**Resumo sobre o que foi passado nas aulas de estrutura de dados**

**1 – Vetor e Matriz**

1 - Estrutura de dados Homogênea ou Matriz é uma coleção de variáveis de mesmo tipo, acessíveis com um único nome e armazenados contiguamente (um após o outro) na memória.

2 - A individualização de cada variável é feita através do uso de índices.

3 - Os Vetores são matrizes de uma só dimensão, isto é, necessita apenas 1 índice para acesso as variáveis.

4 - As Matrizes possuem mais de uma dimensão, isto é, necessitam de um índice para cada dimensão para acesso as variáveis.

**2 - Ponteiros**

Input:

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

Um array de caracteres chamado palavra é declarado com espaço para 60 caracteres. Em seguida, o programa pede ao usuário que digite algo e usa a função scanf() para ler a entrada do usuário e armazená-la no array palavra. O formato de conversão "%[^\n]s" é usado para ler todos os caracteres digitados pelo usuário, exceto o caractere de nova linha "\n", que é inserido pelo usuário ao pressionar Enter.

Observe que não é necessário usar o operador "&" antes do nome do array palavra em scanf(), pois palavra é um ponteiro para o primeiro elemento do array, então ele já contém o endereço da primeira posição do array.

Output:

Texto

Descrição gerada automaticamente

Observe que os endereços da &palavra e palavra são diferentes. O primeiro se refere ao endereço da variável palavra na memória, enquanto o segundo se refere ao endereço do primeiro elemento do array palavra.

Input:

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

O operador \*(ponteiro+i) é utilizado para acessar o valor da posição i do vetor a partir do ponteiro que aponta para o vetor.

Em outras palavras, o ponteiro é inicializado com o endereço da primeira posição do vetor. Ao usar o operador de referenciamento \* em conjunto com o ponteiro e um índice i, podemos acessar o valor armazenado na posição i do vetor.

A expressão \*(ponteiro+i) é equivalente a ponteiro[i], pois ambos acessam o valor da posição i do vetor. Entretanto, a primeira forma é mais utilizada quando se trabalha com ponteiros em vez de arrays.

O uso de \*(ponteiro+i) em vez de ponteiro[i] pode ajudar a lembrar que estamos trabalhando com ponteiros e que o operador de referenciamento \* é necessário para acessar o valor da posição do vetor apontada pelo ponteiro.

Output:

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

**3 – Dados Heterogêneos**

Uma estrutura que apresenta mais de um tipo básico de dado é denominada heterogênea.

Uma imagem contendo Aplicativo

Descrição gerada automaticamente

Linha do tempo

Descrição gerada automaticamente

Para definir uma struct, iniciamos com typedef struct e passamos os parâmetros dentro do seu bloco:

Texto

Descrição gerada automaticamente

Depois temos que chamar o stuct dentro da nossa classe main, mais ou menos parecido com POO:

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

Na frente da referência Carro, declaramos o nome, e toda vez que formos utilizar os parâmetros da classe Carro, temos que referencia-la pelo nome

Podemos também, adicionar uma struct dentro de outra para acessar os seus dados:

Texto

Descrição gerada automaticamente

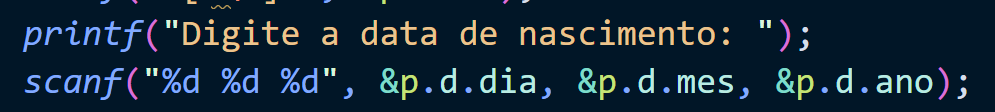
Neste exemplo, vemos o uso de uma função utilizando struct:

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

A função é do tipo Pessoa, e tem como objetivo receber os dados das pessoas, e retorna os dados no final, para que possamos referenciá-la

Podemos também adicionar mais de um valor na entrada de dados, com podemos ver neste exemplo:



Outra função, só que dessa vez, ela recebe a struct nos parâmetros da função:

Texto

Descrição gerada automaticamente

Ela retorna apenas os dados no final do programa, por isso ela é void

Na classe main, utilizaremos do vetor, para armazenar as pessoas, para isso declaramos o vetor do tipo Pessoa, já que Pessoa é um struct e o vetor vai receber diferentes tipos de dados.

Texto

Descrição gerada automaticamente

Utilizamos da estrutura de repetição para alimentarmos o vetor homogêneo, e após atingir o tamanho, ele pula para a próxima estrutura de repetição, assim imprimindo os resultados cadastrados.