Funções

Exercício:

Escreva o código fonte de um programa, na linguagem C, que manipule um vetor de inteiros com dez elementos. O programa deve possuir uma função que receba o vetor e retorne o maior valor contido no mesmo. As seguintes manipulações devem ser feitas: o vetor deve ser inicializado e, por meio da utilização da função mencionada, o maior valor contido no vetor deve ser impresso na saída padrão.

```c
#include <stdio.h>
#define n 10
int maior_valor (int vetor[n]) (int *vetor)
{
   int i,maior_v=vetor[0];
   for(i=1;i<n;++i)
      if(maior_v<vetor[i])
         maior_v=vetor[i];
   return maior_v;
}
main()
{
   int i,v[n];
   for(i='\0';i<n;++i)
   {
      printf("\nEntre com o elemento v[%d]: ",i+1);
      scanf("%d",v+i);
   }

   return 0;
}
```

Funções

- Variáveis globais

Variáveis globais são declaradas fora de todas as funções do programa. Elas são conhecidas e podem ser alteradas por todas as funções do programa que seguem sua declaração.

Quando uma função tem uma variável local com o mesmo nome de uma variável global a função dará preferência à variável local. Vamos ver um exemplo. Observação: O mesmo vale para parâmetros formais.

```
int z,k;
func1 (...)
{
    int x,y;
    ...
}
void func2 (...)
{
    int x,y,z;
    ...
    z=10;
    ...
}
main ()
{
    int count;
    z=7;
    func2(...);
    ...
}
```

Exercício:

Analise o programa abaixo e indique o que será impresso na saída padrão.

Precedência (Hierarquia nas operações)

Hierarquia	Operação
1	Parênteses
2	Função
3	++,--
4	- (menos unário)
5	*, /, %
6	+, -

```c
#include <stdio.h>
int num;
int func(int a, int b)
{
    a = (a+b)/2;  = (0+50)/2  = 25
    num -= a+1;  = num – (a+1)  = 10 – (25+1)  = – 16
    return a;
}
int main()
{
    int first = 0, sec = 50;
    num = 10;                10                0                50
    printf("\nnum antes = %d\tfirst antes = %d\tsec antes = %d", num, first, sec);
    num += func(first, sec); num  +  func(0, 50)   ou seja   -16 +  25   =   9
    printf("\nnum depois = %d\tfirst depois = %d\tsec depois = %d\n", num, first, sec);
}
```

num antes = 10 ? first antes = 0 sec antes = 50
num depois = 9 first depois = 0 sec depois = 50

Funções

Exercício:

Escreva o código fonte de um programa, na linguagem C, que manipule um vetor de inteiros com dez elementos. O programa deve possuir uma função responsável pela inicialização do vetor e outra função que efetue a impressão do vetor na saída padrão. Por meio das funções mencionadas, o programa deve inicializar o vetor e em seguida retorná-lo no monitor.

```c
#include <stdio.h>
#define n 10
int vetor[n];
void inicializar_vetor()
{
    int i;
    for(i='\0';i<n;++i)
    {
        printf("\nEntre com o elemento v[%d]: ",i+1);
        scanf("%d",vetor+i);
    }
}
```

```c
void imprimir_vetor()
{
    int i;
    for(i='\0';i<n;++i)
        if(!i)
            printf("vetor = { %d, ", vetor[i]);
        else
            if(i==n-1)
                printf("%d }", vetor[i]);
            else
                printf("%d, ", vetor[i]);
}
```

```c
int main()
{
    inicializar_vetor();
    imprimir_vetor();
}
```

Funções
Passagem de Parâmetros

Funções

- Passagem de parâmetros por valor e passagem por referência

Vimos que quando chamamos uma função os parâmetros formais desta recebem uma cópia dos valores dos parâmetros que são passados para a mesma. Isto quer dizer, que alterações nos parâmetros formais, que ocorrem dentro da função, não geram alterações nos valores dos parâmetros externos à função.

UNIVASF

Funções

- Passagem de parâmetros por valor e passagem por referência (continuação)

Este tipo de passagem de parâmetros é denominado passagem por valor. Isto ocorre, porque são passados para a função apenas os valores dos parâmetros e não os próprios parâmetros. Vejamos o exemplo a seguir:

```c
#include <stdio.h>
float sqr (float num)
{
    num=num*num;
    return num;
}
int main ()
{
    float num,sq;
    printf ("Entre com um numero: ");
    scanf ("%f",&num);
    sq=sqr(num);
    printf ("\n\nO numero original e: %f\n",num);
    printf ("O seu quadrado vale: %f\n",sq);
}
```

Funções

- Passagem de parâmetros por valor e por referência (continuação)

Outro tipo de passagem de parâmetros para uma função ocorre quando alterações nos parâmetros formais, dentro da função, alteram os valores dos parâmetros que foram passados para a função. Este tipo de chamada de função tem o nome de "passagem por referência".

Funções

- Passagem de parâmetros por valor e por referência (continuação)

A linguagem C só permite passagem por valor.

Isto é bom quando queremos usar os parâmetros formais à vontade dentro da função, sem termos que nos preocupar em estar alterando os valores das variáveis que tiveram seus valores passados aos parâmetros da função.

Mas, isto também pode ser ruim, porque, às vezes, podemos desejar que mudanças nos valores dos parâmetros formais reflitam nas variáveis que tiveram seus valores copiados para os parâmetros formais da função.

UNIVASF

Funções

- Passagem de parâmetros por valor e passagem por referência (continuação)

Quando queremos alterar, dentro das funções, as variáveis que tiveram seus valores passados para uma função, nós podemos declarar os parâmetros formais como sendo *ponteiros*.

O ponteiro é a "referência" que precisamos para poder alterar dentro da função uma variável externa à função. O único inconveniente é que, quando usarmos uma função, que se utiliza deste artifício, teremos de lembrar de colocar um **&** na frente da variável que estivermos passando sua referência para a função.

Veja um exemplo:

```c
#include <stdio.h>
void incrementa (int *a,int b)
{
    *a+=b;
}
int main (void)
{
    int num;
    num=100;
    printf ("\nValor de num antes da chamada %s%d",
    "da funcao:  ", num);
    incrementa (&num,50);
    printf ("\nValor de num depois da chamada %s%d",
    "da funcao:  ", num);
}
```

Funções

Exercício:

Construa um programa em ‘C’ que possua uma função responsável por fazer a troca de valores entre duas variáveis. O programa deve receber dois valores inteiros, através da entrada padrão, armazená-los em variáveis, com a utilização da função mencionada trocar o conteúdo das variáveis e, depois, retorná-las na saída padrão.

```c
#include <stdio.h>
void Swap (int *a,int *b)
{
     int temp;
     temp=*a;
     *a=*b;
     *b=temp;
}
main ()
{
     int num1,num2;
     printf("\nEntre com o primeiro valor: ");
     scanf ("%d",&num1);
     printf("\nEntre com o segundo valor: ");
     scanf ("%d",&num2);
     Swap (&num1,&num2);
     printf ("\n\nEles agora valem %d  %d\n",
     num1,num2);
}
```

Funções

Exercício:

Escreva o código fonte de um programa, na linguagem C, que manipule um vetor de inteiros com dez elementos. O programa deve possuir uma função responsável pela inicialização do vetor e outra função que efetue a impressão do vetor na saída padrão. Por meio das funções mencionadas, o programa deve inicializar o vetor e, em seguida, retorná-lo no monitor. **Obs. o programa não pode possuir variáveis globais.**

UNIVASF