Algoritmos e Lógica de Programação

PROF. NILTON

Aula de Hoje

- Programação Modular - Funções

Programação Modular

A Forma geral para definir um módulo é:

```
tipo_retorno nome_do_módulo( lista de parâmetros )
{
        corpo do módulo
        return valor_tipo_retorno
}
```

Programação Modular

Para ilustrar a criação de módulos, faremos um exemplo que calcula o fatorial de um numero inteiro:

```
// Função Principal
void main() {
        int n;
        cout << “Digite um numero”;
        cin >> n;
        fat(n);
}
```

Programação Modular

```
// Função fatorial com passagem de parâmetro
void fat(int n){
        int i;
        int f = 1;
        for ( i = 1; i <= n; i++)
                f *= i;
        cout << “Fatorial: ” + f;
}
```

Neste exemplo o módulo fat não tem nenhum valor de retorno, portanto colocamos a palavra **void** antes do nome do módulo, indicando a ausência de um valor de retorno.

Parâmetros

Parâmetros são os meios pelos quais nós passaremos dados para um módulo, e estes módulos irão trabalhar especificamente sobre essas informações que passamos.

Sempre que um módulo receber dados, estes serão recebidos através dos parâmetros.

Em C ou C++, você precisa deixar bem claro quais os parâmetros e qual o tipo dos dados que você vai passar.

Parâmetros

Vejamos o exemplo anterior:

```
void fat(int n){


}
```

O parâmetro que será passado para o módulo **fat** deverá ser obrigatoriamente do tipo inteiro.

Retornando Valores

Retorno de Valores

Os módulos podem ou não retornar valores. Esta característica pode ser extremamente útil e amplia bastante a possibilidade de utilização das funções. Caso um módulo não retorne nenhum valor, podemos dizer que se trata de um módulo com retorno neutro(nulo).

Em algumas linguagens os módulos que não retornam valores são chamadas de procedimentos e os que retornam valores são chamados de função.

Retornando Valores

Retorno de Valores

O retorno de valores é a saída do módulo. Por meio dele o módulo pode dar uma "resposta" para o módulo ativador.

Essa resposta pode ser um indicador de que todas as operações realizadas dentro do módulo foram realizadas com sucesso, o resultado de uma operação, entre outros.

As respostas devem ser valores do pertencentes aos tipos de dados da linguagem. (int, boolean, String, etc.).

Essa resposta de ser recebida e tratada pelo módulo ativador.

Retornando Valores

Outro exemplo:

```
boolean par(int n){
    if (n%2 == 0)
        return true
    else
        return false;
}
// método ativador
    if (par(n) == true)
            cout << “O Numero é Par”;
```

Obrigado

NILTON.SACCO@FATEC.SP.GOV.BR