Aula 06
LÓGICA DE PROGRAMAÇÃO

tbt DIGITAL

Professor
Tiago Amaro

Lógica de Programação

Conteúdo:

- Algoritmos de ordenação
  - Bolha, inserção e rápida
- Método de busca binária

Algoritmos de Ordenação

- Ordenam uma coleção em uma determinada ordem
  - Normalmente em ordem crescente
- Recebe uma coleção
- Retorna a coleção ordenada

Algoritmos de Ordenação (Bolha)

- Compara elementos adjacentes de uma lista
- Se um elemento for menor, troca de posição
- Termina quando toda lista for percorrida
- Simples implementação, mas não é rápido
    - Para coleções grandes tem um alto custo computacional

Algoritmos de Ordenação (Bolha)

- Exemplo: https://visualgo.net/pt/sorting
  - Em inglês: bubble sort

Algoritmos de ordenação (Bolha)

```
function bubbleSort (items) {
  const length = items.length
  for (let i = (length - 1); i >= 0; i--) {
    for (let j = (length - i); j > 0; j--) {
      // Compare the adjacent positions
      if (items[j] < items[j - 1]) {
        // Swap the numbers
        const temporary = items[j]
        items[j] = items[j - 1]
        items[j - 1] = temporary
      }
    }
  }
}
```

Fonte: https://gist.github.com/tiagoamaro/b4035380d32b49b89705e094dd050b1f

Algoritmos de Ordenação (Inserção)

- Divide uma coleção em duas partes
  - Uma ordenada e outra não ordenada
- Compara cada elemento da lista não ordenada
- Insere o elemento na posição correta da lista ordenada

Algoritmos de Ordenação (Inserção)

- Exemplo: https://visualgo.net/pt/sorting
  - Em inglês: insertion sort

Algoritmos de ordenação (Inserção)

```
function insertionSort (collection) {
  const len = collection.length

  for (let unsortedIndex = 1; unsortedIndex < len; unsortedIndex++) {
    let sortedIndex = unsortedIndex
    const element = collection[unsortedIndex]

    while (sortedIndex > 0 && collection[sortedIndex - 1] > element) {
      collection[sortedIndex] = collection[sortedIndex - 1]
      sortedIndex--
    }

    collection[sortedIndex] = element
  }

  return collection
}
```

Fonte: https://gist.github.com/tiagoamaro/b4035380d32b49b89705e094dd050b1f

Algoritmos de Ordenação (Rápida)

- Técnica de "dividir e conquistar"
- Seleciona um elemento pivô de uma coleção
  - Particiona a coleção em duas
  - A escolha do pivô varia de acordo com a implementação
- Ordena recursivamente elementos das coleções

Algoritmos de Ordenação (Rápida)

- Exemplo: https://visualgo.net/pt/sorting
  - Em inglês: quick sort

Algoritmos de ordenação (Rápida)

```
function quicksort (array) {
  if (array.length <= 1) {
    return array
  }

  const pivot = array[0]
  const left = []
  const right = []

  for (let i = 1; i < array.length; i++) {
    array[i] < pivot ? left.push(array[i]) : right.push(array[i])
  }

  return quicksort(left).concat(pivot, quicksort(right))
};
```

Fonte: https://gist.github.com/tiagoamaro/b4035380d32b49b89705e094dd050b1f

Algoritmos de ordenação (Nativo)

```
const collection = [3, 2, 6, 10]
collection.sort() // will return sorted array
collection        // won't mutate the variable
```

Fonte: https://gist.github.com/tiagoamaro/b4035380d32b49b89705e094dd050b1f

Método de Busca Binária

- Busca um elemento dentro de uma coleção ordenada
- A cada iteração, percorre metade da coleção
- Eficiente, executando poucas operações

Método de Busca Binária

- Exemplo: Binary and Linear Search Visualization
  - Em inglês: binary search
  - Fonte: David Galles (University of San Francisco)

Método de Busca Binária

```
function binarySearch (collection, element) {
  let start = 0
  let end = collection.length - 1

  while (start <= end) {
    // Find the middle
    const mid = Math.floor((start + end) / 2)

    // Found the element!
    if (collection[mid] === element) {
      return true
    } else if (collection[mid] < element) {
      start = mid + 1
    } else {
      end = mid - 1
    }
  }

  return false
}
```

Fonte: https://gist.github.com/tiagoamaro/b4035380d32b49b89705e094dd050b1f

Algoritmos de Busca (Nativo)

```
const collection = [5, 12, 8, 130, 44];
const found = collection.find(element => element === 12);

console.log(found) // will print 12
```

Fonte: https://gist.github.com/tiagoamaro/b4035380d32b49b89705e094dd050b1f

MATONDO!

(OBRIGADO)

toti

Diversidade para Inovação.

contato@toti.site

toti.site

@toti.br

/toti.diversidade

/company/toti-br