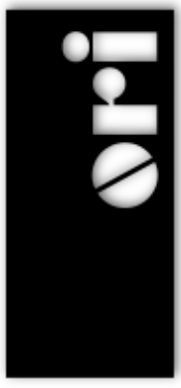

22667– Trabalho 1

Jander Moreira*

Entrega: 4 de setembro de 2018

Conteúdo

1 Introdução e objetivos

O objetivo deste trabalho é fazer a manipulação de leitura de blocos de disco e recuperação de registros de tamanho variável.

2 Contexto

Considere que um sistema de arquivos esteja configurado para recuperação de dados da memória secundária em blocos de 512 bytes[1].

Dentro destes blocos devem ser acondicionados registros inteiros, isto é, um registro fica sempre contido inteiro dentro de um único bloco, sem quebras. Espaços dentro de blocos que não puderem ser usados tornar-se-ão fragmentação externa. Os dois últimos bytes de cada bloco devem ser usados para um inteiro sem sinal, em formato *little-endian*[2], que indica o espaço usado dentro do bloco (bytes válidos).

Os registros a serem armazenados devem possuir todos os mesmos campos, compondo informações sobre um aluno, conforme segue:

- Um código inteiro para o RA;
- O nome completo do aluno;
- A abreviação do curso[3];
- O ano de ingresso, com quatro dígitos.

Cada registro possui tamanho variável e é composto por campos de tamanho variável. Como terminador de campo deve ser usado o caractere de código ASCII 13 (OD_{16}) e como terminador de registro o caractere ASCII 10 ($0A_{16}$). A quantidade de registros por bloco varia conforme o tamanho dos registros que ele contém. Não havendo espaço para um novo registro em um bloco (preservando o controle de espaço), ele deve ser colocado em um novo bloco.

Todos os campos devem ser mantidos no formato texto. Portanto, o RA e o ano devem ser formados pelos dígitos e não ser armazenado em uma representação binária.

*Moreira, J. – Universidade Federal de São Carlos – Departamento de Computação – *Rodovia Washington Luis, km 235 – 13565-905 - São Carlos/SP – Brasil* – jander@dc.ufscar.br

[1] Valor fictício, é claro!

[2] Consultar: https://pt.wikipedia.org/wiki/Extremidade_(ordena%C3%A7%C3%A3o)

[3] Se necessário, consulte a lista de abreviações dos cursos da UFSCar em http://www.spdi.ufscar.br/siglas/nomes-e-siglas-oficiais-dos-cursos-de-graduacao-da-ufscar/view.

3 Especificação

Os alunos devem se organizar em grupos de dois ou três membros. Alunos das turmas A e B da disciplina podem se "misturar". Não serão aceitas submissões individuais.

O grupo deve elaborar um programa que realize as seguintes ações:

- Criação do arquivo e população com dados; e
- Busca por um registro por RA.

A **criação e população** do arquivo é a operação de, a partir de um conjunto de dados já disponíveis, fazer o preenchimento dos blocos do arquivo, gradando-os em memória secundária. A ordem de entrada de dados deve ser preservada. O controle de espaço do bloco deve ser mantido.

Para se fazer a **busca** por um dado registro, deve ser fornecido um valor para o RA e os dados completos do registro devem ser apresentados, usando-se o formato seguinte, terminado com uma mudança de linha. Não devem ser acrescentados espaços ou tabulações. Em caso de falha, uma linha com um único asterisco (`*`) deve ser apresentada.

```
RA:Nome:Sigla:Ingresso
```

O nome do arquivo gerado deve ser o nome fornecido (veja seção 3.1) com extensão `.dat`. Por exemplo: `dados_1.dat`.

3.1 Formatação da entrada

O programa elaborado deve esperar que dos dados sejam digitados de um modo específico, sem variações. Assim, as entradas seguem a seguinte ordem:

- A primeira linha do arquivo contém o nome do arquivo de dados que deverá ser criado (sem extensão);
- A segunda linha contém o número de registros que virão em seguida;
- As linhas seguintes contém os campos de cada registro (um campo por linha; quatro linhas por registro, portanto) na seguinte ordem: RA, nome completo, sigla do curso e ano de ingresso;
- As demais linhas contém os RAs que devem ser consultados no arquivo, um RA por linha;
- A última linha contém o valor zero (0), que é um sentinela que indica o fim das consultas[4].

Alguns arquivos com extensão `.in` estão disponibilizados e exemplificam os dados como serão digitados.

3.2 Sobre o programa

O programa elaborado deve realizar todas as leituras exclusivamente a **partir do teclado**. Todas as mensagens de interface do usuário devem ser direcionadas para a saída de erros[5]. Apenas as saídas das buscas devem ser direcionadas para a saída padrão. Caso se deseje, podem ser omitidas todas as interações com o usuário.

Para cada arquivo `.in` exemplo, há um arquivo `.out` correspondente, o qual contém o resultado que se espera que o programa produza.

3.3 Sobre o domínio dos dados

Considere as seguintes informações:

- Nenhum registros excederá 510 bytes;
- Cada campo de entrada nunca excederá 200 caracteres;
- O número máximo de registros de dados é 2^{31};

3.4 Restrições

O código deve ser elaborado na forma de um arquivo único com o código fonte, usando-se as linguagens C ou C++, usando-se apenas as bibliotecas padrão. Na avaliação, o código fonte será compilado no Linux (Ubuntu 18.04, usando `gcc` ou `g++`). Assegure-se que seu código seja compatível. Em caso

[4] Esta linha não deve ser usada para busca e, portanto, não deve produzir saída.

[5] Em C, sugere-se o uso da função `fprintf`. Por exemplo:

```
fprintf(stderr, "linha a ser ignorada na correcao\n");
printf("linha a ser considerada %d\n", ano);
```

de dúvidas, pergunte antecipadamente no fórum de dúvidas do AVA. Para uso de outra linguagem qualquer, consulte o professor.

Em nenhuma hipótese o arquivo pode ser lido inteiro para a memória principal. A manipulação deve ser sempre bloco a bloco. A violação desta restrição desqualifica o trabalho.

4 O que e quando

O código fonte (arquivo único) deve ser submetido no AVA. Apenas um dos membros do grupo deve submeter o código. A data limite deve ser respeitada; envios posteriores não serão aceitos.

5 Critérios de avaliação

O código será avaliado quanto vários quesitos:

- Qualidade do código fonte;
- Resultado da execução;
- Adequação do formato do arquivo de dados gerado.

Quanto à **qualidade** serão considerados: documentação adequada, uso de variáveis com nomes significativos, indentação adequada, organização do código etc.

Para o **resultado da execução**, a correção da saída (buscas por RAs) será por comparação direta com o resultado esperado. Para cada arquivo `.in`, espera-se como saída o conteúdo exato do arquivo `.out`. Nenhum espaçamento ou linha adicional devem ser incluídos, sob pena de falha na comparação, que será feita por um programa[6].

O **formato do arquivo** `.dat` gerado será avaliado para verificar se está em conformidade com a especificação (seção 2). Espera-se, com isso, que os arquivos gerados possam ser intercambiados entre os grupos.

6 Regras de conduta

Serão aplicadas as regras e sanções previstas no texto "Conduta do aluno"[7].

[6] Será usado o `diff` (https://linux.die.net/man/1/diff).

[7] Drive UFSCar: https://drive.ufscar.br/d/2947e699b2/?p=/00_representacao/conduta_do_aluno&mode=list.