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]))
```

Exercicios

A - A and B and Compilation Errors

```
qtd = input()
error1 = sum(map(int, input().split()))
error2 = sum(map(int, input().split()))
error3 = sum(map(int, input().split()))

print(error1-error2)
print(error2-error3)
```

Exercicios

B - Snacktower

```
from queue import PriorityQueue


def get_na_fila(show_snack, maior):
    while True:
        if fila_preferencia.empty():
            break

        snack = -1*fila_preferencia.get()
        if snack == maior:
            show_snack.append(snack)
            maior -= 1
        else:
            fila_preferencia.put(-1*snack)
            break
    return maior
```

Exercicios

B - Snacktower

```
maior = int(input())
snacks = map(int, input().split())
fila_preferencia, show_snack = PriorityQueue(), list()


for snack in snacks:
    if snack == maior:
        show_snack.append(snack)
        maior -= 1
        maior = get_na_fila(show_snack, maior)
        print(*show_snack)
        show_snack = list()
    else:
        fila_preferencia.put(-1*snack)
        print()
```

Exercicios

C - Sushi for Two

```
from collections import defaultdict
qtd = int(input())
sushi = list(map(int, input().split()))
sushi.append(0)
pecas, atual, sequencia = list(), sushi[0], 0

for peca in sushi:
    if atual != peca:
        pecas.append(sequencia)
        atual = peca
        sequencia = 0
    sequencia += 1

res = 0
for i in range(1, len(pecas)):
    valor = 2*min(pecas[i-1], pecas[i])
    if valor > res:
        res = valor

print(res)
```

Exercicios

D - Sport Mafia

```
movimentos, doces_final = map(int, input().split())
left, right = 0, movimentos + 1

while left < right -1 :
    media = (left + right)//2
    cedidos = (media * (media + 1)) // 2
    comidos = movimentos - media
    if cedidos - comidos > doces_final:
        right = media
    else:
        left = media
print(movimentos - left)
```

Exercicios

E - Planning The Expedition

```
from collections import Counter

pessoas, pacotes = map(int,input().split())
comidas = map(int, input().split())
vetor = list(Counter(comidas).values())
dias = pacotes//pessoas
while dias > 0:
    if sum([comida//dias for comida in vetor]) >= pessoas:
        break
    dias-=1
print(dias)
```