UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO – CAMPUS PAU DOS FERROS
BACHARELADO EM TECNOLOGIA DA INFORMAÇÃO
DISCIPLINA: LABORATÓRIO DE ALGORITMOS E ESTRUTURAS DE DADOS I
PROFESSOR: THIAGO RIQUE

PROJETO – 2016.1

Escreva um programa que mantenha um conjunto de registros de alunos (turma) numa lista encadeada. Este programa deve utilizar os módulos descritos nos itens a seguir para oferecer ao usuário as opções de operações apresentadas no menu de interação, executar cada operação escolhida pelo usuário e apresentar resultados ou mensagens de erro, conforme for o caso.

a) Escreva um módulo em C, denominado aluno, que exporte o seguinte:

   i. O tipo Aluno, definido como uma estrutura capaz de conter as seguintes informações sobre um aluno: nome, matrícula, três notas (variando de 0.0 a 10.0) e a média dessas notas.

   ii. A macro CARACTERES_EM_MATRICULA, que representa o número (exato) de caracteres permitidos numa matrícula.

   iii. Uma função que aloca dinamicamente uma estrutura do tipo Aluno, configurando as suas informações com valores passados por parâmetro, exceto a média, que deve ser calculada.
   Protótipo: Aluno* aluno_cria(char* nome, char* matricula, float nota1, float nota2, float nota3);

   iv. Uma função que imprime na saída padrão os dados de um aluno.
   Protótipo: void aluno_imprime(Aluno* aluno);

   v. Uma função que, dado o endereço de uma estrutura do tipo Aluno, permite modificar os seus campos, exceto a matrícula.
   Protótipo: void aluno_edita(Aluno* aluno, char* nome, float nota1, float nota2, float nota3);

   vi. Uma função que retorna o nome de um aluno.
   Protótipo: char* aluno_nome(Aluno* aluno);

   vii. Uma função que retorna a matrícula de um aluno.
   Protótipo: char* aluno_matricula(Aluno* aluno);

   viii. Uma função que retorna a primeira nota de um aluno.
   Protótipo: float aluno_nota1(Aluno* aluno);

   ix. Uma função que retorna a segunda nota de um aluno.

Protótipo: float aluno_nota2(Aluno* aluno);

x. Uma função que retorna a terceira nota de um aluno.
Protótipo: float aluno_nota3(Aluno* aluno);

xi. Uma função que retorna a média de um aluno.
Protótipo: float aluno_media(Aluno* aluno);

xii. Uma função que libera o espaço em memória reservado para uma estrutura do tipo Aluno.
Protótipo: void aluno_libera(Aluno* aluno);

b) Escreva um módulo em C, denominado turma, que exporte o seguinte:

i. O tipo No, definido como uma estrutura capaz de conter as seguintes informações sobre o nó de uma lista encadeada para armazenar um conjunto de registros de alunos: um ponteiro para uma estrutura do tipo Aluno e um ponteiro para o tipo No, representando o próximo nó da lista.

ii. O tipo Turma, definido como um ponteiro para o tipo No.

iii. Uma função que cria uma lista vazia representando a turma.
Protótipo: Turma turma_cria(void);

iv. Uma função que carrega os dados de alunos de um arquivo texto para a estrutura de lista encadeada que representa a turma. Essa função é responsável pela criação da lista, retornando o endereço do primeiro nó.
Protótipo: Turma turma_carrega(FILE* fp);

v. Uma função que insere um aluno na turma, mantendo a lista ordenada em ordem alfabética.
Protótipo: Turma turma_insere(Turma turma, Aluno* aluno);

vi. Uma função que busca um aluno cuja matrícula seja igual à passada como parâmetro. Caso a matrícula não seja encontrada na lista, esta função deve retornar NULL.
Protótipo: No* turma_busca(Turma turma, char* matricula);

vii. Uma função que remove um aluno da turma.
Protótipo: Turma turma_remove(Turma turma, char* matricula);

viii. Uma função que modifique os dados de um aluno armazenado em um nó da lista que representa a turma.
Protótipo: void turma_edita(No* no, char* nome, float nota1, float nota2, float nota3);

ix. Uma função que imprime os dados dos alunos da turma.
Protótipo: void turma_imprime(Turma turma);

x. Uma função que calcula a média da turma.
Protótipo: float turma_media(Turma turma);

xi. Uma função que salva os dados da turma em um arquivo texto em virtude da adição ou remoção de registros.
Protótipo: void turma_salva(FILE* fp, Turma turma);

c) Escreva um módulo em C, denominado leitura, que exporte o seguinte:

i. Uma função que lê o nome de um aluno. A condição de validação de um nome é que ele contenha apenas letras e espaços em branco (você pode criar uma função auxiliar para validação e usar a função para leitura de strings exportada pelo módulo interação a seguir).
Protótipo: char* le_nome(void);

ii. Uma função que lê a matrícula de um aluno. A condição de validação de uma matrícula é que ela contenha apenas dígitos na quantidade exata especificada pela constante CARACTERES_EM_MATRICULA exportada pelo módulo aluno (você pode criar uma função auxiliar para validação e usar a função para leitura de strings exportada pelo módulo interação a seguir).
Protótipo: char* le_matricula(void);

iii. Uma função que lê a nota de um aluno. A condição de validação de uma nota é a seguinte: $0.0 \leq \text{nota} \leq 10.0$ (você pode criar uma função auxiliar para validação e usar a função para leitura de números de ponto flutuante exportada pelo módulo interação a seguir).
Protótipo: float le_nota(void);

[**Sugestão:** utilize as funções isdigit e isalpha do módulo ctype nas validações dos itens i e ii.]

d) Escreva um módulo em C, denominado interacao, que exporte o seguinte:

i. Uma função que faça a apresentação do programa. Você pode incluir uma mensagem de boas-vindas, uma breve descrição do programa, autor e versão.
Protótipo: void apresenta_programa(char* msg);

ii. Uma função que faça a despedida do programa.
Protótipo: void mensagem_despedida(char* msg);

iii. Uma função que limpe a tela.
Protótipo: void limpa_tela(void);

[**Sugestão:** não existe função desta natureza na biblioteca padrão de C, mas usualmente várias implementações possuem uma função denominada clrscr num módulo denominado conio.]

iv. Uma função que lê e descarta caracteres encontrados no buffer de entrada.
Protótipo: void limpa_buffer(void);

v. Uma função que lê e valida uma opção digitada pelo usuário ao interagir com o menu do programa.
Protótipo: int le_opcao(int menor_valor, int maior_valor);

vi. Uma função que lê uma cadeia de caracteres.
Protótipo: char* le_string(void);

vii. Uma função que lê um valor de ponto flutuante.
Protótipo: double le_double(void);

[**Sugestão:** Você pode utilizar a função strtod do módulo stdlib da biblioteca padrão de C, juntamente com a função que lê strings. A função strtod converte uma string para um valor do tipo double. A função encerra o processamento da string logo que encontra um caractere que não pode ser interpretado como parte de um número de ponto flutuante. Seu protótipo é: double strtod(const char* s, char** ptrFinal); neste caso, o argumento ptrFinal apontará para o caractere na string de entrada que causou o encerramento da função.]

viii. Uma função que apresenta um menu com número indeterminado de opções.
Protótipo: void apresenta_menu(int n_itens, int menor_opcao, ...);

O menu de interação deve conter as seguintes opções:

1 – Adicionar um aluno na turma
2 – Remover um aluno da turma
3 – Modificar um registro de aluno
4 – Listar todos os alunos da turma
5 – Imprimir a média da turma
6 – Sair do programa

Nos itens acima já são sugeridos os protótipos das funções de cada módulo. Caso o aluno deseje modificar o protótipo de qualquer função, deve comunicar ao professor com antecedência, apresentando uma justificativa plausível para sua decisão.

O projeto deverá ser apresentado no dia **23/11/2017**.

Bom trabalho!