

UNIVERSIDADE ESTADUAL DE MS
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO



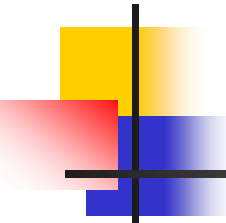
Disciplina: AEDI Modularização

parte 2

Professor: Nilton

E-mail: nilton@comp.uems.br

Rotina

- 
- Sem valor de retorno (procedimento)
 - **void** mensagem(void) {...}
 - Com valor de retorno (função)
 - **int** ehPrimo(void) {...}
 - Sem passagem de parâmetros
 - **int** ehPrimo(**void**) {...} **ou** **int** ehPrimo() {...}
 - Com passagem de parâmetros
 - **float** calculaMedia(**int** A, **int** B, **int** C) {...}

Rotina sem valor de retorno / sem passagem de parâmetros



```
void nomeRotina(void) {  
    <declaração de variável locais>;  
    <comandos>;  
}
```

Rotina sem valor de retorno / sem passagem de parâmetros



- Implemente uma rotina que leia um número e imprima se ele é par ou ímpar.

```
#include <stdlib.h>
#include <stdio.h>

void parImpar(void)
{
    int num;           //declaracao local de variáveis

    printf("\nInforme o numero: ");
    scanf("%d", &num);
    if(num % 2 == 0) {
        printf("numero %d e par!\n", num);
    }
    else {
        printf("numero %d e impar!\n", num);
    }
}

int main(int argc, char *argv[])
{
    parImpar();
    system("PAUSE");
    return 0;
}
```

Rotina com valor de retorno / sem passagem de parâmetros



```
<tipo> nomeRotina(void) {  
    <declaração de variável locais>;  
    <comandos>;  
}
```

<tipo> pode ser int, float, double, char, ...

Rotina com valor de retorno / sem passagem de parâmetros



- Implemente uma rotina que leia um número e devolva 1 se ele é par ou 0 se ele é ímpar.

```
#include <stdlib.h>
#include <stdio.h>

int parImpar(void)
{
    int num;           //declaracao local de variáveis

    printf("\nInforme o numero: ");
    scanf("%d", &num);
    if(num % 2 == 0 ) {
        return 1;
    }
    else {
        return 0;
    }
}

int main(int argc, char *argv[])
{
    int i;             //declaracao local de variáveis do MAIN

    i = parImpar();
    if(i == 1 ) {
        printf("numero e par!\n");
    }
    if( i == 0 ) {
        printf("numero e impar!\n");
    }

    system("PAUSE");
    return 0;
}
```


Rotina com valor de retorno / com passagem de parâmetros



```
<tipo> nomeRotina(<parâmetros>) {  
    <declaração de variável locais>;  
    <comandos>;  
}
```

Rotina com valor de retorno / com passagem de parâmetros



- Implemente uma rotina que receba como parâmetro um número e devolva 1 se ele é par ou 0 se ele é ímpar.

```
#include <stdlib.h>
#include <stdio.h>

int parImpar(int n)
{
    if(n % 2 == 0 ) {
        return 1;
    }
    else {
        return 0;
    }
}

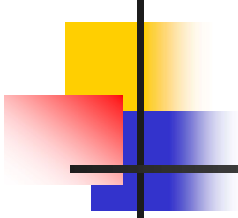
int main(int argc, char *argv[])
{
    int num, i; //declaracao local de variáveis do MAIN

    printf("\nInforme o numero: ");
    scanf ("%d", &num);

    i = parImpar(num);
    if(i == 1 ) {
        printf("numero %d e par!\n", num);
    }
    if( i == 0 ) {
        printf("numero %d e impar!\n", num);
    }

    system("PAUSE");
    return 0;
}
```

Passando vetor como parâmetro

- 
- Implemente uma rotina que encontra o maior número em um conjunto com 50 valores.

```
int Maior_Conj(int v[50]) {  
    int i, maior=v[0];  
  
    for(i=1;i<50;i++)  
        if(v[i]>maior)  
            maior=v[i];  
    return(maior);  
}
```

Passando vetor como parâmetro

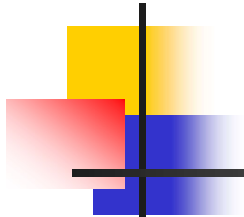
- Implemente uma rotina que encontra o maior número em um conjunto com N valores.

```
int Maior_Conj(int v[], int tamanho) {  
    int i, maior=v[0];  
  
    for(i=1;i<tamanho;i++)  
        if(v[i]>maior)  
            maior=v[i];  
    return(maior);  
}
```

Qual o limite?



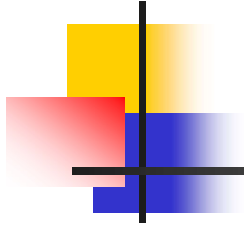
Passando matriz como parâmetro



- Implemente uma rotina que encontra o maior número em uma matriz 30x5.

```
int Maior_Matriz(int m[30][5]) {  
    int i, j, maior=m[0][0];  
  
    for(i=0;i<30;i++) {  
        for(j=0;j<5;j++) {  
            if(m[i][j]>maior)  
                maior=m[i][j];  
        }  
    }  
    return(maior);  
}
```

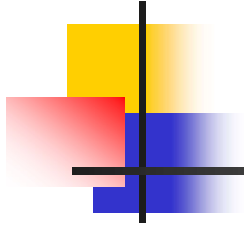
Exemplo 1: criar uma rotina para dividir dois valores inteiros



```
float divisao(int x, int y){  
    float d;  
    if (y==0) d=0;  
    else d = (float)x/y;  
    return d;  
}
```

Chamada: `w = divisao(x,y);`

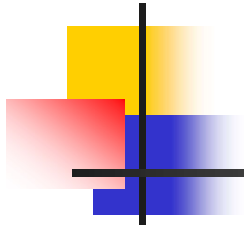
Exemplo 2: criar uma rotina para verificar se X é múltiplo de Y



```
int multiplo (int x, int y){  
    int d;  
    if((x % y) == 0) d = 1;  
    else d = 0;  
    return d;  
}
```

Chamada: d = multiplo1(3,5);

Exercícios – usando módulo em C



1. receba dois valores A e B e realize a troca entre eles.
2. receba dois valores A e B e retorne a soma ou o produto entre eles.
3. receba um conjunto de números e a quantidade de números e retorne quantos números pares existem no conjunto e gere um vetor com esses pares.
4. leia um conjunto de números até que o usuário digite -1. Ao final, todos os valores múltiplos de 5 devem ser guardados em um vetor e a quantidade desses números retornado.
5. receba um conjunto de números, a quantidade de números e um número a ser procurado no conjunto. Retorne 1 se o número procurado existe no conjunto ou 0 se não existe.