Departamento
de Ciência da
COMPUTAÇÃO

UDESC
Joinville

Linguagem C
Funções

André Tavares da Silva

andre.silva@udesc.br

Funções

- É uma coleção de comandos identificados por um nome (identificador);
- As funções executam ações e podem retornar valores;
- *main*, *printf* e *scanf* são exemplos de funções.

```
<tipo retornado> <nome>(<parâmetro 1>,...,<
   parâmetro n >)
{
   <declaração das variáveis locais>
   <comandos>
   [return <valor de retorno>;]
}
```

Funções

- É uma coleção de comandos identificados por um nome (identificador);
- As funções executam ações e podem retornar valores;
- *main*, *printf* e *scanf* são exemplos de funções.

```
int soma(int x, int y) {
   int resultado;
   resultado = x + y;
   return resultado;
}

void ImprimirValor (float v) {
   printf("%.2f", v);
}
```

Funções

- Agrupa um conjunto de comandos e associa a ele um nome.
- Após sua execução, o programa volta ao ponto imediatamente após a chamada.
- A volta ao programa que chamou a função é chamada de retorno.
- O programa sempre inicia sua execução pela função **main**.

Funções

- **Tipo retornado**: pode ser qualquer um dos tipos que já vimos (*int, short int, long int, char*, ...) ou então *void*, que indica que a função não retornará nenhuma informação.
- **Nome**: é o identificador da função, servindo para referenciar a mesma.
- **Parâmetros**: são as informações para trabalhar. Constituem-se de variáveis locais à função, que são declarados individualmente, separados por vírgula.

Passagem de Parâmetros

- **Por valor**: São fornecidas cópias dos valores dos parâmetros na expressão que chama a função. A função pode alterar o valor dos parâmetros, mas esta mudança não é refletida às variáveis originais.

```
void ImprimirValor (float v) {
   printf("%.2f", v);
}
```

- **Por referência**: Este tipo permite que se altere o conteúdo das variáveis passadas como parâmetro. A função recebe um endereço de memória como parâmetro. É exatamente o que acontece quando utilizamos a função scanf. É informado que os dados lidos deverão ser armazenados em variáveis passadas por parâmetro.

```
scanf("%f", &valor); /* note o & */
```

Funções

- No seu retorno, uma função pode retornar resultados ao programa que a chamou.
- Na função de exemplo soma, o valor da variável resultados é passado de volta como o valor da função.

```
return resultados;
```

- As funções que não retornam valores são também chamadas de procedimentos.

Funções

- Se um comando **`return`** é executado, o controle é passado de volta para o trecho que chamou a função. O valor então é retornado para o trecho que chamou a função.
- Se um comando **`return`** inclui uma expressão, o valor da expressão é convertido, se necessário, pelo tipo do valor que a função retorna.
- Se um comando **`return`** não inclui uma expressão, nenhum valor é retornado ao trecho que chamou a função.

Escopo das Variáveis

- O escopo de uma variável é definido pelas blocos (definidos pelas chaves) onde a variável pode ser utilizada.
- Por exemplo, as variáveis declaradas no início do corpo da função *main* podem ser utilizadas em qualquer lugar dentro da função *main*, porém apenas DENTRO dela, ou seja, não podem ser utilizadas em uma outra função.
- Variáveis declaradas no mesmo escopo precisam ter nomes diferentes, mas podem ter nomes iguais se forem de diferentes escopos. Neste caso, somente pode ser acessada a variável do bloco mais interno.

Variáveis Locais

- Variáveis declaradas dentro de uma função.
- Não são reconhecidas fora de seu próprio bloco de código.
- Existem apenas enquanto o bloco de código em que foram declaradas está sendo executado, ou seja, é criada na entrada de seu bloco e destruída na saída.

Variáveis Globais

- São reconhecidas pelo programa inteiro e podem ser usadas por qualquer pedaço de código.
- Guardam seus valores durante toda a execução.
- Devem ser declaradas fora de qualquer função.
- Ocupam memória durante todo o tempo de execução do programa.
- Tornam as funções menos gerais.
- Podem levar a erros no programa devido a efeitos colaterais.

Protótipo de Funções

- É a repetição da declaração da função, porém não necessita dos nomes das variáveis que recebem os parâmetros, encerrando com ponto e vírgula (;).
- Isto ajuda o compilador a verificar se os parâmetros que estão sendo passados nas funções estão de acordo com os tipos que elas esperam receber.
- É feita extraindo-se:
  - Tipo retornado pela função
  - Nome da função;
  - Tipo dos parâmetros entre parênteses

```
void ImprimirValor(float);
```

Recapitulando

```
#include <stdio.h>                /* inclusão de biblioteca - entrada e saída */

const float VALORHORA=10.0;       /* variável global e constante */

int main() {                                  /* função principal */
     int horas = LeHoras();
     printf("A receber: %.2f\n", CalculaSalario(horas) );
     return 0;
}

float CalculaSalario(int h) {  return (float)h*VALORHORA;} /*calcula salário*/

int LeHoras() {      /* função para leitura de horas */
     int horas;         /* variável local */
     printf("Entre numero de horas trabalhadas: ");
     scanf("%d", &horas );
     return horas;
}
```

Recapitulando

```
#include <stdio.h>            /* inclusão de biblioteca - entrada e saída */
int LeHoras();                /* protótipo */
float CalculaSalario(int);    /* protótipo */

const float VALORHORA=10.0;       /* variável global e constante */

int main() {                      /* função principal */
      int horas = LeHoras();
      printf("A receber: %.2f\n", CalculaSalario(horas) );
      return 0;
}

float CalculaSalario(int h) {  return (float)h*VALORHORA;} /*calcula salário*/

int LeHoras() {      /* função para leitura de horas */
      int horas;        /* variável local */
      printf("Entre numero de horas trabalhadas: ");
      scanf("%d", &horas );
      return horas;
}
```

Exercícios

- Escreva um programa em C que executa conversões de temperatura entre os sistemas CELSIUS e FAHRENHEIT. Caso o usuário deseje converter de CELSIUS para FAHRENHEIT ele pressiona a tecla C; pressionando a tecla F o programa faz a conversão de FAHRENHEIT para CELSIUS. **F = C * (9/5) + 32**

- Altere o programa anterior incluindo duas funções: uma para converter de graus Celsius para graus Fahrenheit (`float CelsiusParaFahr(float)`) e outra para converter de graus Fahrenheit para Celsius (`float FahrParaCelsius(float)`).