UNIVERSIDADE FUMEC/FACE
Faculdade de Ciências Empresariais

Curso: Ciência da Computação – BH
Disciplina: ED-I – Professor: Ricardo Terra

INFORMAÇÕES SOBRE ESTA ATIVIDADE:

- Esta atividade valerá 2,0 pontos (*1,0 ponto de trabalho mais 1,0 ponto de AAI*).
- A atividade deve se iniciar no laboratório e ser entregue pelo SINEF **até dia 23/out/2012**.
- Deve ser entregue um arquivo compactado chamado **`Ativ05.zip`** com um arquivo chamado **`grupo.txt`** contendo o nome dos integrantes e os arquivos **`c`** cujos nomes devem seguir o seguinte padrão: **`Ativ05Ex01.c, Ativ05Ex02.c,...`** **Isto será avaliado!**
- Apenas **um** do grupo deverá entregar. **Isto também será avaliado!**
- Atividade não serão recebidas/pontuadas **após o prazo de entrega**. Portanto, fique atento ao prazo.
- **Cópias** (total ou parcial) serão penalizadas com **nota zero**.
- A atividade é em grupo de, no máximo, **dois** alunos.

Atividade 05 - Lista de Exercícios – Estrutura

Dica: É boa prática a utilização de protótipos e a declação de estruturas logo abaixo aos includes.

1. Crie um programa C ANSI que declare uma estrutura data que armazene o dia, mês e o ano como números inteiros. O programa deve ler os três membros da estrutura e depois imprimir a data, porém com o mês por extenso.

   Por exemplo: dia (2), mês (3) e ano (2005) → 2 de março de 2005

2. Crie um programa C ANSI que declare:
   - uma estrutura chamada **`disciplina`** que possua o nome da disciplina e o nome do professor responsável;
   - uma estrutura **`aluno`** que possa a matrícula, nome e a disciplina que o aluno cursa (observe que é uma estrutura dentro da outra).

   Então, devem ser lida todas as informações da estrutura aluno e imprimir algo como:

   ```
   A matrícula XXXX pertence ao aluno(a) XXXX que momentaneamente está cursando a disciplina XXXX com o professor(a) XXXX.
   ```

3. Crie um programa C ANSI que declare uma estrutura chamada **`aluno`** que armazene a matrícula, nome, idade, telefone e uma variável para indicar se o registro está ativo ou não. Deve ser declarado um arranjo de alunos de N posições no método **`main`** e ser passado para todas as funções.

   O programa deverá realizar operações básicas como salvar, alterar, excluir e pesquisar um aluno. Devem estar separadas em métodos (*use protótipos*).

   Ao iniciar o programa, o usuário deve indicar o que deseja fazer por meio do menu de opções abaixo:

```
========================
   CADASTRO DE ALUNOS
========================
1 - SALVAR
2 - ATUALIZAR
3 - EXCLUIR
4 - PESQUISAR
5 - LISTAR
6 - ENCERRAR

DIGITE A OPÇÃO DESEJADA:
```

1 - Salvar
Varrerá o arranjo em busca da primeira posição livre, ao encontrar pedirá ao usuário todas as informações do aluno e gravará no arranjo.
Retornará ao menu de opções.

2 - Atualizar
Pedirá ao o usuário para digitar o número da matrícula.
Caso exista, irá pedir todas as informações do aluno novamente com a exceção da matrícula e gravará no arranjo. Caso não exista, irá avisar ao usuário que não existe aluno com a matrícula informada.
Retornará ao menu de opções.

3 - Excluir
Pedirá para o usuário digitar o número da matrícula.
Caso exista, irá marcar como inativo e avisar ao usuário que a exclusão foi efetuada com sucesso. Caso não exista, isto deverá ser avisado ao usuário.
Retornará ao menu de opções.

4 - Pesquisar
Pedirá para o usuário digitar o número da matrícula.
Caso exista, irá exibir as informações do aluno. Caso não exista, isto deverá ser avisado ao usuário.
Retornará ao menu de opções.

5 - Listar
Se existir algum aluno, irá exibir as informações de todos os alunos ativos. Caso não exista nenhum aluno, isto deverá ser avisado ao usuário.

6 - Encerrar
Deverá encerrar o aplicativo. Pode-se utilizar a função **`int exit( int code )`**.
Ex.: `exit(0);`

Dicas para este exercício:

- No início da aplicação, todos os alunos do arranjo devem estar desativados.
  - Deve ser feito um for desativando todos.
- O *flag* pode ser uma variável inteira pequena (*short int*)
  - 0 indica inativo
  - 1 indica ativo