Aula 02

Rodrigo G. Araújo

25/07/2019

Big O

No pior caso da execução deste algoritmo, o número de operações realizado será proporcional a N, e por simplicidade, eliminamos constantes e fatores não dominantes.

A quantidade de operações por segundo que podem ser executadas em um problema de maratona está por volta de 10^8 e a memória maxima 10^6

O(1)

```python
print("Python > Java")
lista = [1, 2, 3, 4, 5, 6]
print(lista[3])
numeros = {1, 2, 3, 4, 5, 6}
print(3 in numeros)
cidades = {'Brasilia':'DF', 'Goiania':'Goias'}
print('Florianopolis' in cidades)
```

$O(N)$ e $O(N^2)$

```python
# O(N)
for i in range(10):
    print(i)

# O(N²)
for i in range(10):
    for j in range(10):
        print(i, j)
```

Busca Binária

$$O(logN)$$

```
vetor = list(map(int, input().split()))
find = int(input())
inicio, fim = 0, len(vetor)
while inicio < fim:
    media = (inicio+fim)//2
    print(inicio, fim, vetor[inicio:fim], vetor[media])
    if find == vetor[media]:
        print('YES')
        exit()
    if vetor[media] < find:
        inicio = media+1
    else:
        fim = media
print('NO')
```

Pilha

```
pilha = list()
qtd = int(input())
for _ in range(qtd):
    pilha.append(int(input()))

print(pilha.pop())
```

Fila

```
fila = list()
qtd = int(input())
for _ in range(qtd):
    fila.append(int(input()))

fila = list(reversed(fila))
print(fila.pop())
```

Dicionário

```
cardapio = {
    '01':3.50, '03':7.90, '11':5.20
}
pedidos = input().split()
print(sum([cardapio.get(pedido, 0) for pedido in pedidos]))
```