Universidade de São Paulo
Instituto de Ciências Matemáticas e de Computação
Departamento de Ciências de Computação
Disciplina de Organização de Arquivos

docente
Profa. Dra. Cristina Dutra de Aguiar Ciferri (cdac@icmc.usp.br)
alunos PAE
Turma A. Viviana Elizabeth Romero Noguera (viviana.noguera@usp.br)
Turma B. Guilherme Muzzi da Rocha (guilherme.muzzi.rocha@usp.br)
monitores
Turmas A e B. Matheus Carvalho Raimundo (mcarvalhor@usp.br)
Turma B. Gabriel Alfonso Nascimento Salgueiro (gabrielsalgueiro@usp.br)

Primeiro Trabalho Prático

Este trabalho tem como objetivo armazenar dados em um arquivo binário de acordo com uma organização de campos e registros, bem como recuperar os dados armazenados.

*O trabalho deve ser feito **individualmente**. A solução deve ser proposta exclusivamente pelo aluno com base nos conhecimentos adquiridos nas aulas. Consulte as notas de aula e o livro texto quando necessário.*

Descrição de páginas de disco

No trabalho será usado o conceito de páginas de disco. Cada página de disco tem o tamanho fixo de 32.000 bytes. O conceito de página de disco é um conceito lógico, ou seja, deve ser garantido via programação, de forma que cada página de disco contenha, no máximo, o tamanho fixo especificado.

Descrição do arquivo de dados

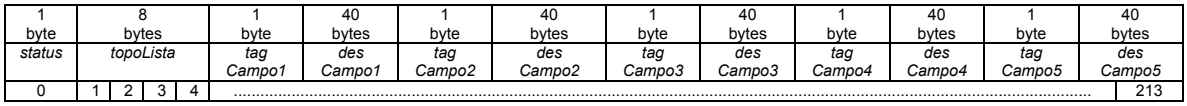
Descrição do Registro de Cabeçalho. O registro de cabeçalho deve conter os seguintes campos:

- *status*: indica a consistência do arquivo de dados, devido à queda de energia, travamento do programa, etc. Pode assumir os valores 0, para indicar que o arquivo de dados está inconsistente, ou 1, para indicar que o arquivo de dados

está consistente. Ao se abrir um arquivo para escrita, seu status deve ser 0 e, ao finalizar o uso desse arquivo, seu status deve ser 1 – tamanho: *string* de 1 byte.

- *topoLista*: armazena o *byte offset* de um registro logicamente removido, ou -1 caso não haja registros logicamente removidos – tamanho: inteiro de 8 bytes.
- *tagCampo1*: valor resumido da *tag* para o campo *idServidor*. Deve assumir o valor i – tamanho: *string* de 1 byte.
- *desCampo1*: descrição completa do campo *idServidor*. Deve assumir o valor 'numero de identificacao do servidor' – tamanho: *string* de 40 bytes.
- *tagCampo2*: valor resumido da *tag* para o campo *salarioServidor*. Deve assumir o valor s – tamanho: *string* de 1 byte.
- *desCampo2*: descrição completa do campo *salarioServidor*. Deve assumir o valor 'salario do servidor' – tamanho: *string* de 40 bytes.
- *tagCampo3*: valor resumido da *tag* para o campo *telefoneServidor*. Deve assumir o valor t – tamanho: *string* de 1 byte.
- *desCampo3*: descrição completa do campo *telefoneServidor*. Deve assumir o valor 'telefone celular do servidor' – tamanho: *string* de 40 bytes.
- *tagCampo4*: valor resumido da *tag* para o campo *nomeServidor*. Deve assumir o valor n – tamanho: *string* de 1 byte.
- *desCampo4*: descrição completa do campo *nomeServidor*. Deve assumir o valor 'nome do servidor' – tamanho: *string* de 40 bytes.
- *tagCampo5*: valor resumido da *tag* para o campo *cargoServidor*. Deve assumir o valor c – tamanho: *string* de 1 byte.
- *desCampo5*: descrição completa do campo *cargoServidor*. Deve assumir o valor 'cargo do servidor' – tamanho: *string* de 40 bytes.

Representação Gráfica do Registro de Cabeçalho. O tamanho do registro de cabeçalho deve ser de 214 bytes, representado da seguinte forma:



Página de disco. O registro de cabeçalho deve ocupar uma página de disco. Seu tamanho é menor do que o tamanho da página de disco. Neste caso, a página de disco deve ser preenchida com caractere '@' até completar o seu tamanho.

Observações Importantes.

- O registro de cabeçalho deve seguir estritamente a ordem definida na sua representação gráfica.
- Os nomes dos atributos também devem seguir estritamente os nomes definidos na especificação dos mesmos.
- Deve ser feita a diferenciação entre o espaço utilizado e o lixo. Para tanto, todas as *strings* devem ser finalizadas com '\0' e o lixo deve ser identificado pelo caractere '@'.

Registro de Dados. Deve ser considerada a *organização híbrida de campos e registros*, da seguinte forma:

- Campos de tamanho fixo e campos de tamanho variável. Para os campos de tamanho variável, deve-se usar o método *indicador de tamanho*.
- Registros de tamanho variável usando o método *indicador de tamanho*.

Observação. Cuidado ao definir a organização do arquivo de dados. Analise os slides com título "Organização híbrida de campos e registros" disponíveis no arquivo http://wiki.icmc.usp.br/images/3/33/SCC0215012018camposRegistros.pdf.

Descrição do Registro de Dados. Cada registro do arquivo de dados deve conter dados relacionados ao cadastro de servidores do Portal da Transparências. Esses dados foram gerados usando dados reais obtidos do Portal da Transparência e também dados sintéticos. Cada registro representa um servidor e contém os seguintes campos:

- Campos de tamanho fixo: 26 bytes
  - *idServidor* – inteiro – tamanho: 4 bytes
  - *salarioServidor* – número de dupla precisão – tamanho: 8 bytes

- *telefoneServidor* – tamanho: 14 bytes, no formato (DD)NNNNN-NNNN

- Campos de tamanho variável:
  - *nomeServidor* – *string* de tamanho variável
  - *cargoServidor* – *string* de tamanho variável

Os dados dos participantes dos servidores são fornecidos juntamente com a especificação deste trabalho prático por meio de um arquivo .csv, sendo que sua especificação encontra-se disponível na página da disciplina. No arquivo .csv, o separador de campos é vírgula (,).

Além de armazenar os dados relacionados aos servidores, cada registro de dados contém campos adicionais para a implementações futuras relacionadas à remoção de registros logicamente removidos. Esses campos são:

- *removido*: indica se o registro se encontra removido ou não. Pode assumir os valores '*', para indicar que o registro é um registro removido, ou '-', para indicar que o registro não é um registro removido – tamanho: *string* de 1 byte.
- *tamanhoRegistro*: indica o tamanho do registro – tamanho: inteiro de 4 bytes.
- *encadeamentoLista*: armazena os *byte offsets* dos registros logicamente removidos – tamanho: inteiro de 8 bytes.

Representação Gráfica do Registro de Dados. O tamanho de cada registro de dados é variável, representado da seguinte forma:

1 byte	4 bytes	8 bytes	4 bytes	8 bytes	14 bytes	4 bytes	1 byte	tamanho variável	4 bytes	1 byte	tamanho variável
removido	*tamanho Registro*	*encadeamento Lista*	*id Servidor*	*salario Servidor*	*telefone Servidor*	indicador de tamanho	*tagCampo4 (deve ter o valor n)*	*nome Servidor*	indicador de tamanho	*tagCampo5 (deve ter o valor c)*	*cargo Servidor*
0	1	2	.....								

Observações Importantes.

- Cada registro de dados deve seguir estritamente a ordem definida na sua representação gráfica.
- Os nomes dos atributos também devem seguir estritamente os nomes definidos na especificação dos mesmos.

- Deve ser feita a diferenciação entre o espaço utilizado e o lixo. Para tanto, todas as *strings* devem ser finalizadas com '\0' e o lixo deve ser identificado pelo caractere '@'.
- O campo *idServidor* não aceita valores repetidos e nem valores nulos. O arquivo .csv com os dados de entrada já garante essa característica.
- Os campos *salarioServidor*, *telefoneServidor*, *nomeServidor* e *cargoServidor* aceitam valores repetidos e valores nulos. O arquivo .csv com os dados de entrada já garante essa característica.
- Para os campos de tamanho fixo, os valores nulos devem ser representados da seguinte forma:
    - se o campo é inteiro ou de dupla precisão, então armazena-se o valor -1
    - se o campo é do tipo *string*, então armazena-se '\0@@@@@@@@@@'
- Para os campos de tamanho variável, os valores nulos devem ser representados da seguinte forma:
    - não devem ser armazenados os campos referentes: (i) ao indicador de tamanho; (ii) à *tag* que representa o dado; e (iii) ao valor do dado.
- Não é necessário realizar o tratamento de truncamento de dados. O arquivo .csv com os dados de entrada já garante essa característica.
- O campo *tamanhoRegistro* deve ser preenchido com o tamanho do registro.
- Neste primeiro trabalho prático, os campos *removido* e *encadeamentoLista* não são utilizados. Para cada registro, esses campos devem ser inicializados da seguinte forma: (i) *removido* deve ser inicializado com o valor '-'; e (ii) *encadeamentoLista* deve ser inicializado com o valor -1.

Página de disco. Os registros de dados não devem ser armazenados na mesma página de disco que o registro de cabeçalho. Adicionalmente, os registros de dados devem ser armazenados em várias páginas de disco, de acordo com a quantidade de registros gerados. Lembre-se que um registro sempre deve estar contido em uma e no máximo uma página de disco. Para tanto, os alunos devem controlar a quantidade de bytes armazenada em cada página de disco, até o limite máximo de bytes da página de disco. Caso sobrem bytes da página de disco sem serem utilizados, esses bytes devem ser preenchidos com o caractere '@' até completar o tamanho da página de disco. Esses

caracteres devem fazer parte do último registro da página, e devem ser contabilizados no seu indicador de tamanho.

Programa

Descrição Geral. Implemente um programa em C por meio do qual o usuário possa obter dados de um arquivo de entrada e gerar um arquivo binário com esses dados, bem como exibir os dados armazenados no arquivo binário. Deve-se levar em consideração a descrição e a organização do arquivo de dados especificados anteriormente.

Importante. A definição da sintaxe de cada comando bem como sua saída devem seguir estritamente as especificações definidas em cada funcionalidade. Para especificar a sintaxe de execução, considere que o programa seja chamado de "programaTrab1". Essas orientações devem ser seguidas uma vez que a correção do funcionamento do programa se dará de forma automática. De forma geral, a primeira entrada da entrada padrão é sempre o identificador de suas funcionalidades, conforme especificado a seguir.

Descrição Específica. O programa deve oferecer as seguintes funcionalidades:

[1] Permita a leitura de vários registros obtidos a partir de um arquivo de entrada (arquivo no formato CSV) e a gravação desses registros em um arquivo de dados de saída. O arquivo de entrada é fornecido juntamente com a especificação do projeto, enquanto que o arquivo de dados de saída deve ser gerado como parte deste trabalho prático.

Entrada do programa para a funcionalidade [1]:

```
1 arquivo.csv  // arquivo de entrada no formato .csv
```

Saída caso o programa seja executado com sucesso:

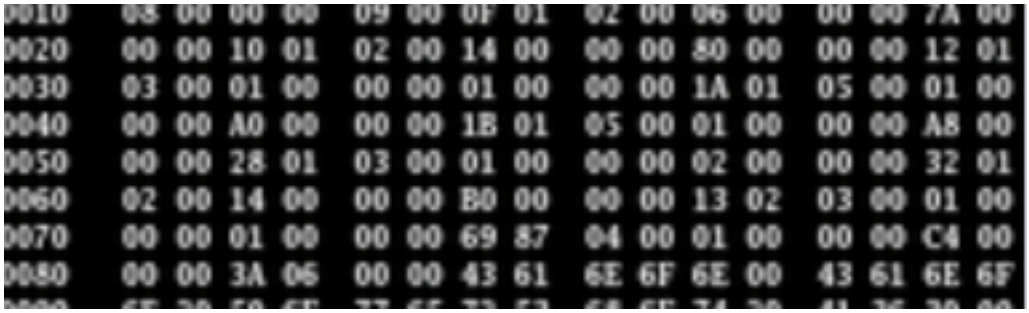
```
Listar na saída padrão o arquivo binário gerado.
```

Mensagem de saída caso algum erro seja encontrado:

```
Falha no carregamento do arquivo.
```

Exemplo de execução:

```
./programaTrab1
1 arquivo.csv
```



[2] Permita a recuperação dos dados, de todos os registros, armazenados no arquivo de dados, mostrando os dados de forma organizada na saída padrão para permitir a distinção dos campos e registros. O tratamento de 'lixo' deve ser feito de forma a permitir a exibição apropriada dos dados. Depois de mostrar todos os registros, deve ser mostrado na saída padrão o número de páginas de disco acessadas.

Entrada do programa para a funcionalidade [2]:

```
2 arquivo.bin   //arquivo binário gerado na funcionalidade [1]
```

Saída caso o programa seja executado com sucesso:

Cada registro deve ser mostrado em uma única linha e os seus campos devem ser mostrados de forma sequencial separado por espaço. Campos de tamanho fixo que tiverem o valor nulo devem ser mostrados também. A quantidade de espaços em branco a ser exibida é a quantidade de bytes do campo. Para os campos com tamanho variável, mostre também seu tamanho em bytes, além dos valores de seus dados. Para os campos de tamanho variável com valores nulos, não deve ser exibido nada. Ao final, deve ser exibido o número de páginas de disco acessadas.

Mensagem de saída caso não existam registros:

```
Registro inexistente.
```

Mensagem de saída caso algum erro seja encontrado:

```
Falha no processamento do arquivo.
```

Exemplo de execução (são mostrados apenas 2 registros)**:**

```
./programaTrab1
2 arquivo.bin
5008717 6092.58 (18)99654-3379 25 FERNANDA TEIXEIRA EITERER 34 ASSISTENTE EM CIENCIA E TECNOLOGIA
8509597 5114.44 (38)98139-8135 30 CARLA BEATRIZ DE CASTRO BARROS 21 AGENTE ADMINISTRATIVO
Número de páginas de disco acessadas: XX
```

[3] Permita a recuperação dos dados de todos os registros que satisfaçam um critério de busca determinado pelo usuário. Por exemplo, o usuário pode solicitar a exibição de todos os registros de um determinado *número de identificação do servidor*. Qualquer campo pode ser utilizado como forma de busca. Essa funcionalidade pode retornar 0 registros (quando nenhum satisfaz ao critério de busca), 1 registro ou vários registros.

Os dados solicitados devem ser mostrados na saída padrão da seguinte forma. Para cada registro, mostre os metadados, seguidos por dois pontos (:), seguidos de um espaço em branco, seguidos pelos valores de seus campos, de forma que cada campo apareça em uma linha diferente. Os metadados devem ser lidos do registro de cabeçalho (campos *desCampo1*, *desCampo2*, *desCampo3*, *desCampo4*, *desCampo5*). Caso o valor seja nulo, escreva 'valor nao declarado'. Depois do registro, deve-se pular uma linha em branco. Depois de mostrar todos os registros, deve ser mostrado na saída padrão o número de páginas de disco acessadas.

Sintaxe do comando para a funcionalidade [3]:

```
3 arquivo.bin NomeDoCampo valor
```

Saída caso o programa seja executado com sucesso:

Para cada registro, devem ser exibidos seus campos da seguinte forma. Cada campo deve ser exibido em uma linha diferente. Primeiro, deve ser colocado o valor do metadado para aquele campo, e depois o seu valor. Campos com valores nulos devem ser representados por 'valor nao declarado'. Depois de cada registro, pule uma linha em branco. Deve ser exibido o número de páginas de disco acessadas.

Mensagem de saída caso não seja encontrado o registro que contém o valor do campo ou o campo pertence a um registro que esteja removido:

```
Registro inexistente.
```

Mensagem de saída caso algum erro seja encontrado:

```
Falha no processamento do arquivo.
```

Exemplo de execução:

```
./programaTrab1
3 arquivo.bin idServidor 6715183
numero de identificacao do servidor: 6715183
salario do servidor: 4652.43
telefone celular do servidor: (58)99957-9775
nome do servidor: valor nao declarado
cargo do servidor: TECNICO EM ENFERMAGEM
--- pular uma linha em branco ----
Número de páginas de disco acessadas: 2
```

Restrições

As seguintes restrições têm que ser garantidas no desenvolvimento do trabalho.

[1] O arquivo de dados deve ser gravado em disco no **modo binário**. O modo texto não pode ser usado.

[2] Os dados do registro descrevem os nomes dos campos, os quais não podem ser alterados. Ademais, todos os campos devem estar presentes na implementação, e nenhum campo adicional pode ser incluído. O tamanho e a ordem de cada campo deve obrigatoriamente seguir a especificação.

[3] Deve haver a manipulação de valores nulos, conforme as instruções definidas.

[4] Não é necessário realizar o tratamento de truncamento de dados.

[5] Devem ser exibidos avisos ou mensagens de erro de acordo com a especificação de cada funcionalidade.

[6] Os dados devem ser obrigatoriamente escritos e lidos campo a campo. Ou seja, não é possível escrever e ler os dados registro a registro.

[7] O(s) aluno(s) que desenvolveu(desenvolveram) o trabalho prático deve(m) constar como comentário no início do código (i.e. NUSP e nome do aluno). Para trabalhos desenvolvidos por mais do que um aluno, não será atribuída nota ao aluno cujos dados não constarem no código fonte.

[8] Todo código fonte deve ser documentado. A **documentação interna** inclui, dentre outros, a documentação de procedimentos, de funções, de variáveis, de partes do código fonte que realizam tarefas específicas. Ou seja, o código fonte deve ser

documentado tanto em nível de rotinas quanto em nível de variáveis e blocos funcionais.

[9] A implementação deve ser realizada usando a linguagem de programação C. As funções das bibliotecas <stdio.h> devem ser utilizadas para operações relacionadas à escrita e leitura dos arquivos. A implementação não pode ser feita em qualquer outra linguagem de programação. O programa executará no [run.codes].

Fundamentação Teórica

Conceitos e características dos diversos métodos para representar os conceitos de campo e de registro em um arquivo de dados podem ser encontrados nos *slides* de sala de aula e também nas páginas 96 a 107 do livro *File Structures* (*second edition*), de Michael J. Folk e Bill Zoellick.

Material para Entregar

Arquivo compactado. Deve ser preparado um arquivo .zip contendo:

- Código fonte do programa devidamente documentado.
- Makefile para a compilação do programa.

Instruções de entrega. A entrega deve ser feita via [run.codes]:

- página: https://run.codes/Users/login
- código de matrícula para a **Turma A**: **SRAZ**
- código de matrícula para a **Turma B**: **29QF**

Critério de Correção

Critério de avaliação do trabalho. Na correção do trabalho, serão ponderados os seguintes aspectos.

- Corretude da execução do programa.
- Atendimento às especificações do registro de cabeçalho e dos registros de dados.
- Atendimento às especificações da sintaxe dos comandos de cada funcionalidade e do formato de saída da execução de cada funcionalidade.
- Qualidade da documentação entregue.

Restrições adicionais sobre o critério de correção.

- A não execução de um programa devido a erros de compilação implica que a nota final da parte do trabalho será igual a zero (0).
- O não atendimento às especificações do registro de cabeçalho e dos registros de dados implica que haverá uma diminuição expressiva na nota do trabalho.
- O não atendimento às especificações de sintaxe dos comandos de cada funcionalidade e do formato de saída da execução de cada funcionalidade implica que haverá uma diminuição expressiva na nota do trabalho.
- A ausência da documentação implica que haverá uma diminuição expressiva na nota do trabalho.
- A inserção de palavras ofensivas nos arquivos e em qualquer outro material entregue implica que a nota final da parte do trabalho será igual a zero (0).
- Em caso de plágio, as notas dos trabalhos envolvidos serão zero (0).

Data de Entrega do Trabalho

Na data especificada na página da disciplina.

Bom Trabalho !