HO09: Armazenamento e Organização de Dados

In [ ]: import math

```
# Parametro: t Tamanho do bloco
# Parametro: r Tamanho do registro
# Parametro: n Quantidade de registros do arquivo
def organizacao_de_dados(t: int, r: int, n: int):
  F = math.floor(t / r) #fator de bloco
  print("Fator de bloco:", F)
  B = math.ceil(n / F) #numero de blocos necessarios
  print("Número de blocos necessários:", B)
  U = t - (F * r)
  print("Espaço desperdiçado:", U, "bytes")
  espaco_total = B * t
  print("Espaço Total:", espaco_total, "bytes")
```

Arquivo Atores¶

Atores (10.000 registros) → Codigo (16B), Nome (160B)

In [ ]: t = 2 * 1024 *#quantidade de bytes por bloco*

In [ ]: r = 16 + 160 *#quantidade de bytes por registro*

```
F = math.floor(t / r) #fator de bloco
print("Fator de bloco:", F)
```

Fator de bloco: 11

In [ ]: n = 10000 *#numero de registros do arquivo*

```
B = math.ceil(n / F) #numero de blocos necessarios
print("Número de blocos necessários:", B)
```

Número de blocos necessários: 910

In [ ]: U = t - (F * r)

```
print("Espaço desperdiçado:", U, "bytes")
```

Espaço desperdiçado: 112 bytes
In [ ]: espaco_total = B * t
print("Espaço Total:", espaco_total, "bytes")

Espaço Total: 1863680 bytes

Resposta:
Fator de bloco: 11
Número de blocos necessários: 910
Espaço desperdiçado: 112 bytes
Espaço Total: 1.863.680 bytes

Arquivo Clientes¶
Clientes (100.000 registros) → CPF (11B), Nome (160B), Endereco (200B), Telefone (16B), DataNascimento (12B), Sexo (1B)

In [ ]: r = 11 + 160 + 200 + 16 + 12 + 1 *#quantidade de bytes por registro*
F = math.floor(t / r) *#fator de bloco*
print("Fator de bloco:", F)

Fator de bloco: 5
In [ ]: n = 100000 *#numero de registros do arquivo*
B = math.ceil(n / F) *#numero de blocos necessarios*
print("Número de blocos necessários:", B)

Número de blocos necessários: 20000

In [ ]: U = t - (F * r)
print("Espaço desperdiçado:", U, "bytes")

Espaço desperdiçado: 48 bytes

In [ ]: espaco_total = B * t
print("Espaço Total:", espaco_total, "bytes")

Espaço Total: 40960000 bytes

Resposta
Fator de bloco: 5
Número de blocos necessários: 20.000
Espaço desperdiçado: 48 bytes
Espaço Total: 40960000 bytes

Arquivo Filmes¶
Filmes (2.000.000 registros) → Codigo (16B), Nome (160B), Genero (80B)

In [ ]: r = 16 + 160 + 80 *#quantidade de bytes por registro*
F = math.floor(t / r) *#fator de bloco*
print("Fator de bloco:", F)

Fator de bloco: 8

In [ ]: n = 2000000 *#numero de registros do arquivo*
B = math.ceil(n / F) *#numero de blocos necessarios*
print("Número de blocos necessários:", B)

Número de blocos necessários: 250000

In [ ]: U = t - (F * r)
print("Espaço desperdiçado:", U, "bytes")

Espaço desperdiçado: 0 bytes

In [ ]: espaco_total = B * t
print("Espaço Total:", espaco_total, "bytes")

Espaço Total: 512000000 bytes

Resposta
Fator de bloco: 8
Número de blocos necessários: 250.000
Espaço desperdiçado: 0 bytes
Espaço Total: 512.000.000 bytes

Arquivo Funcionarios¶
Funcionarios (3.500registros) → CPF (11B), Nome (160B)

In [ ]: t = 2 * 1024 *#quantidade de bytes por bloco*
r = 11 + 160 *#quantidade de bytes por registro*
F = math.floor(t / r) *#fator de bloco*
print("Fator de bloco:", F)

Fator de bloco: 11

In [ ]: n = 3500 *#numero de registros do arquivo*
B = math.ceil(n / F) *#numero de blocos necessarios*
print("Número de blocos necessários:", B)

Número de blocos necessários: 319

In [ ]: U = t - (F * r)
print("Espaço desperdiçado:", U, "bytes")

Espaço desperdiçado: 167 bytes

In [ ]: espaco_total = B * t
print("Espaço Total:", espaco_total, "bytes")

Espaço Total: 653312 bytes

Resposta
Fator de bloco: 11
Número de blocos necessários: 319
Espaço desperdiçado: 167 bytes
Espaço Total: 653.312 bytes

Arquivo Midias¶
Midias (10.000.000 registros) → Identificador (24B), Tipo (8B), PrecoDiaria (24B)

In [ ]: r = 24 + 8 + 24 *#quantidade de bytes por registro*
F = math.floor(t / r) *#fator de bloco*

print("Fator de bloco:", F)

Fator de bloco: 36

In [ ]: n = 10000000 *#numero de registros do arquivo*
B = math.ceil(n / F) *#numero de blocos necessarios*
print("Número de blocos necessários:", B)

Número de blocos necessários: 277778

In [ ]: U = t - (F * r)
print("Espaço desperdiçado:", U, "bytes")

Espaço desperdiçado: 32 bytes

In [ ]: espaco_total = B * t
print("Espaço Total:", espaco_total, "bytes")

Espaço Total: 568889344 bytes

Resposta
Fator de bloco: 36
Número de blocos necessários: 277.778
Espaço desperdiçado: 32 bytes
Espaço Total: 568.889.344 bytes

Arquivo Aluguel¶
Aluguel (20.000.000 registros) → DataLocacao (12B), DataDevolucao (10B), ValorPagar (24B)

In [ ]: r = 12 + 10 + 24 *#quantidade de bytes por registro*
F = math.floor(t / r) *#fator de bloco*
print("Fator de bloco:", F)

Fator de bloco: 44

In [ ]: n = 20000000 *#numero de registros do arquivo*
B = math.ceil(n / F) *#numero de blocos necessarios*

print("Número de blocos necessários:", B)

Número de blocos necessários: 454546

In [ ]: U = t - (F * r)
print("Espaço desperdiçado:", U, "bytes")

Espaço desperdiçado: 24 bytes

In [ ]: espaco_total = B * t
print("Espaço Total:", espaco_total, "bytes")

Espaço Total: 930910208 bytes

Resposta
Fator de bloco: 44
Número de blocos necessários: 454.546
Espaço desperdiçado: 24 bytes
Espaço Total: 930.910.208 bytes

Arquivo Pagamentos¶
Pagamentos (50.000.000registros) → Codigo (48B), Data (12B), Valor (24B)

In [ ]: r = 48 + 12 + 24 *#quantidade de bytes por registro*
F = math.floor(t / r) *#fator de bloco*
print("Fator de bloco:", F)

Fator de bloco: 24

In [ ]: n = 50000000 *#numero de registros do arquivo*
B = math.ceil(n / F) *#numero de blocos necessarios*
print("Número de blocos necessários:", B)

Número de blocos necessários: 2083334

In [ ]: U = t - (F * r)
print("Espaço desperdiçado:", U, "bytes")

Espaço desperdiçado: 32 bytes

In [ ]: espaco_total = B * t
print("Espaço Total:", espaco_total, "bytes")

Espaço Total: 4266668032 bytes

Resposta
Fator de bloco: 24
Número de blocos necessários: 2.083.334
Espaço desperdiçado: 32 bytes
Espaço Total: 4.266.668.032 bytes

Arquivo AtoresEmFilmes¶
AtoresEmFilmes (1.000.000 registros) → CodFilme (16B), CodAtor (16B)

In [ ]: r = 16 + 16 *#quantidade de bytes por registro*
F = math.floor(t / r) *#fator de bloco*
print("Fator de bloco:", F)

Fator de bloco: 64

In [ ]: n = 1000000 *#numero de registros do arquivo*
B = math.ceil(n / F) *#numero de blocos necessarios*
print("Número de blocos necessários:", B)

Número de blocos necessários: 15625

In [ ]: U = t - (F * r)
print("Espaço desperdiçado:", U, "bytes")

Espaço desperdiçado: 0 bytes

In [ ]: espaco_total = B * t
print("Espaço Total:", espaco_total, "bytes")

Espaço Total: 32000000 bytes

Resposta
Fator de bloco: 64
Número de blocos necessários: 15.625
Espaço desperdiçado: 0 bytes
Espaço Total: 32.000.000 bytes