Universidade Federal da Bahia - UFBA
Instituto de Matemática - IM
Departamento de Ciência da Computação - DCC
Curso de Bacharelado em Ciência da Computação

MATA40 - Estrutura de Dados e Algoritmos Período: 2019.2 Data: 13/08/2019.

Prof. Antonio L. Apolinário Jr.

Roteiro do Laboratório 1 - Vetores e Registros

Objetivos:

- Compreender os conceitos de Agregados Homogeneos (Vetores) e Heterogeneos (Registros);
- Reforçar os conceitos básicos de estruturas de controle em linguagem C;
- Implementar, em linguagem C, programas que manipulam vetores e registros.

Conceitos básicos:

Arranjo:

Estrutura de dados que armazena uma coleção de elementos de tal forma que cada um dos elementos pode ser identificado por um ou mais índices. Arranjos unidimensionais são usualmente denominados **vetores**, enquanto que aqueles de dimensão maior ou igual a 2 são chamados de **matrizes**.

Usualmente, todos os elementos do arranjo possuem mesmo tamanho e tipo de dados. Eles podem ser acessados individualmente e instantaneamente por sua posição no arranjo, dada por um índice. O índice geralmente utiliza uma sequência de números inteiros, mas o índice pode ter qualquer valor ordinal. Em linguagem C o índice de um vetor sempre inicia em 0. A seguir um exemplo, em linguagem C, de declaração e acesso aos elementos de um vetor.

```
int a[50]; // Vetor
a[20] = 20;
a[44] = a[12] * a[33];
```

Registro:

Estrutura de dados que armazena um grupo de elementos, cada qual associado a um identificador, que podem ser acessados de forma agrupada. Cada elemento do registro é denominado campo. Seu acesso a cada campo pode ser feito a partir do operador . (ponto). Os campos não precisam tem mesmo tamanho ou mesmo tipo, daí sua denominação de agregado heterogêneo. Em linguagem C um registro é identificado pela palavra reservada **`struct`**. A seguir um exemplo de declaração de um registro e acesso aos seus campos.

```
struct pessoa  {    int       matricula;
                    char*     nome;
                    float     salario;
struct pessoa p;
p.matricula = 123;
p.nome = "Maria";
p.salario = 13000.89;
```

Roteiro:

1. Baixe do Moodle os códigos fontes desse Laboratório. Descompacte, compile os códigos e rode os exemplos.

2. Analise o código de **`vetor.c`**. Esse programa cria um vetor e o preenche com valores aleatórios, imprimindo seu conteúdo. Tire todas as suas dúvidas quanto ao seu funcionamento antes de prosseguir para o próximo item.

3. Modifique o programa **`vetor.c`** para que alguns dados estatísticos do vetor sejam calculados: sua média, desvio padrão, maior valor e menor valor. Compile e teste seu código garantindo que os cálculos estão corretos.

4. Analise a complexidade de cada um dos algoritmos de cálculo feitos no item 3.

5. Altere o programa do item 3 para que o número de elementos aleatórios a serem gerados seja uma escolha do usuário do programa. Lembre-se de que o usuário não pode escolher mais elementos do que foram especificados na definição do vetor. Compile e teste seu código garantindo que vetores de tamanhos diferentes podem ser gerados a cada interação.

6. Modifique o programa do item 5 para que o usuário possa fornecer um número **`n`**, e o programa determine se **`n`** pertence ou não ao conjunto de valores armazenados no vetor. Compile e teste seu código.

7. Analise a complexidade do algoritmo de busca que voce implementou no item 6.

8. Avalie o algoritmo de busca que você implementou no item 6, contabilizando o número de comparações da sua implementação com o valor teórico encontrado no item 7. Para tanto varie a quantidade de elementos do vetor com valores relativamente grandes.

9. Analise o código de **`registro.c`**. Esse programa cria um registro para armazenar uma data. Uma variável é criada e os seus valores determinados pelo usuário. A data fornecida é apresentada ao usuário no formato DD/MM/AA. Tire todas as suas dúvidas quanto ao seu funcionamento antes de prosseguir para o próximo item.

10. Modifique o programa **`registro.c`** para que apenas datas válidas possam ser fornecidas. Compile e teste seu código.

11. Altere o programa do item 10 para que duas datas válidas possam ser fornecidas e que seja calculado o intervalo em dias entre as duas datas. Para facilitar, considere que todos os meses têm 30 dias. Compile e teste seu código.

12. Crie no programa do item 10 um novo registro de nome **`struct periodo`**, cujos campos são **`data_inicio`**, **`data_fim`** e **`numero_dias`**, respectivamente dos tipos **`struct data`** e **`int`**. Modifique o programa para que os cálculos sejam feitos com esse novo registro. Compile e teste seu código.

13. Pesquise como na linguagem C é possĩvel criar um tipo definido pelo usuário. Aplique esse recurso da linguagem para modificar o programa do item 10, criando dois tipos: **`tData`** e **`tPeriodo`**. Compile e teste seu novo programa.

14. A partir do programa do item 12, crie um programa capaz de registrar diversos períodos, armazenados em um vetor. Promova o preenchimento automático dos valores utilizando uma variante da função de geração de números aleatórios do programa **`vetor.c`**, com o cuidado de gerar apenas valores válidos para os campos dos tipos **`tData`**. Compile e teste seu programa.

15. Modifique o programa do item 13 para que, após o preenchimento, o usuário possa fornecer uma data e o programa indique se existe algum intervalo que contém essa data. Compile e teste seu programa.

Desafios:

1. Avalie se existem alternativas melhores para o algoritmo de busca que você implementou no item 6. Se for o caso, implemente esse outro algoritmo e avalie seu desempenho (em termos de número de comparações) e compare com o algoritmo original e sua classe de complexidade.
2. Melhore o programa do item 12 para que o cálculo do período entre datas leve em conta não só a variação de dias entre os meses mas também os anos bisextos.