INTRODUÇÃO À PROGRAMAÇÃO

Prof.ª Priscilla Abreu

priscilla.braz@rj.senac.br

Introdução à Programação

Roteiro de Aula

- Objetivo da aula
- Funções e procedimentos

Introdução à Programação

Objetivo da aula

Implementar técnicas de programação modular utilizando a linguagem C.

BIBLIOTECAS DA LINGUAGEM C (REVISÃO)

Introdução à Programação

BIBLIOTECAS BÁSICAS

Cabeçalhos	Descrição
ctype.h	Utilizado para testar e converter caracteres
errno.h	Utilizado na identificação e no tratamento de erros
locale.h	Define informações sobre localização (país, língua, mensagens, moeda, etc)
math.h	Define várias funções matemáticas
stdio.h	Permite realizar operações de entrada/saída
stdlib.h	Define várias funções de propósito geral como gerenciamento de memória dinâmica, geração de número aleatório, comunicação com o ambiente, aritmética de inteiros, busca, ordenação e conversão
string.h	Define funções para manipulação de strings
time.h	Define funções para ler e converter datas e horas

Introdução à Programação

BIBLIOTECA <locale.h>

Função que configura caracteres especiais e acentuação.

```
#include <locale.h>
int main(){
        setlocale(LC_ALL,"");
        printf("Já posso usar acentuação!");
}
```

Introdução à Programação

BIBLIOTECA MATEMÁTICA <math.h>

```
#include <stdio.h>
#include <math.h>
int main(){
        int n1,n2,res;
        printf("Informe a base: ");
        scanf("%d",&n1);
        printf("Informe o expoente: ");
        scanf("%d",&n2);
        res=pow(n1,n2);
        printf("%d elevado a %d é: %d",n1, n2,res);
}
```

PROGRAMAÇÃO MODULAR
FUNÇÕES E PROCEDIMENTOS

Análise e Desenvolvimento de Sistemas
2021.2

Introdução à Programação

PROGRAMAÇÃO MODULAR

- Uma característica fundamental da Programação Estruturada é decompor a lógica de programas complexos em programas menores e, depois, juntá-los para compor o programa final.
- Essa técnica de programação é denominada programação modular.
- Facilita a construção de grandes programas, através de sua divisão em pequenas etapas, que são os módulos ou sub-rotinas.

Introdução à Programação

PROGRAMAÇÃO MODULAR

- Possibilita o reaproveitamento de código;
- Permite que diferentes programadores trabalhem simultaneamente na solução de um mesmo problema.
- Pode ser feita através de procedimentos e funções.
- Associa-se um nome a uma sequência de comandos. Para que o bloco de comandos seja executado, informamos o nome do procedimento ou função com seus respectivos parâmetros.

PROGRAMAÇÃO MODULAR – EXEMPLO

Exemplo: O programa abaixo calcula a média aritmética entre 2 notas, **sem o uso de programação modular**:

```
#include <stdio.h>
int main(){
        float NOTA1,NOTA2,MEDIA;
        printf("Digite a primeira nota: ");
        scanf("%f",&NOTA1);
        printf("Digite a segunda nota: ");
        scanf("%f",&NOTA2);
        MEDIA = (NOTA1 + NOTA2) / 2;
        printf("Media = %f",MEDIA);
}
```

Introdução à Programação

PROGRAMAÇÃO MODULAR – EXEMPLO

- **Exemplo:** Utilização de programação modular:

```
#include <stdio.h>
float NOTA1,NOTA2;
void leitura(){
        printf("Digite a primeira nota: ");
        scanf("%f",&NOTA1);
        printf("Digite a segunda nota: ");
        scanf("%f",&NOTA2);
}
```

Variáveis acessíveis por todo o programa

Introdução à Programação

PROGRAMAÇÃO MODULAR – EXEMPLO

```
float media(){
        return ((NOTA1 + NOTA2) / 2);
}
int main(){
        leitura();
        printf("Media = %f", media());
}
```

Interrompe o ponto em que estava no main e vai até o procedimento leitura para executá-lo.

PROGRAMAÇÃO MODULAR

Toda sub-rotina (procedimento ou função) mantém as mesmas características de um algoritmo comum:

- Pode ter dados de entrada;
- Dados de saída; e
- Conter qualquer tipo de comando aceito por um algoritmo.

Por convenção, uma sub-rotina deve ser declarada acima dos módulos que o chamam;

FUNÇÕES PROCEDIMENTOS

FUNÇÕES

- Blocos de programas que retornam um valor.
- Na linguagem C não há o conceito de um programa principal. O que existe é uma função chamada *main* que é sempre a primeira a ser executada.
- Um programa pode ser escrito apenas com a função *main* e mais as funções existentes nas bibliotecas da linguagem C. No entanto, o uso de funções pode facilitar o desenvolvimento de programas de diversas maneiras.

Introdução à Programação

FUNÇÕES – BIBLIOTECA C

```
#include <stdio.h>
#include <math.h>
int main(){
        float n1, res;
        printf("Informe um número: ");
        scanf("%f",&n1);
        res=sqrt(n1);
        printf("Raiz de %.1f = %.1f",n1,res);
}
```

Introdução à Programação

FUNÇÕES – EXEMPLO

```
#include <stdio.h>
int n1;
int Square (  ) {
        return ( n1*n1 ) ;
}
int main ( ) {
        int n2 ;
        printf ( “Entre com um numero : ” ) ;
        scanf ( “%d” , &n1 ) ;
        n2 = Square( ) ;
        printf (“O seu quadrado vale : %d\n”, n2) ;
}
```

Chama a função Square

return => n2

Introdução à Programação

FUNÇÕES – FORMATO GERAL

```
tipo NomeDaFuncao (Lista_de_Parametros){
    corpo da função
}
```

onde:

- tipo especifica o tipo de valor que o comando return da função devolverá.
- Lista_de_Parâmetros: lista de nome de variáveis com seus respectivos tipos de dados, que recebem os valores dos argumentos quando a função é chamada.

FUNÇÕES

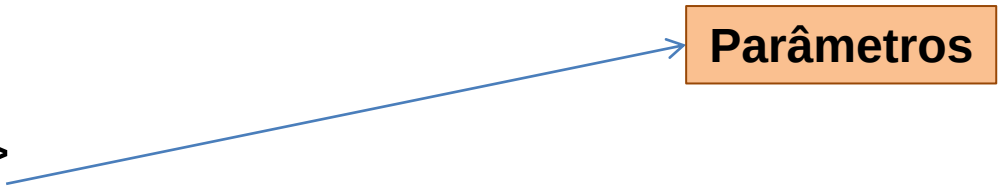
Para se criar uma função é necessário:

- Tipo de retorno/resultado da função;
- Um identificador (o nome da função);
- Uma lista de parâmetros (que possibilitam a comunicação entre a função principal e a função que está sendo criada);
- As ações a serem executadas (que formam o corpo da função).

As funções devem ser definidas antes da função main.

Introdução à Programação

FUNÇÕES – EXEMPLO




```
#include <stdio.h>
int soma (int n1, int n2) {
        return ( n1 + n2 ) ;
}
int main ( ) {
        int n1,n2;
        printf ( “Entre com dois números: ” ) ;
        scanf ( “%d” , &n1 ) ;
        scanf ( “%d” , &n2 ) ;
        printf (“A soma de %d e  %d é : %d\n”, n1, n2,
soma(n1,n2)) ;
}
```

FUNÇÕES – Retorno

Há basicamente duas maneiras de terminar a execução de uma função.

- Normalmente usa-se o comando *return* para retornar o resultado da função para o local onde a função for chamada.
- Quando não há valor para retornar o comando return não precisa ser usado e a função termina quando a chave que indica o término do corpo da função é atingido. Nesse caso, chamamos de procedimento.

PROCEDIMENTO

Em inglês, void quer dizer vazio e é isto mesmo que o void é. Ele nos permite fazer funções que não retornam nada, o que chamamos de **procedimento**.

Podemos agora escrever o protótipo de uma função que não retorna nada:

void nome_da_função (declaração_de_parâmetros);

Neste caso, o comando *return* não é necessário na função.

Introdução à Programação

PROCEDIMENTO – EXEMPLO

```
#include <stdio.h>
float NOTA1,NOTA2;
void leitura(){
        printf("Digite a primeira nota: ");
        scanf("%f",&NOTA1);
        printf("Digite a segunda nota: ");
        scanf("%f",&NOTA2);
}
int main(){
        float MEDIA;
        leitura();
        MEDIA = (NOTA1 + NOTA2) / 2;
        printf("Media = %f",MEDIA);
}
```

ESCOPO DE VARIÁVEIS

As variáveis podem ser declaradas basicamente em três lugares:

- dentro de funções,
- fora de todas as funções,
- na lista de parâmetros das funções.

As variáveis definidas dentro das funções são chamadas de **variáveis locais**, as que aparecem fora de todas as funções chamamos de **variáveis globais** e aquelas que aparecem na lista de parâmetros são os **parâmetros formais**.

ESCOPO DE VARIÁVEIS

Variáveis Locais - Exemplo

```
#include <stdio.h>
void pares(){
        int i;
        for (i = 2; i <= 10; i += 2){
                printf ("%d: ", i);
        }
}
int main (){
        pares ();
}
```

Variável local: existe apenas no escopo da função e enquanto ela estiver ativa.

ESCOPO DE VARIÁVEIS

Variáveis Globais – Exemplo

```
#include <stdio.h>
int i;
void calc(){
        i += 1;
        printf(" Procedimento calc: i = %d\n", i);
}
int main (){
        i = 0;
        calc();
        printf("Funcao main : i = %d\n", i);
}
```

ESCOPO DE VARIÁVEIS

Parâmetros formais – Exemplo

```
#include <stdio.h>
float sqr (float n) {
        n=n*n;
        return n;
}
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);
}
```

Parâmetro: variável que receberá informação do tipo ***int*** quando for chamada.

PARÂMETROS

Os parâmetros ou argumentos são usados para comunicação entre as sub-rotinas (procedimentos e funções), incluindo a função main.

Já usamos essa comunicação com os procedimentos/comandos pré-definidos da linguagem, como por exemplo printf. Quando temos um trecho de programa como a seguir:

```
printf(“Teste”);
```

Introdução à Programação

PARÂMETROS

E também nas funções predefinidas das bibliotecas da linguagem C.

```
#include <stdio.h>
#include <Math.h>
int main(){
        float n1, res;
        printf("Informe um número: ");
        scanf("%f",&n1);
        res=sqrt(n1);
        printf("Raiz de %.1f = %.1f",n1,res);
}
```

Introdução à Programação

PARÂMETROS – EXEMPLO

```
#include <stdio.h>
float AreaTriangulo(int b, int a) {
        return ( a*b/2 ) ;
}
int main ( ) {
        int base, alt ;  float res;
        printf ( “Informe a base do triângulo: ” ) ;
        scanf ( “%d” , &base) ;
        printf ( “Informe a altura do triângulo: ” ) ;
        scanf ( “%d” , &alt) ;
        res = AreaTriangulo(base, alt) ;
        printf (“A área do triângulo é: %f\n”, res) ;
}
```

EXERCÍCIO

Faça um programa que leia dois números inteiros e imprima o produto desses números. Seu programa deve calcular esse produto utilizando uma função e o resultado deve ser impresso na função main.

PROTÓTIPO DE FUNÇÕES

O protótipo de uma função é uma declaração de função que omite o corpo, mas especifica o seu nome, tipo de retorno e lista de parâmetros.

A forma geral de definição de um protótipo é a seguinte:

tipo nome (tipo nome1, tipo nome2, ..., tipo nomeN);

Introdução à Programação

PROTÓTIPO DE FUNÇÕES – EXEMPLO

```
#include < stdio.h>
/* Prototipo da funcao */
int soma ( int a, int b);
int main () {
        int a=5, b=9;
        printf ("%d\n", soma (a,b));
        return 0;
}
/* Definicao da funcao */
int soma ( int a, int b) {
        return a+b;
}
```

Introdução à Programação

TAREFA – ENTREGA ATÉ 23/10

Com o objetivo de simular uma calculadora simplificada, faça um programa que implemente uma função para cada operação básica dessa calculadora, considerando que para cada cálculo sejam passados por parâmetro dois valores inteiros. Na função main, o programa deve divulgar as opções para os usuários, solicitar que ele escolha a operação desejada e executar e exibir o resultado do cálculo.

Introdução à Programação

Considerações

A programação modular é uma importante técnica de Programação Estruturada que nos permite organizar nosso código em módulos ou trechos, que podem ser reutilizados.