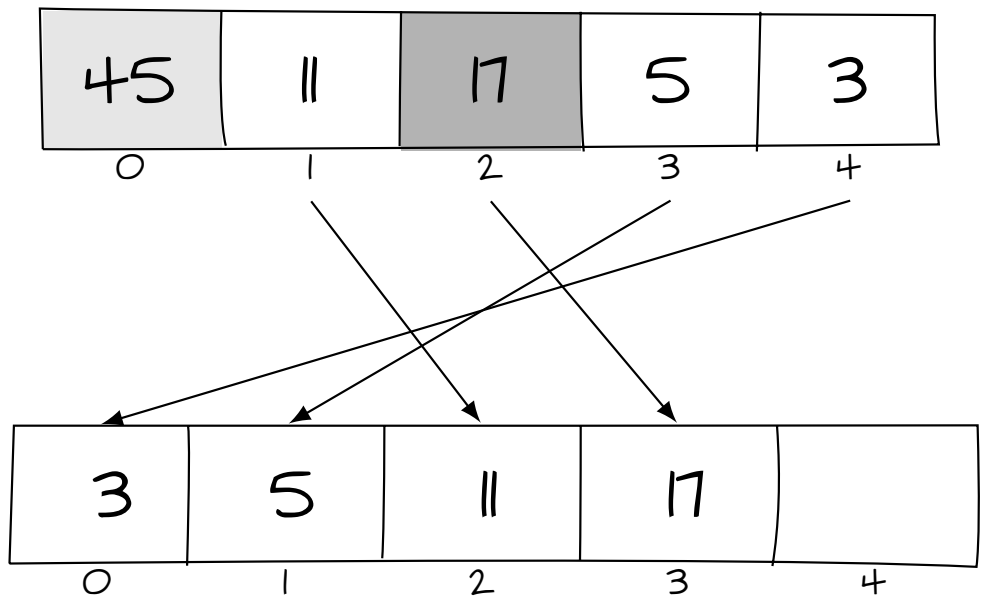
# Introdução

- Problema de ordenação
  - Entrada: sequência de $n$ números $E_1, E_2, \ldots, E_n$
  - Saída: sequência de $n$ números $E'_1, E'_2, \ldots, E'_n$, onde $E'_1 \leq E'_2 \leq \ldots \leq E'_n$
  - Algoritmo de ordenação simples



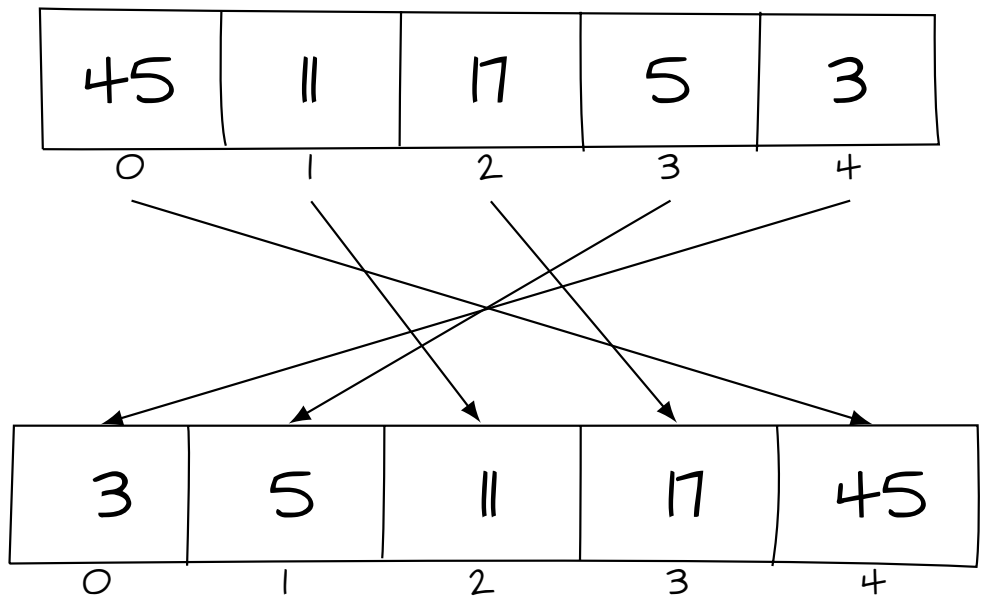
# Introdução

- Problema de ordenação
  - Entrada: sequência de $n$ números $E_1, E_2, \ldots, E_n$
  - Saída: sequência de $n$ números $E'_1, E'_2, \ldots, E'_n$, onde $E'_1 \leq E'_2 \leq \ldots \leq E'_n$
  - Algoritmo de ordenação simples

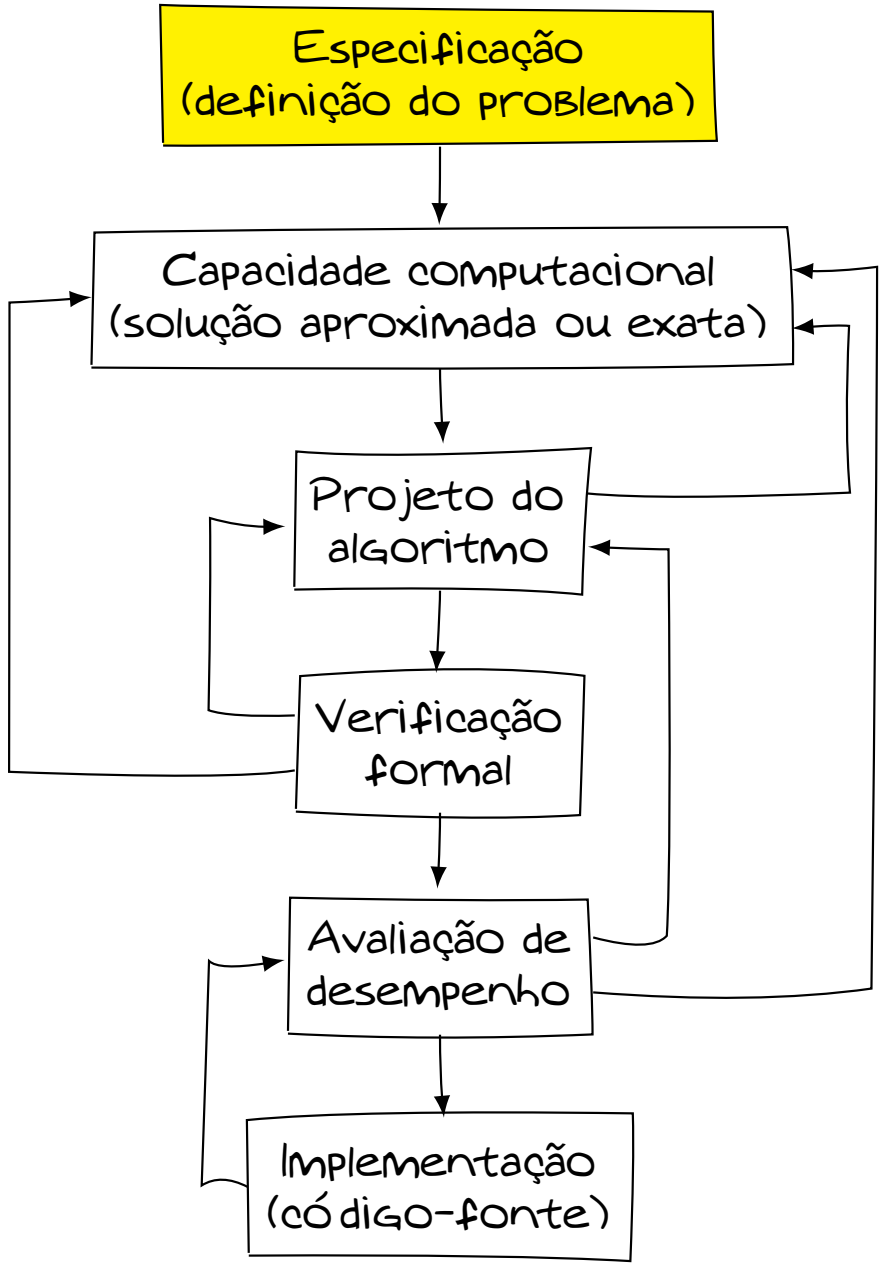


# Introdução

- Problema de ordenação
  - Entrada: sequência de $n$ números $E_1, E_2, \ldots, E_n$
  - Saída: sequência de $n$ números $E'_1, E'_2, \ldots, E'_n$, onde $E'_1 \leq E'_2 \leq \ldots \leq E'_n$
  - Algoritmo de ordenação simples



# Fluxo de desenvolvimento



# Fluxo de desenvolvimento

- Especificação (definição do problema)
  - Capturar os requisitos funcionais e não funcionais
    - Funcionais = comportamentos
    - Não funcionais = restrições

# Fluxo de desenvolvimento

- Especificação (definição do problema)
  - Capturar os requisitos funcionais e não funcionais
    - Funcionais = comportamentos
    - Não funcionais = restrições
  - Realizar experimentos e construir protótipos

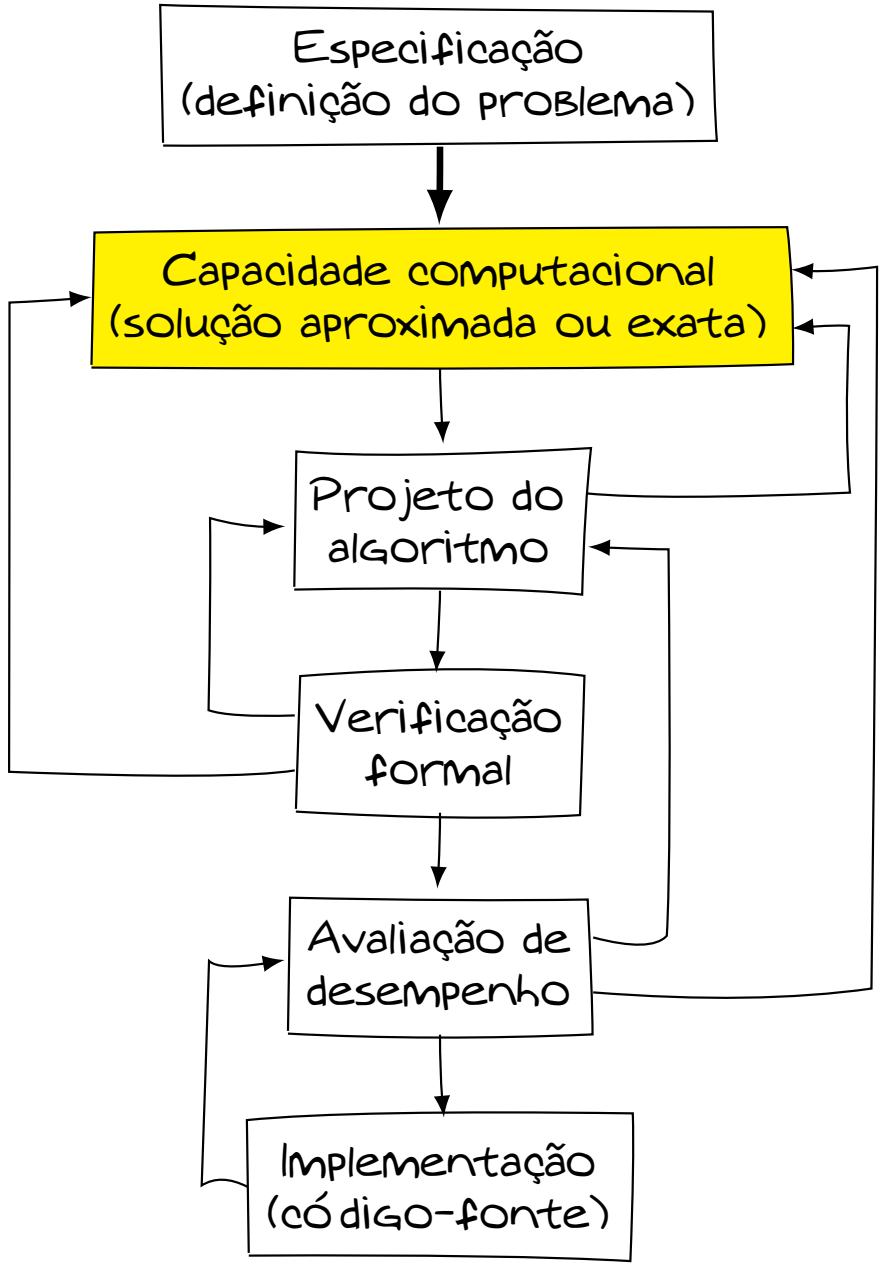
# Fluxo de desenvolvimento

- Especificação (definição do problema)
  - Capturar os requisitos funcionais e não funcionais
    - Funcionais = comportamentos
    - Não funcionais = restrições
  - Realizar experimentos e construir protótipos
  - Delimitar o escopo da solução

# Fluxo de desenvolvimento

- Especificação (definição do problema)
  - Requisito funcional
    - Ordenar números em ordem crescente
  - Requisito não funcional
    - Números inteiros positivos
  - Escopo
    - Números inteiros de 32 bits
    - Sequências de até 1.000.000 números

# Fluxo de desenvolvimento



# Fluxo de desenvolvimento

- Capacidade computacional
  - Qual é o poder de processamento?

# Fluxo de desenvolvimento

- Capacidade computacional
  - Qual é o poder de processamento?
  - Quanto de armazenamento será necessário?

# Fluxo de desenvolvimento

- Capacidade computacional
  - Qual é o poder de processamento?
  - Quanto de armazenamento será necessário?
  - Fluxo de execução sequencial ou paralelo

# Fluxo de desenvolvimento

- Capacidade computacional
  - Limite de execução de 3 segundos
  - 1.000.000 números de 32 bits = 4 MiB em memória
  - 4 núcleos de processamento

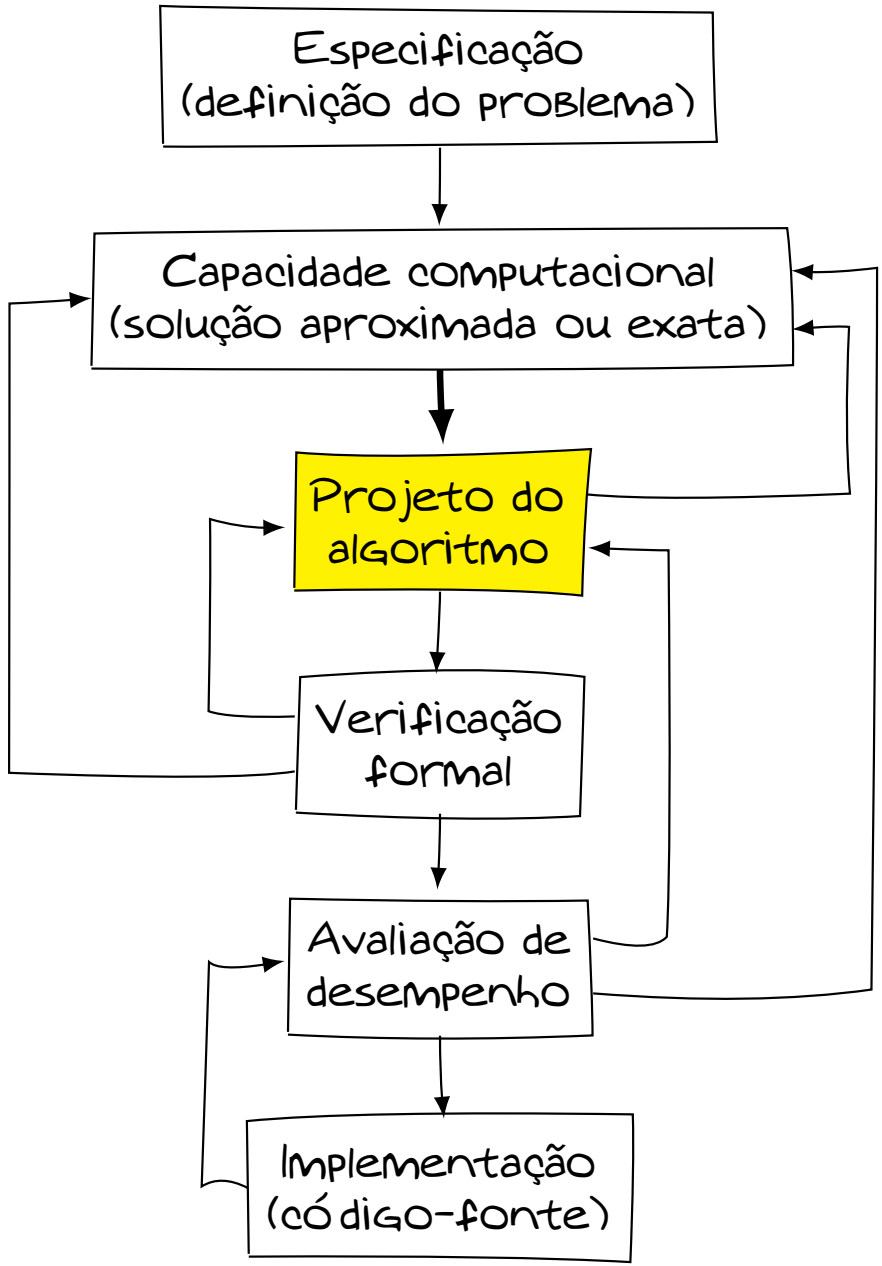
# Fluxo de desenvolvimento

- Capacidade computacional
  - Solução aproximada
    - Baseada em heurística ou estatística
    - Tempo de processamento reduzido
    - Única opção para alguns problemas

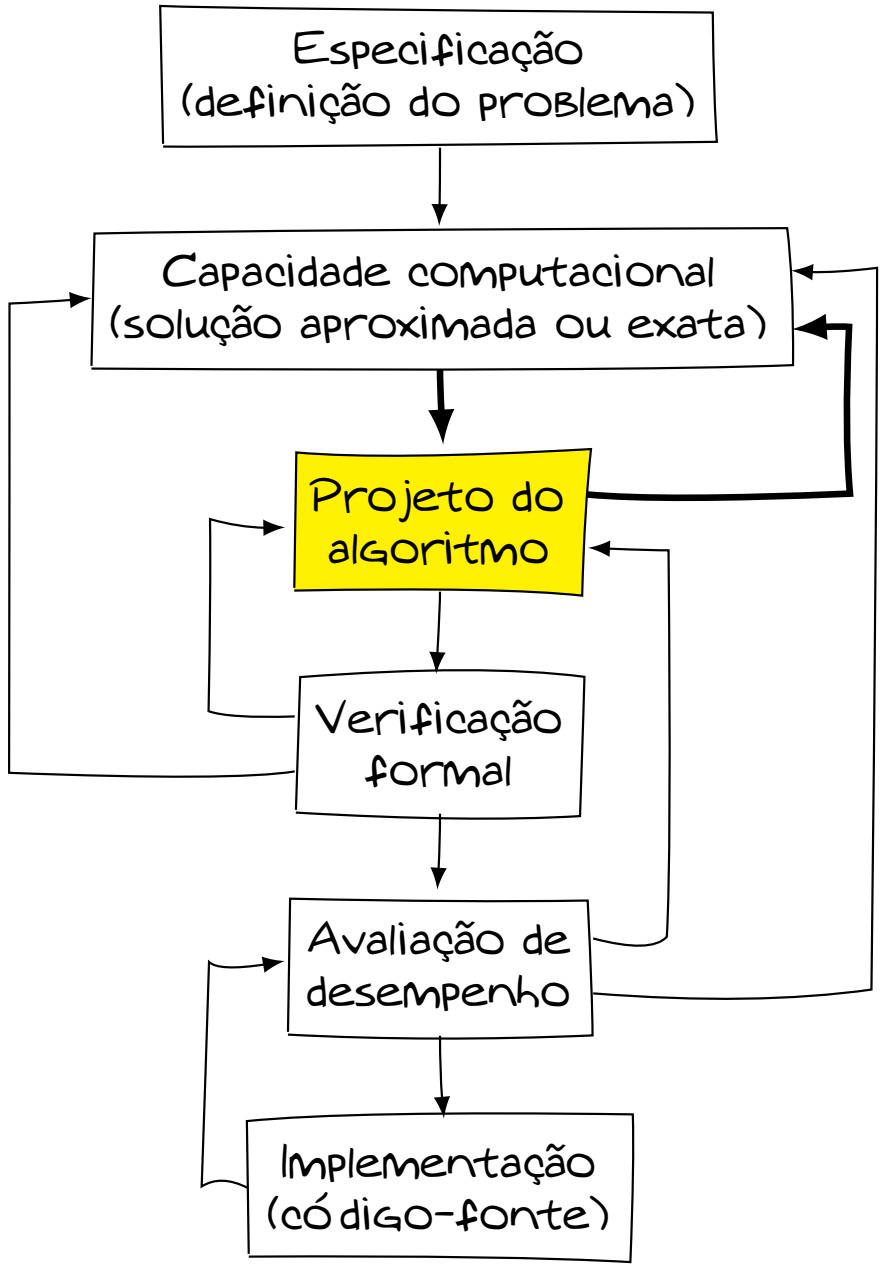
# Fluxo de desenvolvimento

- Capacidade computacional
  - Solução aproximada
    - Baseada em heurística ou estatística
    - Tempo de processamento reduzido
    - Única opção para alguns problemas
  - Solução determinística (exata)
    - Saída e tempo de execução bem definidos
    - Pode ser muito custosa
    - Nem todos os problemas tem solução exata

# Fluxo de desenvolvimento



# Fluxo de desenvolvimento



# Fluxo de desenvolvimento

- Projeto do algoritmo
  - Definir as estratégias, paradigmas e técnicas
    - Iteração ou recursão?
    - Dividir para conquistar
    - Programação dinâmica

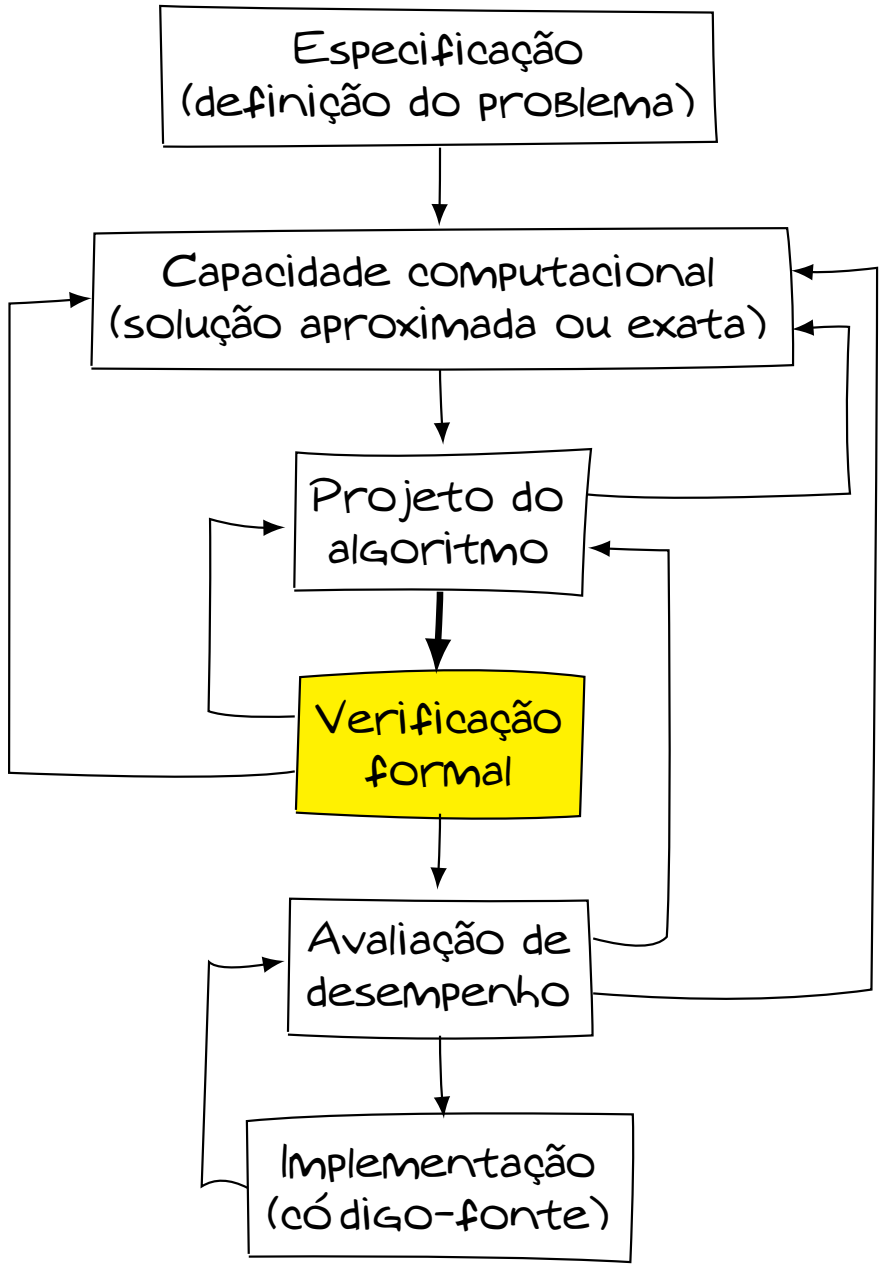
# Fluxo de desenvolvimento

- Projeto do algoritmo
  - Definir as estratégias, paradigmas e técnicas
    - Iteração ou recursão?
    - Dividir para conquistar
    - Programação dinâmica
  - Escolher as estruturas de dados mais adequadas
    - Árvores
    - Listas
    - Vetores
    - ...

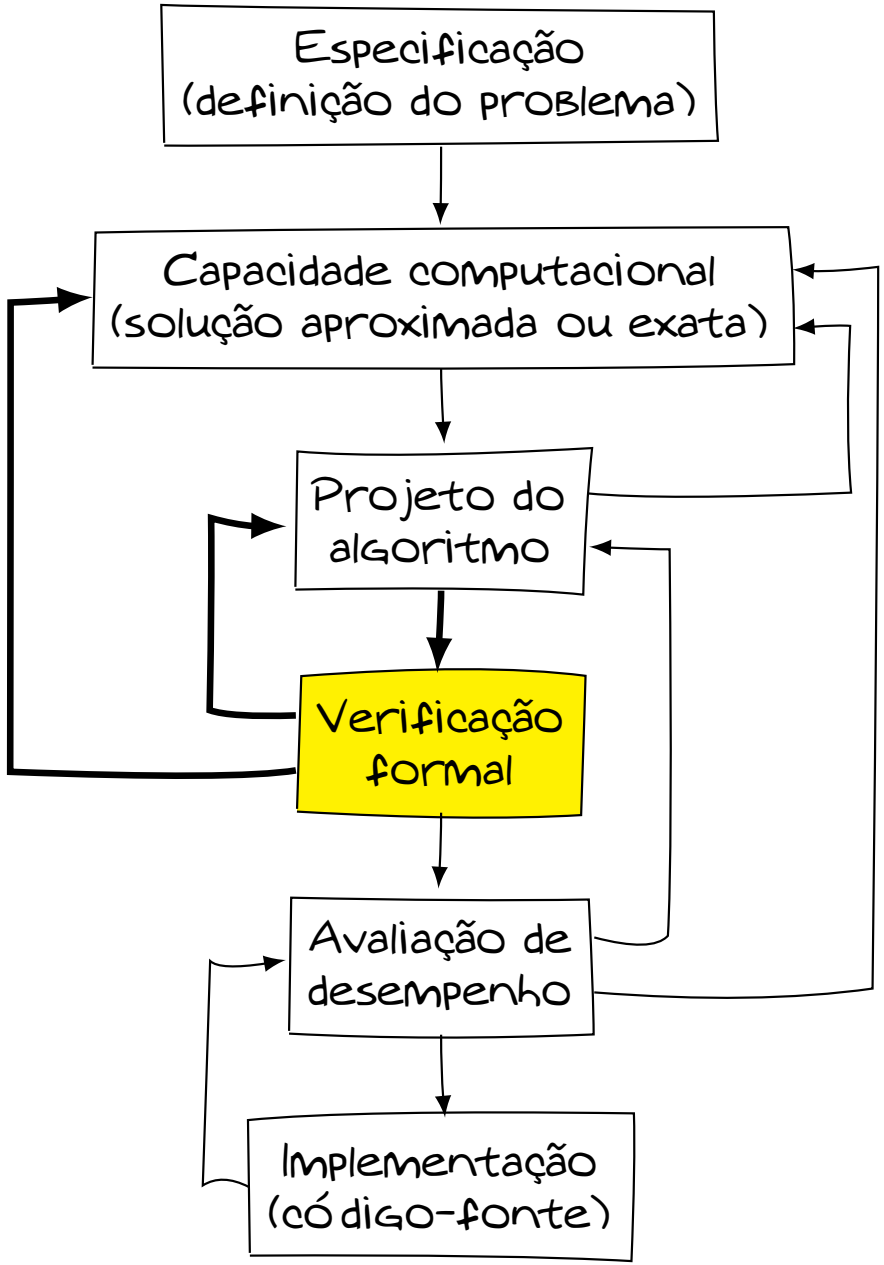
# Fluxo de desenvolvimento

- Projeto do algoritmo
  - Definir as estratégias, paradigmas e técnicas
    - Iteração ou recursão?
    - Dividir para conquistar
    - Programação dinâmica
  - Escolher as estruturas de dados mais adequadas
    - Árvores
    - Listas
    - Vetores
    - ...
  - Descrição pela utilização de pseudocódigo ou visualização através diagramas de fluxo

# Fluxo de desenvolvimento



# Fluxo de desenvolvimento



# Fluxo de desenvolvimento

- Verificação formal
  - Prova matemática que o algoritmo está correto

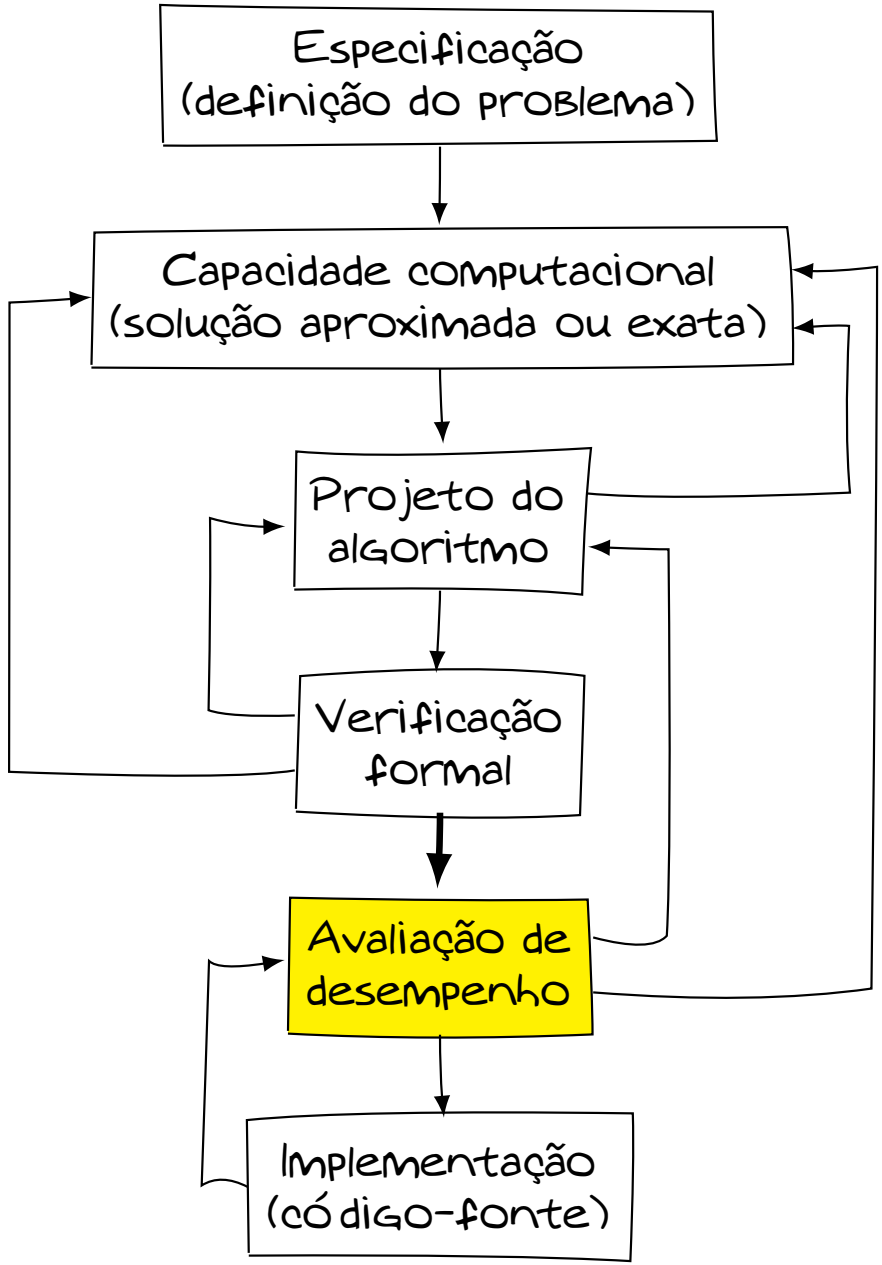
# Fluxo de desenvolvimento

- Verificação formal
  - Prova matemática que o algoritmo está correto
  - Pode ser extremamente complexa em alguns casos

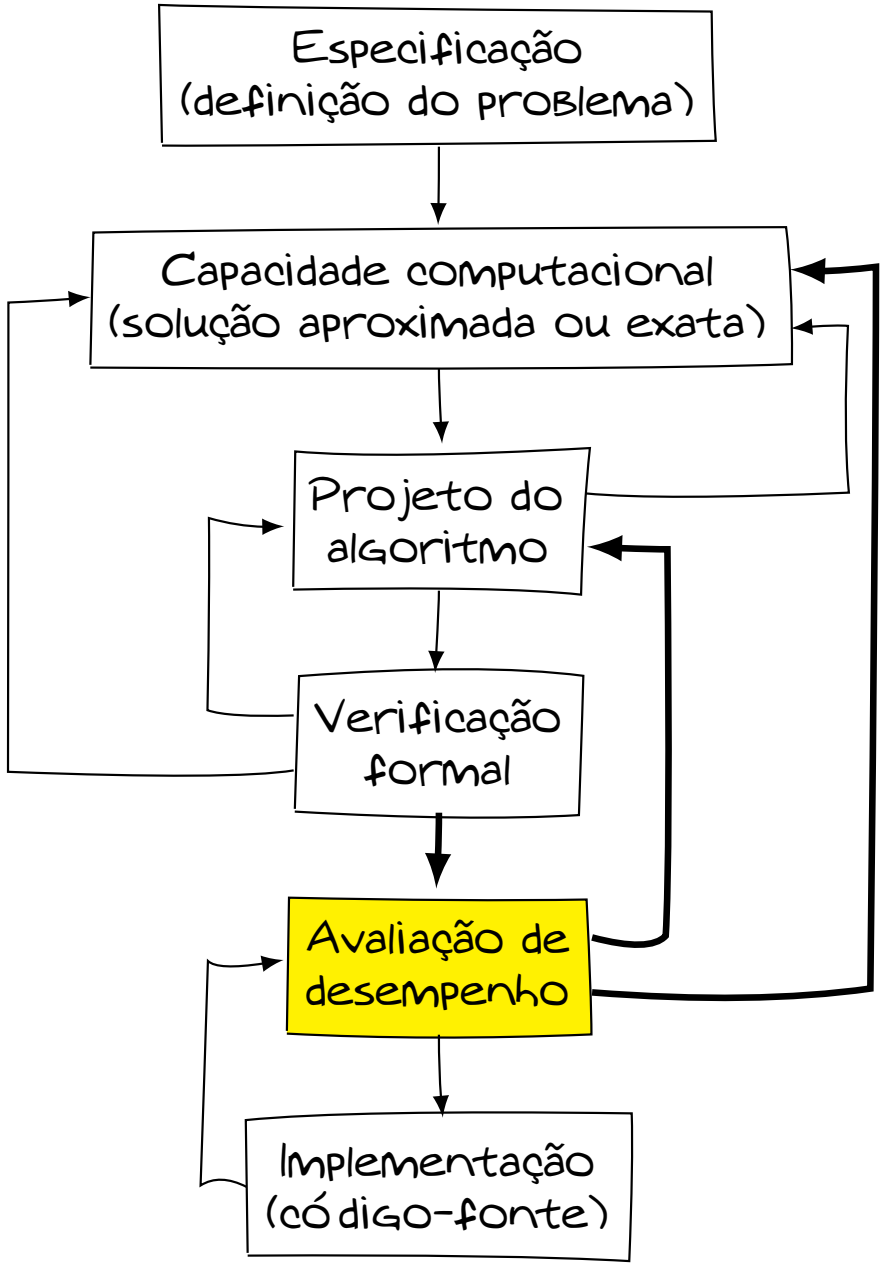
# Fluxo de desenvolvimento

- Verificação formal
  - Prova matemática que o algoritmo está correto
  - Pode ser extremamente complexa em alguns casos
  - Para algoritmos aproximados é medido o erro

# Fluxo de desenvolvimento



# Fluxo de desenvolvimento



# Fluxo de desenvolvimento

- Avaliação de desempenho
  - Eficiência de espaço e de tempo

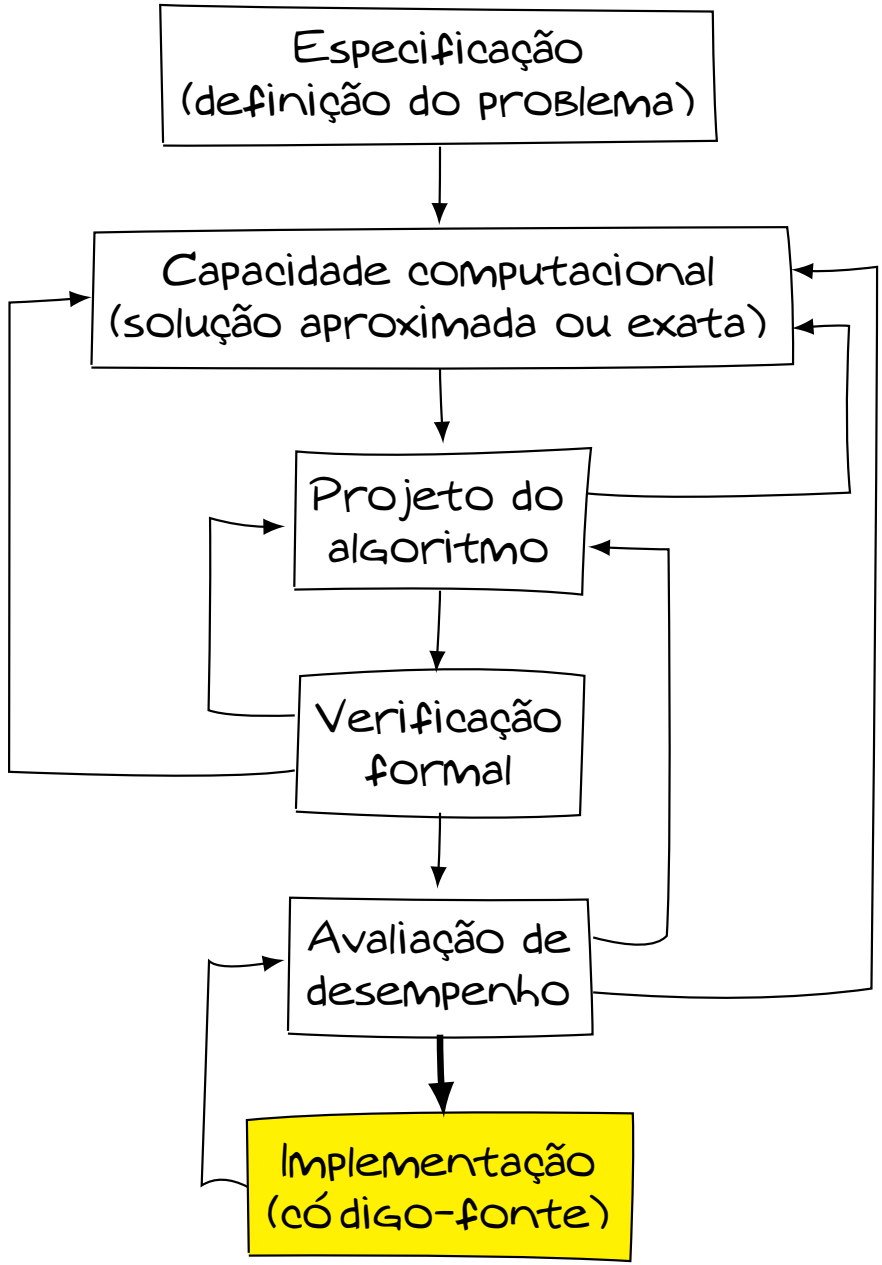
# Fluxo de desenvolvimento

- Avaliação de desempenho
  - Eficiência de espaço e de tempo
  - Generalidade da solução

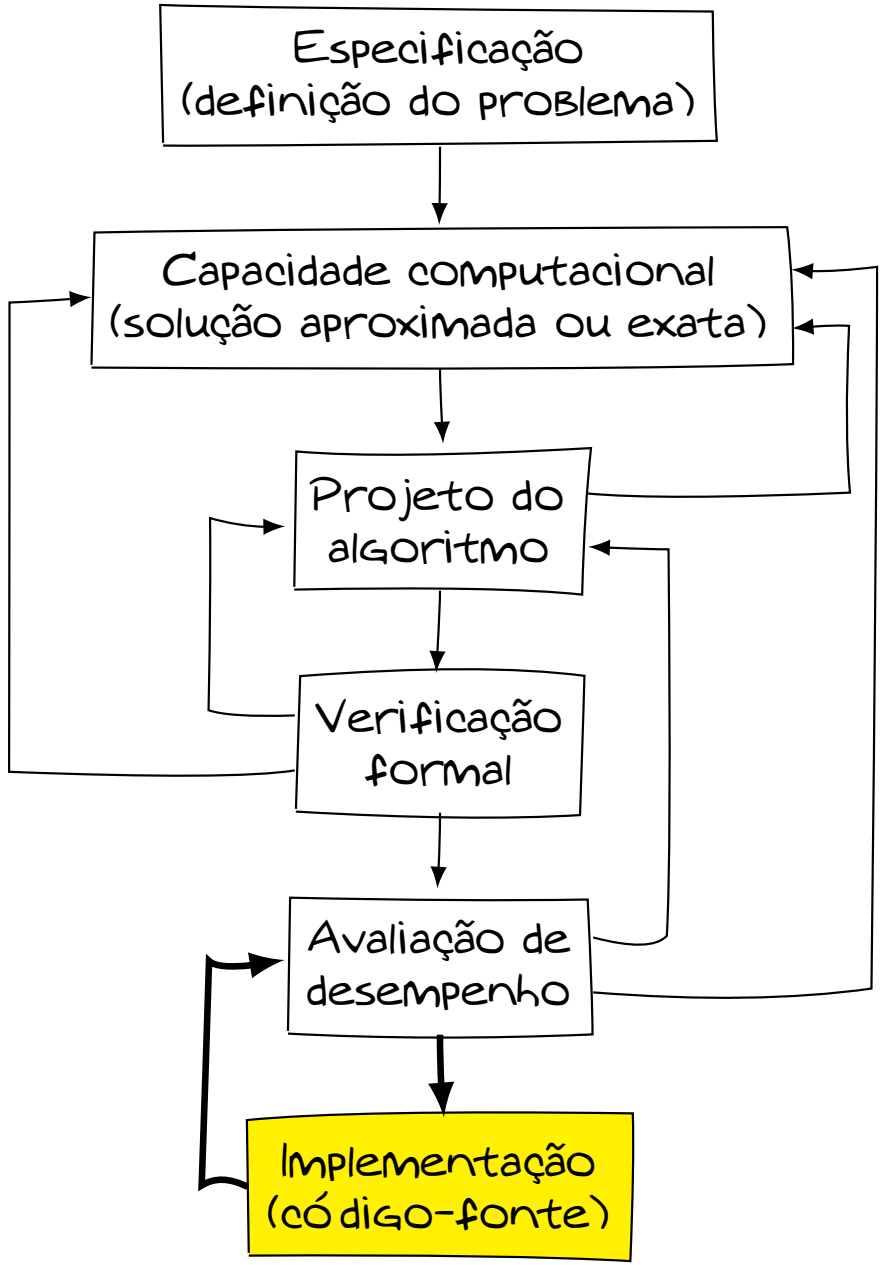
# Fluxo de desenvolvimento

- Avaliação de desempenho
  - Eficiência de espaço e de tempo
  - Generalidade da solução
  - Simplicidade das operações

# Fluxo de desenvolvimento



# Fluxo de desenvolvimento



# Fluxo de desenvolvimento

- Implementação (código-fonte)
  - Definição da linguagem de programação
    - Requisitos do sistema

# Fluxo de desenvolvimento

- Implementação (código-fonte)
  - Definição da linguagem de programação
    - Requisitos do sistema
    - Bibliotecas, ferramentas e recursos disponíveis

# Fluxo de desenvolvimento

- Implementação (código-fonte)
  - Definição da linguagem de programação
    - Requisitos do sistema
    - Bibliotecas, ferramentas e recursos disponíveis
    - Domínio da sintaxe e semântica da linguagem

# Exemplo

- Considerando o problema de ordenação de números reais de 64 bits, com sequências com tamanho de até 1.000.000 números
  - Defina qual o seu problema, com requisitos funcionais e não funcionais, além do escopo
  - Descreva seu algoritmo em pseudocódigo
  - As etapas de verificação formal e de implementação podem ser omitidas