Introdução à Programação e Pensamento Computacional

Status	Done
Assign	
Principais Conhecimentos	Abstração Algoritmos Lógica de Programação
Data	@June 20, 2022 8:00 PM

Aula 01: Pensamento Computacional

Introdução

💡 Processo de pensamento envolvido nas soluções em passos computacionais ou algoritmos que podem ser implementados no computador

> Quando escrevemos, tiramos o que está na memória e colocamos no HD — Prof. Juliana Mascarenhas

Pensamento computacional não é uma disciplina académica é uma habilidade genérica

Pilares do Pensamento Computacional

- Decomposição: Dividir grandes problemas em problemas menores e resolver um de cada vez
- Reconhecimento de padrões: identificar tendências ou padrões, ou seja, identificar similaridades em coisas no mesmo contexto ou em contextos diferentes.
- Abstracção: Extrapolar o conceito do problema para uma forma mais generalista. Pode ser idealizar algo para se adaptar as necessidades dos usuários.
- Design de algoritmos: automatizar — definir um passo a passo para a resolução do problema.

No pensamento computacional podemos usar o melhor dos dois mundo - computacional e humano. Os humanos conseguem com muita facilidade fazer reconhecimento de padrões enquanto os computadores, por exemplo, precisam de aprender antes como fazer isso.

Competências do Pensamento Computacional

- Pensamento sistemático/analítico
- Colaboração dentro da equipe
- Criatividade
- Facilitador

Habilidades Complementares

Raciocínio lógico

Forma de pensamento estruturado que permite encontrar a conclusão ou determinar a resolução de um problema.

Classificado em:

- **Indução**: parte de um fenómeno observado que permite criar leis e teorias — muito relacionando as ciências experimentais;
- Dedução: parte de leis e teorias para fazer previsões e explicações — relacionado com as ciências exactas
- Abdução: A partir de um facto, tiramos uma premissa que não precisa ser verdadeira — relacionado com processo de investigação e diagnósticos

> 💡 **OBS: Abdução e Indução estão relacionadas directamente para análises sintéticas já a dedução está relacionada com a análise analítica**

Aperfeiçoamento

Consistem em, a partir de uma solução encontrar um ponto de melhora e refinamento.

O de aperfeiçoar envolve:

- **Melhorar os recursos:** encontrar soluções eficientes e optimizar processos.
- **Melhorar códigos e algoritmos:** simplificar linhas de códigos e funções bem definidas

Pilares do Pensamento Computacional — Decomposição

Dividir um grande problema em problemas menores.

Estratégias de decomposição

- **Analise:** Processo de divisão e determinar as partes menores que possam ser geridas e depois estudar, explorar ou realizar um exame detalhado;
- **Síntese:** Combinar os elementos recompondo o problema, ou seja, consiste em reunir de maneira coerente os elementos devidos em um único.
- Ordem de execução de tarefas:
  - Sequencial: Quando existem dependências entre tarefas, uma fila de tarefas e existe uma ordem de execução;
  - Paralelo: Podem ser feitas em simultâneo — mais eficiência, menos tempo.

NA decomposição existem variáveis nos problemas pequenos determinados pelos problemas maiores. Da mesma forma que o raciocínio lógico, é preciso treinar. Possa haver várias formas de resolver um problema.

Como fazer a decomposição

- Identificar ou colectar os dados relacionados com o problema;
- Agregar os dados;
- Funcionalidades.

Pilares do Pensamento Computacional — Reconhecimento de Padrões

Um padrão pode ser encarado como um modelo base que não varia e é temporalmente infinito.

Podemos reconhecer os padrões no contexto computacional através das similaridades e diferenças.

Reconhecemos padrões desde que nascemos através de modelos através das formas, diferenças e similaridades das coisas e sua composição.

Porque determinar padrões?

Permite generalizar para resolução de problemas diferentes. Isso pode ser feito através das categorias e da classificação

Como o computador reconhece os padrões?

Representação de atributos

Conceito associado ao objecto

Armazenamento de dados

Regras de decisão

Pilares do Pensamento Computacional — Abstracção

> Processo intelectual de isolamento um objecto da realidade
>
> Generalização: reunir numa classe geral um conjunto de seres ou fenômenos familiares.
>
> Os dados podem ser classificados através de:
>
> - características
> - pontos essenciais
> - Generalizar ou detalhar

Pilares do Pensamento Computacional — Algoritmos

Algoritmos servem para resolver problemas passo-à-passo. Precisa ser entendido por um humano e por uma máquina

Para desenvolver um programa é preciso:

- Analise: Estudar e definição dos dados de entrada e saída;
- Algoritmo: Descrever o problema por meio de ferramentas narrativas, fluxogramas ou pseudocódigo
- Codificação: escrever o programa em uma liguagem de programação.

Todos esses passos consistem em:

- Sequência de passos com objectivos bem definidos
- Execução de tarefas específicas;
- Conjunto de operações que resultam em uma sucessão de finitas acções

Como construir um algoritmo?

- Compreender o problema;

- Definição dos dados de entrada;
- Definir processamento;
- Definir dados de saída;
- Utilizar um método de construção.

Métodos de construção de algoritmos

- Narrativa: Utilização da linguagem natural;
- Fluxograma: Uso de símbolos pré-definidos;
- Pseudocódigo: Portugol — passos a serem seguidos

Estudo de caso conceitual: perdido

Como resolver o problema utilizando o pensamento computacional?

- Identificar mecanismos
- Recursos comuns
- Detalhes mais importantes

▼ Aula 02: Introdução à Lógica de Programação

O que é a lógica?

A lógica serve para solucionar um problema com numerosas soluções

Lógica pode ser:

- Forma de raciocínio;
- Organizar as coisas de forma coesa

> 💡 Na computação, a lógica é a organização e planeamento das instruções, assertivas em um algoritmo, para viabilizar a implementação de um programa

> Everybody in this country should learn how to program a computer because it teaches you how to think - Steve Jobs

Técnicas de Lógica de Programação

Técnica linear

Execução sequenciada de uma série de operação com recursos limitados.

- Não tem vinculo
- Estrutura hierárquica
- Programação

Técnica estruturada

Tem como objectivos:

- Escrita de programas;
- entendimento do código;
- validação;
- Manutenção.

Técnica Modular

São definidas partes independentes com controladas por um conjunto de regras - cada módulo tem a sua regra específica

Dentro dessa técnica encontramos o modelo padrão:

- Dados de entrada;
- Processo de transformação;
- Dados de saída.

Essa técnica permite:

- simplificar
- decompor o problema
- verificar cada módulo

▼ Aula 03: Fundamentos de algoritmos

Tipologias e Variáveis

O computador tem como função processar as informações que recebem que podem ser dados ou instruções.

Os dados são tratados e processados. Os tipos de dados são:

- Numéricos: Aonde encontramos os inteiros (todos os números positivos ou negativos que não possuem casas decimais) e reais (todo o espectro numérico)
- Caracteres: tudo o que não representa um número
- Lógicos (booleano): aonde temos apenas dois resultados, verdadeiro(1) e falso (0)

Variável

Pode assumir qualquer um dos valores de um determinado conjunto de valores.

As variáveis possuem uma série de regras para sua definição:

- Atribuição de um ou mais caracteres;
- primeira letra — não número;
- sem espaços em branco;
- Utilização de palavras reservadas.

Constante

Tudo aquilo que é fixo ou estável

- inalterável
- Não muda
- Invariável

Instruções Primitivas

As instruções determinam as operações que serão executadas.

Estruturas Condicionais e Operadores

A ideia das estruturas condicionais é permitir uma operação sempre que uma condição é efectuada.

Existem 3 tipos de estruturas condicionais:

- Simples: verifica se a condição fi satisfeita
- Composta: verifica se a condição foi satisfeita, caso contrário é accionada uma execeção
- Encadeada: é uma sucessão de estruturas condicionais

Operadores relacionais

= : igual a

<> : diferente de

> : maior que

< : menor que

≥ : maior ou igual a

≤ : menor ou igual a

Operadores lógicos

- AND: intersercecção
- OR: União-
- NOT: Negação

Estruturas de Repetição

Estruturas de repetição permite:

- Redução de linhas;
- Compreensão de linhas;
- Redução de erros

Vectores e Matrizes

Um vector é caracterizado por uma variável dimensionada com tamanho pré-fixado. Também pode ser encarado como um container.

Matriz é uma tabela organizada em linhas e colunas no formato m x n, aonde m representa o número de linhas(horizontal) e o n o número de colunas (vertical)

O que são Funções?

Funções são blocos de instruções que realizam tarefas específicas - decomposição do algoritmo ou modularização (decompor as tarefas em vários módulos)

Modularização do programa tem as seguintes vantagens:

- código mais claro e conciso;
- Reutilizar instrumentos

Instruções de entrada/saída

Inserção e recepção de dados do mundo real por meio de alguma interface seja teclado, mouse, arquivos entre outros.

Existem dois tipos de saída:

- Por interrupção
- Programada

▼ Aula 04: Linguagens de Programação

Introdução à linguagem de Programação

Historia da computação

- **3mil AC - Primeiro dispositivo de cálculo (ábaco romano)**
- **1937 - Charles Babbage**

  Começou a criar conceitos de software e a determinar que algumas operações podiam ser realizadas e dados podiam ser usados, mas não sabia como associar todas as ideias e começar a programar;
- **1942 - 1943 - Ada Lovelace**

  Ada trabalhou com Babbage e era seguidora de De Morgan um dos percursores da álgebra booleana. Nessa altura, ela fez um manuscriste de Babbage com instruções de formas para executar um prototipo de um computador. Ada é a primeira programadora do mundo.
- 1940 - Alan Turing

  Alan Turing e mais 9 matemáticos fizeram a descriptação. Ele definiu conceito de algoritmos e de inteligência artificial.
- 1946 - Von Neuman e Alan Turing

  Trabalharam no Advanced Computer engenerring um projecto de quase um computador
- 1950 - Alan Turing e a IA

  Escreveu um artigo denominado: “Computer MAchinery And Intelligence” aonde falou sobre a possibilidade de uma máquina pensar
- 1980 - Maquina de Cartões usado no censo americano
- Primeiro computador possuía válvulas e depois relés
- Depois surgiu o Harvard Mark 1 criado pela Universidade de Harvard que fazia separação de memórias e decisão através de algoritmos.
- Por fim, surgiu o Eniac, possuía 18mil válvulas que era programável através de fios. As instruções eram recebidas e depois com os fios eram destinadas manualmente.
- 1943 — urgiu o Colossus que também era programável por fios.
- 1975 — Surge Paul Allen e Bill Gates criaram a linguagem BASIC
- 1976 — surge a Apple

- 1977 — Apple lança o Apple II
- 1980 — Apple lança o Apple III
- 1981 — A IBM lança o IBM PC com tecnologia Open source
- 1983 — Apple laçou o fracasso Lisa
- 1985 — Windows 1.0
- 1988 — Windows 2000
- 2001 — Windows XP

Com toda essa demora na evolução do Hardware, já havia uma linguagem de programação sonante na época.

Em 1949 surgiu a primeira linguagem de programação ou linguagem de montagem, ou máquina chamada Assembly.

Em 1950 era usado o COBOL.

Entre 60 e 70 surgiu a linguagem C e o Prolog.

Na década de 91 começaram a surgir novos conceitos como Java, C#, Python, Ruby entre outras.

> 💡 Linguagem de programação é um método padronizado composto por um conjunto de regras sintácticas e semânticas de implementação de um código-fonte.

Como um computador entende o programa?

Código-fonte é código gerado do programa escrito em uma linguagem de programação de alto nível.

Esse código gerado deve ser traduzido em linguagem de máquina para o computador entender existem duas formas:

- Tradução: Criação de um programa objecto e a sua execução (execução rápida e programas menores)
- Interpretação: Programa fonte executado directamente

Caracteristicas de um programa

Directrizes:

- Legibilidade

  O código deve ser coerente de ser fácil de ler e compreender e deve ter as definições adequadas das suas estruturas

- Regibilidade

  Coerência nas instruções, ou seja, o código deve ser bem escrito considerando a escrita simples e a reutilização do código

- confiabilidade

  Verificação de tipos e compatibilidade entre compiladores. O programa deve fazer o que foi programado para fazer

- Custo

  Análise de impacto para isso é necessário treinamento, codificação, compilação e a infra-estrutura a ser usada.

Outras características importantes são:

- Actualizações
- uso de IA
- Disponibilidade de ferramentas
- Comunidade activa
- Adopção no mercado

Análise de código

Analise léxica

Tem como função fazer a leitura do código, caractere por caractere.

Analise sintaxica

Forma como define a estrutura relacionada na linguagem de programação

Anlise semantica

Relação entre os significados. Os erros de semantica não faz o que é esperado

Paradigmas da Pogramação

▼ Aula 05: Primeiro contacto com a programação

Algoritmos em Portugol

Considerações finais