Trabalho 1 – Laboratório de Programação II

Observações para este trabalho:

- ✓ Data da entrega: 24 de março, até as 23h.
- ✓ Formato da entrega: enviar os arquivos fonte .c (não anexar arquivos executáveis!) num único arquivo zipado para deise@inf.ufsm.br, sendo um arquivo .c para cada questão do trabalho.
- ✓ Trabalho em grupos de 2 alunos. Enviar um email por dupla.
- ✓ Caso não receba confirmação do recebimento do e-mail em 48hs, contate novamente o professor (reclamações posteriores sobre possíveis problemas no envio do email não serão aceitas).
- ✓ Cópias da internet e/ou colegas de outras duplas anulam a nota do trabalho.
- ✓ Trabalhos com erros de compilação não serão corrigidos.
- ✓ Professor usará o Dev-C ou Code Blocks para corrigir os trabalhos. Podem usar qquer IDE para implementar as questões, mas certifique-se de que o arquivo está compilando corretamente em alguma destas IDEs.
- ✓ Obrigatoriamente a solução de vocês deve fazer uso de funções e passagem de parâmetros (eventuais respostas que pareçam corretas na execução, mas não usarem funções e passagem de parâmetros, serão consideradas erradas). Por exemplo, todo o código da questão dentro da função *main* será considerado errado; uso de variáveis globais sem passagem de parâmetros será considerado errado.

Questões sobre os conteúdos: funções, ponteiros, passagem de parâmetros e structs

1. Implemente uma função que calcule as raízes de uma equação do segundo grau, do tipo $ax2 + bx + c = 0$. Esta função deve obedecer ao protótipo:

int raizes (float a, float b, float c, float x1, float*x2)*

Esta função deve ter como valor de retorno o número de raízes reais e distintas da equação. Se existirem raízes reais, seus valores devem ser armazenados nas variáveis apontados por *x1* e *x2*.

2. Implemente uma função que calcule a área da superfície e o volume de uma esfera de raio *r*. Esta função deve obedecer ao protótipo:

*void calc_esfera (float r, float *area, float *volume)*

A área da superfície e o volume são dados, respectivamente, por $4r^2$ e $4r^3/3$

3. Faça um programa que leia a base e a altura de um retângulo (tipo Retângulo). A seguir, chame uma função que calcule e mostre na tela o perímetro e a área deste retângulo.

Sugestão para a definição do tipo Retangulo:

```
struct retangulo
{
        float base;
        float altura;
        float perimetro;
        float area;
};
typedef struct retangulo Retangulo;
```

Considere um vetor de 5 retângulos.

Sugestão para o protótipo da função: *void calculaPeriArea (Retangulo ret);*

4. Desenvolva um algoritmo que efetue a leitura de três valores para os lados de um triângulo, considerando lados como: A, B e C. O algoritmo deverá verificar se os lados fornecidos formam realmente um triângulo (cada lado é menor que a soma dos outros dois lados). Se for esta condição verdadeira, deverá

ser indicado qual tipo de triângulo foi formado: isósceles (dois lados iguais e um diferente), escaleno (todos os lados diferentes) ou equilátero (todos os lados são iguais).

OBS: há diversas formas de estruturar este exercício. Sugestão de estrutura do tipo *Triangulo*:

```
struct triangulo
{
 int ladoA;
 int ladoB;
 int ladoC;
 bool ehTriangulo;
 char tipo[20];
};
typedef struct triangulo Triangulo;
```

Sugestão de protótipo:
O *main* lê os lados e chama a função: *Triangulo verificaTriangulo (Triangulo t).* Esta função verifica se é um triângulo e se for, qual o seu tipo correspondente. Altera os campos *ehTrinangulo* e *tipo* da *struct* triangulo, e retorna a estrutura alterada para o *main*. O *main* imprime se os lados informados compõem um triângulo e qual o seu tipo.

5. Em uma pesquisa de campo, uma editora solicitou os seguintes dados para os entrevistados: sexo, idade e quantidade de livros que leu no ano de 2019. Faça um programa que leia os dados de 500 usuários. Depois, calcule e imprima:

a) A quantidade total de livros lidos pelos entrevistados menores de 10 anos.
b) A quantidade de mulheres que leu 5 livros ou mais.

Deve ser usada a seguinte *struct*:

```
struct usuario
{
        char sexo;
        int idade;
        int qtdade;
};
typedef struct usuario Usuario;
```

Os dados devem ser armazenados em um vetor de usuários.

O programa **deve** ser estruturado nas seguintes funções:

void leVetor (int tamanho, Usuario vet);* //esta funcao recebe o tamanho do vetor e o endereço do primeiro elemento do vetor; e preenche o vetor com dados digitados pelo usuário

int calculaQtidadeLivros (int tamanho, Usuario vet);* // esta função recebe o tamanho do vetor e o endereco do primeiro elemento do vetor; calcula a quantidade total de livros lidos pelos entrevistados menores de 10 anos, e retorna esta quantidade para o main

int calculaQtidadeMulheres (int tamanho, Usuario vet);* // esta função recebe o tamanho do vetor e o endereco do primeiro elemento do vetor; calcula a quantidade de mulheres que leu 5 livros ou mais, e retorna esta quantidade para o main.

Ao final, o *main* mostra na tela as duas quantidades calculadas.

6. Faça um programa que leia as aulas ministradas em uma disciplina. Para cada disciplina informada, considere 5 alunos. Para cada aluno, leia duas notas obtidas pelo aluno na disciplina e o número de aulas assistidas. A seguir, calcule e mostre a média final deste aluno e verifique se ele foi aprovado ou reprovado (média >= 6 e mínimo de 75% de frequência para ser aprovado).

Considere as seguintes estruturas:

```
struct aluno
{
 float nota1;
 float nota2;
 int aulasAssistidas;
 float media;
 bool status;
};
typedef struct aluno Aluno;

struct disciplina
{
 Aluno vet[5];
 int aulasMinistradas;
};
typedef struct disciplina Disciplina;
```