	INSTITUTO FEDERAL GOIANO – CÂMPUS URUTAÍ			
	AULA TEÓRICA			
	Curso:	GTI	Turma:	II
	Disciplina:	LTPI	Data:	15/02/21
	Professora:	**Vívian Cirino de Lima**		

FUNÇÃO – MODULARIZAÇÃO

Observe o esquema abaixo:

1. **Estruturas de controle**

1.1. **Estrutura Sequencial –** os comandos são executados na sequência um após o outro, sem interrupção.

1.2. Estruturas Condicionais, de Decisão ou de Seleção – os comandos são executados de acordo com as situações definidas nas condições.

1.2.1. Estrutura Condicional Simples – uma condição (uso do *if*)
1.2.2. Estrutura Condicional Composta – existência de um bloco V (uso do *if)* e um bloco F (uso do *else*).
1.2.3. Estrutura Condicional Encadeada – existência de um agrupamento de condições, enquanto for necessário**.**
1.2.3.1. Estrutura de Seleção de Múltipla Escolha – Composta por uma série de estruturas de seleção simples encadeadas, em que observamos as seguintes prioridades:

- Todas as condições nas decisões são de igualdade.
- Todas as condições comparam uma mesma expressão a uma constante.
- Todas as constantes consideradas são de tipo inteiro ou caractere.

1.3. **Estruturas de Repetição –** Serve para repetir a execução de um bloco de comandos.
1.3.1. Repetição Contada – for
1.3.2. Repetição com precondição (while)
1.3.3. Repetição com poscondição (do/while)

2. **Estruturas de Dados Compostas**
2.1. Homogêneas – Vetor e Matriz
2.2. Heterogênea – Registro (Struct)

3. **Função (Modularização)**

É um bloco de código que visa atingir um objetivo específico. Para criação de uma função em C, temos um padrão definido pela ISO (International Organization for Standardization):

```
Tipo_de_retorno nome (parâmetros){
    variáveis locais;
    comandos;
```

```
}
```
Neste padrão:

- tipo: refere-se ao tipo do valor devolvido como resposta ao final da execução da função (caso não for devolvido nenhum valor, esse tipo deve ser void).
- nome: identifica a função, para que seja referenciada no programa.
- parâmetros: lista de variáveis que representam os dados de entrada, necessários para a execução da função (caso não haja dados de entrada, deverá ser void)
- **variáveis locais**: variáveis acessíveis apenas dentro da função.
- comandos: passos a serem executados pela função.

Ex1: Dentro da função main é chamada diretamente a função multiplica que, por sua vez, realiza a multiplicação entre dois valores e retorno um resultado.

```
#include<stdio.h>
#include<locale.h>
void imprime_cabec (void); //protótipo da função
int multiplica (int n1, int n2); //protótipo da função
int main ( )
{
	int v1, v2, calc;
	setlocale (LC_ALL,"Portuguese");
	imprime_cabec();
	printf("\n\tDigite Valor 1 - ");
	scanf ("%d", &v1);
	printf("\n\tDigite Valor 2 - ");
	scanf ("%d", &v2);
	calc = multiplica (v1, v2);
	printf ("\n\tResultado = %d",calc);
	return (0);
}
void imprime_cabec (void)
{
	printf ("*********************************************************************************\n");
	printf ("*******************************FUNÇÃO EM C*****************************\n");
	printf ("*********************************************************************************\n");
	return;

}
int multiplica (int n1, int n2)
{
	int res;
	res = n1 * n2;
	return (res);
}
```

Tipos de funções:

Ao criarmos uma função, a primeira tarefa é definir seu tipo. Precisamos então definir se, quando chamarmos essa função, o objetivo será dar uma ordem ou fazer uma pergunta. Se for dar uma ordem, a execução da função deve produzir um efeito; caso contrário, deverá dar uma resposta.

Em C, quando a função for apenas produzir um efeito deve ser definida como void; já a função que devolve resposta deve ser definida com o tipo dessa resposta (por exemplo, char, int ou float).

Vamos analisar algumas funções predefinidas em C:

- sqrt (2): uma chamada a esta função corresponde à pergunta “qual a raiz quadrada de 2?”; como a resposta desta pergunta é um número real, ela deve ser definida do tipo float.
- toupper (‘a’): corresponde a responder a pergunta: qual a maiúscula da letra ‘a’? Como a resposta esperada para esta pergunta é um caractere, esta função pode ser criada com o tipo char.

Ex2: Numa disciplina são dadas duas provas e dois trabalhos, mas a média é calculada considerando apenas a maior a nota de prova e a maior nota de trabalho. Dadas as 4 notas de um aluno, informe sua média. Crie e use as funções maior ( ), que determina o maior entre dois números, e media (), que calcula a média aritmética de dois números.

```
#include <stdio.h>
float maior (float a, float b){
    if (a > b)
        return a;
    else
        return b;
}
float media (float a, float b){
    float c;
    c = (a+b)/2;
    return c;
}
int main()
{
    float p1, p2, t1, t2, m;
    printf ("\n\tProva 1? ");
    scanf ("%f", &p1);
    printf ("\n\tProva 2? ");
    scanf ("%f", &p2);
    printf ("\n\tTrabalho 1? ");
    scanf ("%f", &t1);
    printf ("\n\tTrabalho 2? ");
    scanf ("%f", &t2);
    m = media (maior(p1,p2), maior (t1,t2));
    printf ("\n\tMedia = %.1f", m);
    return(0);
}
```