3-ARRAYS E PONTEIROS

- Vimos 2 tipos de arrays
    - Vetores (uni-dimensionais)
    - Matrizes (multi-dimensionais)

- Vamos fazer um progama.

```c
int main(){//inicio_main
    printf("\n");


    int valores[5];

    //recebendo resultado
    for (int i = 0; i < 5;i++){//inicio_for1
        printf("Informe o valor %d/5 valor: ",(i+1));
        scanf("%d",&valores[i]);
    }//fim_for1

    printf("Os valores informados foram: \n");

    //imprimindo resultados
     for (int i = 0; i < 5;i++){//inicio_for2
        printf("%d\n", valores[i]);

    }//fim_for2

    printf("\n");
    printf("\n");
    return 0;
}//fim_main
```

```
Informe o valor 1/5 valor: 2
Informe o valor 2/5 valor: 6
Informe o valor 3/5 valor: 8
Informe o valor 4/5 valor: 1
Informe o valor 5/5 valor: 9
Os valores informados foram:
2
6
8
1
9
```

- O que tem haver ponteiros e arrays ...
- Quando estamos declarando um vetor (array) do tipo inteiro, a linguagem C cria um ponteiro(invisivel) que aponta para o primeiro endereço de memoria do array.
    - Quando fazemos a declaração de um vetor, o compilador pega o primeiro endereço de memoria do vetor e coloca um ponteiro apontando para ele. Assim que ele tem o controle de quais são os elementos do array.

```
printf("Endereco de memoria(0): %p\n valor(0) : %d\n",&valores[0], valores[0]);
printf("Endereco de memoria: %p\n valor(0) : %d",valores, valores[0]);

printf("Endereco de memoria(1): %p\n valor(0) : %d\n",&valores[1], valores[1]);
printf("Endereco de memoria: %p\n valor(0) : %d",valores, valores[1]);
```

```
Endereco de memoria(0): 0061FF04
 valor(0) : 1
Endereco de memoria(vetor): 0061FF04
 valor(0) : 1

Endereco de memoria(1): 0061FF08
 valor(1) : 2
Endereco de memoria(vetor): 0061FF04
 valor(1) : 2
```

```
printf("Endereco de memoria(0): %p\n valor(0) : %d\n",&valores[0], valores[0]);
printf("Endereco de memoria(vetor): %p\n valor(0) : %d\n\n",valores, valores[0]);

printf("Endereco de memoria(1): %p\n valor(1) : %d\n",&valores[1], valores[1]);
printf("Endereco de memoria(vetor): %p\n valor(1) : %d",valores, valores[1]);
```

- O endereço de memoria na posição 0 é o mesmo endereço de memoria da variavel.

```
Endereco de memoria(0): 0061FF00
 valor(0) : 1
Endereco de memoria(vetor): 0061FF00
 valor(0) : 1

Endereco de memoria(1): 0061FF04
 valor(1) : 2
Endereco de memoria(vetor): 0061FF00
 valor(1) : 2

Endereco de memoria(2): 0061FF08
 valor(2) : 3
Endereco de memoria(vetor): 0061FF00
 valor(2) : 3

Endereco de memoria(3): 0061FF0C
 valor(3) : 4
Endereco de memoria(vetor): 0061FF00
 valor(3) : 4

Endereco de memoria(4): 0061FF10
 valor(4) : 5
Endereco de memoria(vetor): 0061FF00
 valor(4) : 5
```

```c
}//fim_for2


for(int i  = 0; i < 5; i++){
    printf("Endereco de memoria(%d): %p\n valor(%d) : %d\n",i, &valores[i], i, valores[i]);
    printf("Endereco de memoria(vetor): %p\n valor(%d) : %d\n\n",valores, i, valores[i]);

}
```