**Dadas as seguintes estruturas:**

|  |  |
| --- | --- |
| **Estrutura Aluno**  início  inteiro Codigo;  texto Nome[50];  texto CodigoDisciplina[6];  inteiro Faltas;  real Notas[3];  real Media;  real Exame;  real NotaFinal;  texto Resultado[10];  fim | **Estrutura Disciplina**  início  texto CodigoDisciplina[6];  texto Disciplina[30];  fim |

**Construa:**

1. Um vetor do tipo **Estrutura Disciplina** para conter 15 disciplinas
2. Um vetor do tipo **Estrutura Aluno** para conter a matrícula de 40 alunos em 1 disciplina
3. Uma função para ler os dados das 15 disciplinas (Código da Disciplina e Disciplina)
   1. void lerdisciplina(tp\_disciplina[], int) *o primeiro parâmetro é o vetor de disciplinas e o segundo o tamanho do vetor*

* Não deve haver códigos de disciplina repetidos e os mesmos devem ser informados pelo usuário
  1. A verificação de códigos não repetidos deve ser realizada por uma função
     1. int verificadisciplina(tp\_disciplina[], int) *o primeiro parâmetro é o vetor de disciplinas e o segundo o tamanho do vetor*

1. Uma função para ler os dados de matrícula (Código, Nome, Código da Disciplina, Faltas e 3 Notas) para uma turma de 40 Alunos, cada um matriculado em 1 disciplina
   1. void leraluno(tp\_aluno[], int) *o primeiro parâmetro é o vetor de alunos e o segundo o tamanho do vetor*

* Não deve haver códigos de alunos repetidos e os mesmos devem ser informados pelo usuário
  + A verificação de códigos não repetidos deve ser realizada por uma função
    - int verificaaluno(tp\_aluno[], int) *o primeiro parâmetro é o vetor de alunos e o segundo o tamanho do vetor*
* Quando o Código da Disciplina for inserido no vetor de Alunos deve haver uma verificação para não permitir que um código não existente seja aceito (somente devem ser aceitas disciplinas que existam no vetor de disciplinas)
  + Utilize a função int verificadisciplina(tp\_disciplina[], int) para verificar a existência

1. Uma função que calcula a média (a média deve ser calculada assim que as três notas forem informadas)
2. Uma função que lê a nota do Exame, somente para os alunos cuja média for maior ou igual a 4,0 e menor do que 7,0 (a nota do Exame deve ser lida assim que a média for calculada, caso seja necessário)
3. Uma função que calcula a Nota Final dos alunos de acordo com os seguintes critérios (a Nota Final deve ser calculada logo depois que a Média for calculada, ou logo após o Exame ter sido lido)

Nota Final = Média se Média >= 7,0

Nota Final = Média se Média < 4,0

Nota Final = (Média + EXAME) / 2 se Média >= 4,0 e Média < 7,0

1. Uma função que determina o Resultado Final para cada Aluno (o Resultado Final deve ser determinado assim que a Nota Final tenha sido calculada)

APROVADO - Nota Final maior ou igual a 7,0;

APROVADO - Nota Final maior ou igual a 5,0, se fez Exame;

REPROVADO - Nota Final menor do que 4,0 ou Nota Final menor do que 5,0, se fez Exame.

1. Uma função que escreve os dados de todos os Alunos na tela, de acordo com o exemplo:

Nome ........: João da Silva

Disciplina ..: IF21B Lógica Aplicada

Faltas ......: 13

Notas .......: 7,5 - 6,8 - 8,2

Média .......: 7,5

Exame .......: 0,0

Nota Final ..: 7,5

Resultado ...: APROVADO

1. Uma função que ordena os dados dos alunos pelo Nome, em ordem alfabética crescente
2. Uma função que ordena os dados dos alunos pelo Nota Final, em ordem decrescente
3. Uma função que mostra o conteúdo do vetor de alunos
4. Uma função que mostra o conteúdo do vetor de disciplinas
5. Um programa que utilize as funções escritas e que possua um menu com as seguintes opções
   * + 1. Cadastrar alunos
       2. Cadastrar disciplinas
       3. Listar disciplinas
       4. Listar alunos em ordem alfabética crescente
       5. Listar alunos em ordem de Nota Final decrescente

**Observações:**

* O exercício pode ser resolvido em grupos de até 3 aluno(a)s
* Apenas um dos aluno(a)s deve postar a solução no Moodle
* O arquivo deve possuir um comentário em seu início, contendo o nome dos alunos que desenvolveram a solução:

**/\***

**Alunos:**

**Aluno1**

**Aluno2**

**Aluno3**

**\*/**

* Todas as funções devem possuir seus protótipos declarados antes da função main()
* As structs (registros) devem ser declaradas com a utilização de typedef