Pensamento Computacional e algoritmos

Introdução ao pensamento computacional

Pensamento computacional

Processo de pensamento envolvido na expressão de soluções em passos computacionais ou algoritmos que podem ser implementados no computador(Aho,2011;Lee,2016).

- Sistemático e eficiente
- Formulação e resolução de problemas
- Sejam capazes de resolver (tanto humanos quanto máquina)

Pilares

Decomposição → Pegar o problema complexo e segmenta-lo em subproblemas, menores, menos complexos e mais resolvíveis.

▼ Reconhecimento de padrões

identificar similaridades e diferenças entre os problemas

▼ Abstração

só pego o que me interessa referente ao problema

▼ Design de algoritmos

Estruturar o algoritmo e suas fases (passo a passo)

> O principal objetivo é dividir um problema complexo em subproblemas

Ciclo básico de desenvolvimento

- Definir a solução
- Testar a solução
- Refatorar

Raciocínio Lógico

É uma forma de pensamento estruturado, ou raciocínio, que permite encontrar a conclusão ou determinar a resolução de um problema.

Indução

A partir de um fenômeno observado → formulação de Leis e teorias

Dedução

A partir de leis e teorias → dedução de previsões e explicações

Abdução

A partir de uma conclusão → gerasse uma premissa (Não necessariamente verdadeira)

Aperfeiçoamento

A partir de uma solução determinar os pontos de melhora e refinamento

Decomposição

> Dado um problema complexo, devemos quebra-lo em problemas menores. Portanto, problemas mais fáceis e gerenciáveis.

- Análise → Processo de quebrar e determinar partes menores e gerenciáveis
- Síntese → Combinar os elementos recompondo o problema original (fazendo sentido)
- A ordem de execução de tarefas pode ser sequencial ou em paralelo

- Identificar ou coletar dados → Agregar os dados → Funcionalidades

Padrões

- Perceber as similaridade e diferenças (extrair características para classificar dados)
- Determina-se os padrões para se conseguir generalizar, com objetivo de obter resolução para problemas diferentes
- Determinamos os padrões através da similaridade (grupos conhecidos x objetos desconhecido)
- O computador percebe padrões através de comparação

Abstração

> Basicamente eu só pego as características que me interessam (pegar um objeto e extrapolar para o mundo abstrato)

- A abstração resulta de uma generalização

Algoritmos

> É sequência de passos finitas e organizadas que quando são executadas resolvem totalmente ou parcialmente um problema.

Fases de um algoritmo

> Entrada → processamento → saída

- Entendido por humanos e por uma máquina
- Devemos entender o problema → pensar no algoritmo → codificar
- Para descrever o algoritmo podemos utilizar uma narrativa, fluxograma ou pseudocódigo

O que é logica

> Problema é uma questão que foge a uma determinada regra, ou melhor é um desvio de percurso, o qual impede de atingir um objetivo com eficiência e eficácia

> A lógica é parte da filosofia que trata das formas do pensamento em geral (dedução, indução, hipótese, inferência etc) e das operações intelectuais que visam à determinação do que é verdadeiro ou não
> -
> Organização e planejamento das instruções, assertivas em um algoritmo, a fim de viabilizar a implantação de um programa

Técnicas de lógica de programação

Linear

- Estrutura hierárquica
- Modelo de desenvolvimento e resolução
- Muito ligada a matemática
- Execução sequencial/Recursos limitados/única dimensão

Estruturada

- Organização, disposição e ordem dos elementos essenciais que compõem um corpo (concreto ou abstrato)
- Não linear (pode ter condições)

Modular

- Partes independentes controladas por um conjunto de regras

- Simplifica
- decompõe o problema

Tipologia e variáveis

▼ Numéricos

Inteiros: -3,-2,-1,0,1,2,3

Reais: -16,555, 3.14, 4,02

▼ Caracteres

Letras/ números/ símbolos → palavras

▼ Lógicos

Verdadeiro(1)/Falso(0)

Variáveis

> Vai armazenar os dados, podendo mudar seu conteúdo durante a execução

- Primeira letra não pode ser número
- Sem espaços em branco
- Não pode utilizar palavras reservadas
- Pode conter caracteres numéricos (sem ser no começo)
- O único caractere especial permitido é “_”

Constante

> Armazena algo que não muda

Instruções primitivas

> Instruções são linguagens de palavras-chave(vocabulário) de uma determinada programação que tem por finalidade comandar um computador que irá tratar os dados

- Determina as ações que serão executados sobre os dados

Estruturas condicionais e operadores

- Dado uma determinada situação isso implicará em uma solução
- Se a condição for satisfeita a condição será executada
- condição → operação (simples)
- condição → operação → senão → operação (composta)

Operadores

Condicional simples

```
se (<condição> então
    <instrução>
fim_se
```

Condicional composta

```
se (<condição> então
    <instrução>
senão
    <instrução>
fim_se
```

Condicional encadeado

```
se (<condição> então
    <instrução>
senão
```

se (<condição> então

<instrução>

senão

<instrução>

fim_se

Operadores lógicos

- and

  Todas as entradas tem que ser verdadeira para a expressão ser verdadeira

- or

  só precisa de uma entrada verdadeira para que toda expressão seja verdadeira

- not

  inverte o valor da expressão (verdadeira → falso e virse versa)

Estruturas de repetição

- Executa parte do código varias vezes
- Possuem condições de parada
- Reduz linhas
- Facilita a compreensão

Enquanto faça

- Enquanto a condição for verdadeira a condição é executada

  Enquanto (<condição>) faça

  <instrução>

  Fim_enquanto

Repita até

- Executa uma vez, e depois testa até que a condição se torne falsa

repita

até <condição>

para

- Possui o número de repetição bem definido
- Estrutura de contagem

  para <inicio> até <condição>

  fim_para

Vetores e matrizes

- Sequencia de dados de um mesmo tipo
- Tamanho pré-fixado
- Acesso através de índices
- vetores são unidimensionais
- vetores são multidimensionais i x j (alguns dizem que é um vetor de vetores)

1. Vetor tamanho 2
2. Matriz 3x3

Funções

- Possibilita modularizar o código, separando em blocos partes do código, que fazem tarefas específicas, deixando o código mais simples e possibilitando a reutilização
- Funções retornam valores e procedimentos não retornam nada
- Possui nomes e parâmetros (assinatura)

Entrada e saída

> Possibilita a interação com o mundo real, através de equipamentos periféricos

Linguagens de programação

- Em 1949 surgiu a primeira linguagem de montagem o Assembly
- Possui regras sintáticas e semânticas
- É um intermediário entre o ser humano e as máquinas
- Pode ser uma linguagem compilada ou interpretada. Nas linguagens compiladas é feito uma transformação do código fonte em instruções que o computador entende, tornando o processo mais rápido porém ocupa mais espaço. Já na interpretada executa linha por linha e traduz linhas por linha, o que economiza o espaço, mas é mais lenta.

Características

- Legibilidade → todos podem entender/ler/compreender
- Redigibilidade →Facilidade na escrita/produção
- Confiabilidade → O código deve executar o que se espera dele
- Custo →Seu custo tanto computacional, quanto monetário, para construção, manutenção etc.. deve ser baixo

Compilador analisa o código

- Faz análises léxicas, sintáticas e semânticas

Análise léxica

- Ler o programa fonte caractere, por caracteres para formar lexemas (tokens/palavras válidas)
- Agrupa os tokens
- Elimina os comentários, espaços em branco

Análise sintática

- Verifica se as palavras(lexemas/tokens) são validos para aquela linguagem

Análise semântica

- Verifica o sentido/significado das palavras/tokens
- Verifica a lógica

Paradigmas de programação

> Forma de resolução de problemas com diretrizes e limitações especificas de cada paradigma utilizando linguagem de programação

Estruturado

- Instruções são executadas em sequência
- Decisão → testes lógicos
- Iteração → funções, laços, condições
- Eficiente para problemas simples e diretos

Programação Orientada a Objeto (POO)

- Tenta aproximar o mundo abstrato ao mundo real
- A unidade fundamental da OO é a classe (definem suas características[estados] e ações)
- Objetos são estruturas do mundo real sejam elas abstratas ou não, generalizadas, são instancias das classes
- As características são denominados atributos e as ações são os métodos

▼ Pilares

Abstração → Só pego o que me interessa

Herança → agrupa comportamentos incomuns para a reutilização, fazendo assim classes herdarem características/comportamentos de outras

Encapsulamento → reduz o escopo(visibilidade) de métodos/variáveis para certos blocos/classes

Polimorfismo → Um método/classe pode assumir várias formas, isso resulta na utilização

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/2f345ff8-a410-462b-b8f2-12b4a40d07dd/mediaAluno.por

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/94213dae-7e91-47c7-8c23-a2101a58b0ee/somaIntervalo.por

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/0f0ca857-6ac8-4b99-a76c-6a158b1006aa/Pensamento_computacional_part_1.pdf

https://s3-us-west-2.amazonaws.com/secure.notion-static.com/be090934-fd16-4400-ad5b-07dc413a55d8/Pensamento_computacional_part_2.pdf