Introdução a Programação

Revisão

O que é função?

Função é um subprograma que consiste de um conjunto de instruções e declarações que executam uma tarefa específica.

Definindo uma função

- Cabeçalho e corpo da função;
- Parâmetros.

Cabeçalho e corpo da função

```
tipo_de_retorno   nome_da_função (tipo param1, tipo param2, ...){
        /* Instruções*/
        ...
}
```

Parâmetros

É um valor fornecido à função quando ela é chamada.

```
int funcao(int a, int b)

float funcao(float preco, int quantidade)
```

Chamadas de funções

- Chamar uma função é transferir o fluxo de execução do programa para a função a fim de executá-la.
- É uma instrução composta pelo nome da função, seguido pela lista de argumentos entre parênteses:

nome_da_função (param1, param2, ...)

- Passagem de parâmetro por valor

Valor de retorno

- Instrução return;
- Encerra imediatamente a execução de uma função;
- **Void** não precisa ter instrução return;
- Sintaxe Geral:

  return expressão;

Alusão

- ➔ Muito semelhante a cabeçalho
- ➔ Não precisa especificar os nomes dos parâmetros
- ➔ Iniciar com a palavra-chave extern

```
extern tipo_de_retorno nome_da_função (tipo
nome_do_param1, tipo nome_do_param2, ...);
```

```c
#include <stdio.h>

/*   Alusão          */
extern int multiplica(int num1 , int num2);

int main(void){
     int entr1 = 0, entr2 = 0, mul = 0;

     printf("Digite o primeiro numero a ser multiplicado:\n");
     scanf("%d", &entr1);

     printf("Digite o segundo numero a ser multiplicado:\n");
     scanf("%d", &entr2);

     mul = multiplica(entr1, entr2); /* Chamada da função */

     printf("%d x %d = %d\n", entr1, entr2, mul);

     return 0;
}

/* Definição de chamada  */
int multiplica(int num1, int num2){
     return (num1 * num2);
}
```

Aula 6

Duração e escopo de variáveis
Qualificadores de tipos

Duração Automática

- ➔ Tem espaço em memória alocado para si apenas quando o escopo dela é executado;
- ➔ Quando encerra a execução do seu escopo, a variável deixa de existir e o espaço em memória é liberado.

Ex.:

```
void main(){
int x;
}
```

Duração Fixa

- ➔ Uma variável de duração fixa é uma variável cujo período de vida é todo o tempo de execução do programa;
- ➔ É iniciada apenas uma vez;

```
Ex.: void funcaoExemplo(){
static int x;
 }
```

Exemplo 1

```
void incrementa() {

int i = 1;

static int j = 1;

i++; j++;

printf("i: %d, j: %d", i, j);

}
```

Exemplo 2

```
int i = 1;

int j = 2*i + 7;     // Legal, j é de DA e i já é conhecida

static int k = i;    /* ILEGAL, porque k é de DF, logo sua iniciação não
                        pode envolver variáveis */
```

Resumo Geral

Variável de Duração Automática	Variável de Duração Fixa
NUNCA tem iniciação implícita	Iniciada implicitamente com ZERO
Iniciação PODE envolver variáveis	Iniciação NÃO PODE envolver variáveis
Pode ser iniciada VÁRIAS vezes	Iniciada uma ÚNICA vez

Escopo

Classificados em :

- ➔ Escopo de Programa
- ➔ Escopo de Arquivo
- ➔ Escopo de Função
- ➔ Escopo de Bloco

Escopo de Programa

- ➔ Variáveis e Funções;
- ➔ Ativo em todos os arquivos e blocos que compõem o programa;
- ➔ Variáveis Globais e Funções Globais;

Obs.: Não precedidas por static

Escopo de Programa

```
#include <stdio.h>
...
 float umFloat;
...
 int minhaFuncao(){
 ...
 return 0;
 }
```

Escopo de Arquivo

- ➔ Válido em todos os blocos do arquivo no qual ela é declarada e a partir do ponto de declaração do identificador;
- ➔ Variáveis definidas fora de funções com static
- ➔ Funções definidas com static;
- ➔ Identificadores associados a tipos de dados definidos pelo programador
- ➔ Identificadores associados a macros

Escopo de Arquivo

CUIDADO COM O STATIC!

Nesse caso, não se trata da duração de variáveis, mas sim da definição de escopo.

Escopo de Arquivo

```
#include <stdio.h>
...
 static float umFloat;
 ...
static void MinhaFuncao(){
 ...
 }
int main(){
...
return 0;
}
```

Escopo de Função

- ➔ Identificador válido do início ao final da função na qual ele é declarado.
- ➔ Apenas rótulos, utilizados em conjunto com instruções goto
- ➔ Devem ser únicos dentro de uma função;
- ➔ Permanecem ativos do início ao final da mesma;

Escopo de Bloco

- ➔ Identificador tem validade a partir do seu ponto de declaração até o final do bloco na qual ele é declarado.
- ➔ Parâmetros e variáveis definidas dentro do corpo de uma função

Obs.: por conta dessa última, não pode existir uma variável com o mesmo nome de um parâmetro

Exercício!

```
int i;
static int j;

void f(int k){
int m;
static short n;
…

 }
```

Resposta

```
int i;      //EP e DA
static int j; //EA e DF

void f(int k){ //f tem EP e
k de bloco
      int m; //EB e DA
      static short n; //EB e
      DF
      …
}
```