Organização e Recuperação de Dados
Profa. Valéria

4ª Lista de Exercícios

1. No contexto de um arquivo, qual é a diferença entre fragmentação interna e externa? Como a compactação afeta a fragmentação interna? E a fragmentação externa?

   Fragmentação interna refere-se a espaços sem uso dentro de registros, já fragmentação externa refere-se a espaços sem uso no arquivo, sem que façam parte de um registro válido.

   A compactação não resolve o problema de fragmentação interna. Já a fragmentação externa é completamente eliminada quando o arquivo é compactado.

2. Quando o arquivo possui registros de tamanho fixo, podemos tratar a Lista de Espaços Disponíveis como uma pilha (PED). Por que o mesmo não pode ser feito em arquivos com registro de tamanho variável?

   Porque em uma pilha só podemos inserir e remover no topo. Quando os registros têm tamanho fixo, faz sentido que os espaços disponíveis fiquem em uma pilha, pois todos os espaços têm o mesmo tamanho e todos serão igualmente adequados para a reutilização. No caso de registros de tamanho variável, os espaços disponíveis têm tamanhos diferentes. Pode acontecer, por exemplo, de um espaço adequado para reutilização estar no meio da lista encadeada de espaços. Se estivéssemos usando uma pilha, não poderíamos reutilizá-lo porque ele não estaria no topo. Mas, usando uma lista, podemos inserir e remover espaços da lista em qualquer posição.

3. Dada os seguintes Topo de PED e arquivo representados na figura:

   Topo da PED → 5

0	1	2	3	4	5	6
João...	Pedro...	Luiz...	*-1	Paula...	*3	Rui...

   Mostre como ficam o topo da PED e o arquivo quando as seguintes alterações são feitas:

   a. Remoção do registro de chave “Rui”;

```
PED = 6

  0      1      2      3      4      5      6
João... Pedro... Luiz... *-1    Paula... *3    *5
```

   b. Remoção do registro de chave “Pedro”;

```
PED = 1

  0      1      2      3      4      5      6
João... *6     Luiz... *-1    Paula... *3    *5
```

   c. Inserção de um registro de chave “Maria”.

```
PED = 6

  0      1      2      3      4      5      6
João... Maria... Luiz... *-1    Paula... *3    *5
```

4. A tabela abaixo mostra o *byte-offset* e o tamanho em *bytes* de alguns registros que estão armazenados em um arquivo com registros de tamanho variável.

Byte-offset	Tam. do registro (em bytes)
28	28
58	30
150	123

4	22
90	58

Suponha que exista uma LED para gerenciar espaços disponíveis desse arquivo. Se removermos os registros de *byte-offset* 28, 150 e 90, nessa ordem, e adicionarmos um registro de 22 bytes de tamanho, qual será o *byte-offset* desse novo registro? Faça o exercício considerando cada uma das estratégias de gerenciamento de LED – (a) *worst-fit*, (b) *best-fit* e (c) *first-fit*.

(a) WORST-FIT

```
LED = 150
4       28          58        90           150
22...   28*-1...    30...     58*28...     123*90...
```

O registro de 22B será inserido no offset 150.

(b) BEST-FIT

```
LED = 28
4       28          58        90           150
22...   28*90...    30...     58*150...    123*-1...
```

O registro de 22B será inserido no offset 28.

(c) FIRST-FIT

```
LED = 90
4       28          58        90           150
22...   28*-1...    30...     58*150...    123*28...
```

O registro de 22B será inserido no offset 90.

5. Considere o arquivo de dados abaixo, com a seguinte organização:
   - Registros de tamanho variável, precedidos por um campo de 2 bytes que armazena o tamanho do registro;
   - Campos delimitados pelo separador "|" (com exceção do campo de tamanho);
   - Cabeçalho de 16 bytes.

```
LED.Head: -1
0          16           39             101             138
Cabeçalho… 21 Soares|… |60 Valdares|… |35 Martineli|… |54 Fonseca|…
```

Considere o uso de uma *LED* e simule as operações seguintes, mostrando, em cada passo, como fica o arquivo e o ponteiro *LED.Head*. Utilize a política de alocação *worst-fit*.

a. Inserção de um novo registro de 45 bytes com chave "Martins"

```
LED.Head: -1
0          16           39             101             138           194
Cabeçalho… 21Soares|… |60Valdares|… |35Martineli|… |54Fonseca|… |45Martins|…
```

b. Remoção do registro de chave "Martineli"

```
LED.Head: 101
0          16           39             101             138           194
Cabeçalho… 21Soares|… |60Valdares|… |35*-1tineli|… |54Fonseca|… |45Martins|…
```

c. Remoção do registro de chave "Valadares"

```
LED.Head: 39
0          16           39             101             138           194
Cabeçalho… 21Soares|… |60*101ares|… |35*-1tineli|… |54Fonseca|… |45Martins|…
```

d. Inserção de um novo registro de 32 bytes com chave “Casale”

```
LED.Head: 101

0          16          39                101             138          194
Cabeçalho… 21Soares|… |60Casale|… |\0… |35*-1tineli|… |54Fonseca|… |45Martins|…
                       (28 bytes de fragmentação interna)
```

e. Remoção do registro de chave “Soares”

```
LED.Head: 101

0          16          39                101             138          194
Cabeçalho… 21*-1res|… |60Casale|… |\0… |35*16tineli|… |54Fonseca|… |45Martins|…
```

6. Repita o exercício anterior utilizando as estratégias (1) *best-fit* e (2) *first-fit*.

(1) BEST-FIT

a. Inserção de um novo registro de 45 bytes com chave “Martins”

```
LED.Head: -1

0          16          39             101             138          194
Cabeçalho… 21Soares|… |60Valdares|… |35Martineli|… |54Fonseca|… |45Martins|…
```

b. Remoção do registro de chave “Martineli”

```
LED.Head: 101

0          16          39             101             138          194
Cabeçalho… 21Soares|… |60Valdares|… |35*-1tineli|… |54Fonseca|… |45Martins|…
```

c. Remoção do registro de chave “Valadares”

```
LED.Head: 101

0          16          39             101             138          194
Cabeçalho… 21Soares|… |60*-1dares|… |35*39tineli|… |54Fonseca|… |45Martins|…
```

d. Inserção de um novo registro de 32 bytes com chave “Casale”

```
LED.Head: 39

0          16          39             101             138          194
Cabeçalho… 21Soares|… |60*-1dares|… |35Casale|…|\0…|54Fonseca|… |45Martins|…
                                     (3 bytes de fragmentação interna)
```

e. Remoção do registro de chave “Soares”

```
LED.Head: 16

0          16          39             101             138          194
Cabeçalho… 21*39res|… |60*-1dares|… |35Casale|…|\0…|54Fonseca|… |45Martins|…
```

(1) FIRST-FIT

a. Inserção de um novo registro de 45 bytes com chave “Martins”

```
LED.Head: -1

0          16          39             101             138          194
Cabeçalho… 21Soares|… |60Valdares|… |35Martineli|… |54Fonseca|… |45Martins|…
```

b. Remoção do registro de chave “Martineli”

```
LED.Head: 101

0          16          39             101             138          194
Cabeçalho… 21Soares|… |60Valdares|… |35*-1tineli|… |54Fonseca|… |45Martins|…
```

c. Remoção do registro de chave “Valadares”

```
LED.Head: 39

0          16          39             101             138          194
Cabeçalho… 21Soares|… |60*101ares|… |35*-1tineli|… |54Fonseca|… |45Martins|…
```

d. Inserção de um novo registro de 32 bytes com chave “Casale”

```
LED.Head: 101

0          16          39              101             138          194
Cabeçalho… 21Soares|… |60Casale|…|\0… |35*-1tineli|… |54Fonseca|… |45Martins|…
                       (28 bytes de fragmentação interna)
```

e. Remoção do registro de chave “Soares”

```
LED.Head: 16

0          16          39              101             138          194
Cabeçalho… 21*101es|… |60Casale|…|\0… |35*-1tineli|… |54Fonseca|… |45Martins|…
```

7. O esquema abaixo representa um arquivo com registros de tamanho variável. Assuma que os 4 primeiros bytes do arquivo são utilizados para o cabeçalho, que os campos são delimitados por "|" e cada registro é precedido por um campo de dois bytes indicando qual é o tamanho do registro. Assuma também que uma LED é utilizada para gerenciar os espaços disponíveis. Assuma que a estratégia de gerenciamento é a *worst-fit* e que as possíveis sobras de espaços resultantes de inserções devem retornar para a LED.

```
LED.head: 201

0   4              82            113           147           201
…   76PIRATA|…     |29PIPA|…     |32THOR|…     |52LADY|…     |24*-1…
```

Simule as operações seguintes, mostrando, em cada passo, como fica o arquivo e o ponteiro *LED.head*.

a) Remoção do registro de chave “PIPA”

```
LED.head: 82

0   4              82            113           147           201
…   76PIRATA|…     |29*201|…     |32THOR|…     |52LADY|…     |24*-1…
```

b) Remoção do registro chave “PIRATA”

```
LED.head: 4

0   4              82            113           147           201
…   76*82ATA|…     |29*201|…     |32THOR|…     |52LADY|…     |24*-1…
```

c) Remoção do registro chave “LADY”

```
LED.head: 4

0   4              82            113           147           201
…   76*147TA|…     |29*201|…     |32THOR|…     |52*82Y|…     |24*-1…
```

d) Inserção de um registro de 40 bytes com chave “ZARA”

```
LED.head: 147

0   4          46          82            113           147           201
…   40ZARA|…   |34*82|…    |29*201|…     |32THOR|…     |52*46Y|…     |24*-1…
```

e) Inserção de um registro de 29 bytes com chave “GHOST”

```
LED.head: 46

0   4          46          82            113           147           178     201
…   40ZARA|…   |34*82|…    |29*201|…     |32THOR|…     |29GHOST|…  |21*-1|…|24*178…
```

f) Inserção de um registro de 19 bytes com chave “NINA”

```
LED.head: 82
0    4          46         67        82            113          147          178      201
…    40ZARA|…  |19*NINA|…|13*-1|…|29*201|…      |32THOR|…     |29GHOST|…  |21*67|…|24*178…
```

Repita o exercício trocando a estratégia de gerenciamento da LED para (1) *first-fit* (primeiro ajuste) e (2) *best-fit* (melhor ajuste).

(1) FIRST-FIT

a) Remoção do registro de chave “PIPA”

```
LED.head: 82
0    4                 82             113           147           201
…    76PIRATA|…       |29*201|…       |32THOR|…     |52LADY|…     |24*-1…
```

b) Remoção do registro chave “PIRATA”

```
LED.head: 4
0    4                 82             113           147           201
…    76*82ATA|…       |29*201|…      |32THOR|…     |52LADY|…     |24*-1…
```

c) Remoção do registro chave “LADY”

```
LED.head: 147
0    4                 82             113           147           201
…    76*82ATA|…       |29*201|…      |32THOR|…     |524DY|…      |24*-1…
```

d) Inserção de um registro de 40 bytes com chave “ZARA”

```
LED.head: 189
0    4                 82             113           147       189     201
…    76*82ATA|…       |29*201|…      |32THOR|…     |40ZARA|…|10*4|…|24*-1…
```

e) Inserção de um registro de 29 bytes com chave “GHOST”

```
LED.head: 35
0    4          35         82           113          147       189     201
…    29GHOST|…|45*189|…  |29*201|…    |32THOR|…    |40ZARA|…|10*82|…|24*-1…
```

f) Inserção de um registro de 19 bytes com chave “NINA”

```
LED.head: 56
0    4          35        56         82           113          147       189     201
…    29GHOST|…|19NINA|…|24*189|…|29*201|…    |32THOR|…    |40ZARA|…|10*82|…|24*-1…
```

(2) BEST-FIT

a) Remoção do registro de chave “PIPA”

```
LED.head: 201
0    4                 82             113           147           201
…    76PIRATA|…       |29*-1A|…       |32THOR|…     |52LADY|…     |24*82…
```

b) Remoção do registro chave "PIRATA"

```
LED.head: 201

0    4              82             113            147            201
…    76-1RATA|…     |29*4A|…       |32THOR|…     |52LADY|…     |24*82…
```

c) Remoção do registro chave "LADY"

```
LED.head: 201

0    4              82             113            147            201
…    76-1RATA|…     |29*147A|…     |32THOR|…     |524ADY|…     |24*82…
```

d) Inserção de um registro de 40 bytes com chave "ZARA"

```
LED.head: 189

0    4              82             113            147       189       201
…    76-1RATA|…     |29*4A|…       |32THOR|…     |40ZARA|… |10*201|…|24*82…
```

e) Inserção de um registro de 29 bytes com chave "GHOST"

```
LED.head: 189

0    4              82             113            147       189       201
…    76-1RATA|…     |29GHOST|…     |32THOR|…     |40ZARA|… |10*201|…|24*4…
```

f) Inserção de um registro de 19 bytes com chave "NINA"

```
LED.head: 189

0    4              82             113            147       189       201
…    76-1RATA|…     |29GHOST|…     |32THOR|…     |40ZARA|… |10*4|…   |19NINA|…|\0…
```

(a sobra de 5 bytes fica como fragmentação interna porque 5 bytes não são suficientes para gravar as informações da LED)

8. Por que a estratégia *worst-fit* não é uma boa escolha quando o espaço é perdido devido à fragmentação interna?

   Porque na estratégia worst-fit a sobra de espaço após uma reutilização será a maior possível, o que aumentaria a fragmentação interna. Dessa forma, quando essa estratégia é utilizada costuma-se retornar o espaço de sobra para a LED.

9. Suponha que um arquivo deva permanecer ordenado. Como isso afeta o gerenciamento de espaços disponíveis?

   Os espaços disponíveis são armazenados em uma LED/PED e os ponteiros dessas estruturas referenciam as posições físicas dos espaços disponíveis. Quando ordenamos o arquivo, os registros podem mudar de RRN/byte-offset, assim como os espaços disponíveis. Dessa forma, após a ordenação, o encadeamento da PED/LED provavelmente estará corrompido.

10. Explique o método de ordenação *keysort* e exemplifique seu funcionamento, passo a passo, para o arquivo representado abaixo (considere que o último registro (chave DGC) ocupa 56 bytes).

Byte-offset	**Chave**	**Título**	**Compositor**
32	LON	Romeo and Juliet	Prokofiev
77	RCA	Quartet in C Sharp Minor	Beethoven
132	WAR	Touchstone	Corea
167	ANG	Symphony No. 9	Beethoven
211	COL	Nebraska	Springsteen

Byte-offset	***Chave***	***Título***	***Compositor***
256	DGC	Symphony No. 9	Beethoven

O keysort é um método de ordenação interna que carrega para a memória apenas as chaves e os endereços dos registros. O método possui 3 passos:

1. Leia o arquivo e coloque em um vetor as chaves e os respectivos RRNs/byte-offset de cada registro
2. Ordene o vetor em memória
3. Reescreva o arquivo de dados segundo a ordem dada pelo vetor ordenado em memória

Para o arquivo do exercício, os passos seriam executados da seguinte forma:

1. Leia o arquivo e coloque em um vetor as chaves e os respectivos byte-offset de cada registro

Vetor em memória

Chave	Byte-offset
LON	32
RCA	77
WAR	132
ANG	167
COL	211
DGC	256

2. Ordene o vetor em memória

Vetor em memória

Chave	Byte-offset
ANG	167
COL	211
DGC	256
LON	32
RCA	77
WAR	132

3. Reescreva o arquivo de dados segundo a ordem dada pelo vetor ordenado em memória

Arquivo em disco ordenado

Byte-offset	***Chave***	***Título***	***Compositor***
32	ANG	Symphony No. 9	Beethoven
76	COL	Nebraska	Springsteen
121	DGC	Symphony No. 9	Beethoven
177	LON	Romeo and Juliet	Prokofiev
222	RCA	Quartet in C Sharp Minor	Beethoven
277	WAR	Touchstone	Corea

Obs.: o tamanho de cada registro foi calculado tendo como referência os bytes-offsets do arquivo original