

Aula Prática 8 – 17/05/2018

Preparem os exercícios de forma que:

1. Os arquivos utilizem a extensão **“.c”**
2. Não seja utilizada função **system**(“pause”)
3. A função **printf** deve ser utilizada apenas para imprimir a saída do programa.

Atenção: Para esta lista de exercícios é necessário utilizar funções conforme pedem as atividades, caso o contrário, a nota será **0**.

1) Faça um programa que simule cadastros em uma agenda de eventos. Para isto deverá ser criadas duas estruturas (structs) para representar uma data e um evento. Estas estruturas devem armazenar valores dos tipos mostrados na tabela abaixo:

Nome da estrutura	Campos da estrutura
Data	<ul style="list-style-type: none">• dia – valor inteiro• mes – valor de um tipo enum• ano – valor inteiro
Evento	<ul style="list-style-type: none">• nome – string de até 100 caracteres• local – string de até 100 caracteres• data – valor do tipo Data

O programa deve funcionar da seguinte forma:

1. Inicialmente deve solicitar a leitura de um número inteiro **n**
2. Em seguida deve solicitar a leitura de **n** registros do tipo **Evento**, estes registros devem ser armazenados em um vetor de **Evento** criado utilizando alocação dinâmica de memória. Esta leitura deve ser feita em uma função chamada:

```
void cadastrar_eventos(Evento agenda[], int n);
```

3. Após a leitura dos **n** registros, o programa deve realizar a leitura de um registro do tipo **Data** e imprimir na tela todos os eventos daquela data na ordem que foram cadastrados. Essa impressão deve ser feita usando uma função chamada:

```
void imprimir_eventos(Evento agenda[], Data d, int n);
```

Para a leitura de *strings* deve ser utilizada a função **scanf**, outras funções não permitem a correta avaliação do Prático.

Exemplo:

Entrada:	3 evento1 local1 2 6 2014 evento2 local2 13 7 2014 evento3 local3 13 7 2014 13 7 2014
Saída:	evento2 local2 evento3 local3

- **Entrada:**
 - um numero inteiro **n**,
 - **n** registros, onde cada registro possui duas strings e três valores inteiros (dia, mês e ano),
 - três números inteiros (dia, mês e ano).
- **Saída:** uma das seguintes mensagens:
 - “Nenhum evento encontrado!\n”
 - “%s %s\n”, sendo a primeira *string* o nome do evento e a segunda o nome do local do evento, para cada evento encontrado.

2) Em Aprendizado de Máquina uma forma simples, e ainda efetiva, de resolver alguns problemas de classificação é encontrar o registro de treinamento mais parecido com o registro que se deseja classificar. Para medir o quão parecido um registro com valores numéricos é de outro, podemos utilizar a distância euclidiana dada pela fórmula abaixo:

$$\sqrt{\sum_{i=1}^n (p_i - q_i)^2}$$

onde p_i e q_i representam elementos dos vetores $P(p_1, p_2, \dots, p_n)$ e $Q(q_1, q_2, \dots, q_n)$ respectivamente.

Neste exercício você deverá fazer um programa para identificar o tipo de uma flor do gênero íris, esse é um problema clássico utilizado para avaliar métodos de classificação (<http://archive.ics.uci.edu/ml/datasets/Iris>). Para isto o programa deve:

1. Implementar uma estrutura chamada **iris** que contém os seguintes campos:
 - *comprimento da sépala* – valor numérico real
 - *largura da sépala* – valor numérico real
 - *comprimento da pétala* – valor numérico real
 - *largura da pétala* – valor numérico real
 - *tipo* – string de até 50 caracteres
2. Realizar a leitura de um número inteiro **n**
3. Realizar a leitura de **n** registros do tipo **iris**, para isto deve ler primeiro os 4 valores reais e depois uma string.
4. Realizar a leitura de 4 valores reais, representando os dados de uma flor que não foi

identificada.

5. Encontrar o registro que possui a menor distância euclidiana em relação aos valores lidos e imprimir o tipo da flor.

A distância euclidiana deve ser calculada considerando os 4 valores reais dos registros. O programa deve implementar e usar a função definida pelo protótipo abaixo:

```
void classificar(iris *nao_identificada, iris registros_identificados[], int n);
```

Esta função recebe como parâmetro um ponteiro para o registro **iris** que contém os valores lidos da flor não identificada e um vetor de **iris** que contem os **n** registros lidos no início do programa.

Exemplo:

Entrada:	3 5.1 3.5 1.4 0.2 Iris-setosa 5.9 3.2 4.8 1.8 Iris-versicolor 6.5 3.2 5.1 2.0 Iris-virginica 5.0 2.0 3.5 1.0
Saída:	Tipo de flor: Iris-versicolor

- **Entrada:**
 - um número inteiro **n**,
 - **n** registros do tipo **iris**, cada registro formado por 4 valores reais e uma string,
 - 4 valores reais.
- **Saída:** a seguinte mensagem:
 - “Tipo de flor: %s\n”.