Introdução à Linguagem C

Tradução

- MONTADOR (*assembler*)
  - Tradutor para linguagens de 2ª geração.
- COMPILADOR:
  - Traduz todo o programa de uma vez.
- INTERPRETADOR:
  - Traduz o programa instrução por instrução.

Histórico da Linguagem C

- Criada por Denis Ritchie, na década de 1970, para uso em um computador DEC PDP-11 em Unix
- BCPL $\rightarrow$ B $\rightarrow$ C $\rightarrow$ C++
- C++ é uma extensão da linguagem C
- O sistema Unix é escrito em C e C++

Estrutura básica de um programa C

```
diretivas para o pré-processador
declaração de variáveis globais
main ()
{
   declaração de variáveis locais da
   função main
   comandos da função main
}
```

Diretivas para o processador - Bibliotecas

- Diretiva `#include` permite incluir uma biblioteca
- Bibliotecas contêm funções pré-definidas, utilizadas nos programas
- Exemplos

`#include <stdio.h>`	Funções de entrada e saída
`#include <stdlib.h>`	Funções padrão
`#include <math.h>`	Funções matemáticas
`#include <system.h>`	Funções do sistema
`#include <string.h>`	Funções de texto

Dicas

- Termine todas as linhas com **;**
- Sempre salve o programa antes de compilar
- Sempre compile o programa antes de executar
- Verifique também a linha anterior, que pode ser a responsável pelo erro, especialmente se faltar o **;**
- Use comentários (/* .. */ ou //)

Template

```
#include <stdio.h>
main()
{
   printf ("Alo mundo!");
   system("PAUSE");
}
```

Variáveis

- Declaram as variáveis e seus tipos
- Os nomes das variáveis devem conter apenas letras, dígitos e o símbolo _
- Até 32 caracteres
- Os principais tipos são: **int**, **float** e **char**
- Exemplos

```
int n;
int quantidade_valores;
float x, y, somaValores;
char sexo;
char nome[40];
```

CASE SENSITIVE
int n, N;
n é diferente de N!

Constantes

#DEFINE

Texto sendo substituído por outro sem nenhum tipo de verificação.

SINTAXE

#define identifier value

Exemplos:

#define N 5

#define max(A, B) ((A > B) ? (A) : (B))

CONST

Uma “variável” que não pode ter o valor alterado. O compilador reserva espaço na memória para ela.

SINTAXE

const tipo variável = valor;

Exemplos:

const int altura = 10;

const float largura = 5.25;

Real: n1, n2, n3, media

```
#include <stdio.h>
main()
{
    float n1, n2, n3, media;




system("PAUSE");
}
```

Comando de atribuição

- Atribui o valor da direita à variável da esquerda
- O valor pode ser uma constante, uma variável ou uma expressão
- Exemplos

```
x = 4;  --> lemos x recebe 4
y = x + 2;
y = y + 4;
valor = 2.5;
sexo = 'F';
```

Entrada e Saída

- Função **scanf**

```
scanf ("formatos", &var1, &var2,...)
```

Exemplos:

```
int i, j;
float x;
char c;
char* nome;
scanf("%d", &i);
scanf("%i %f", &j, &x);
scanf("%c", &c);
scanf("%s", nome);
```

%d ou %i	inteiro
%f	float
%c	char
%s	string

Real: n1, n2, n3, media

ler n1, n2, n3

```
#include <stdio.h>
main()
{
    float n1, n2, n3, media;
   scanf ("%f %f %f",&n1, &n2, &n3);



system("PAUSE");
}
```

Operadores Matemáticos

Operador	Exemplo	Comentário
+	`x + y`	Soma x e y
-	`x - y`	Subtrai y de x
*	`x * y`	Multiplica x e y
/	`x / y`	Divide x por y
%	`x % y`	Resto da divisão de x por y
++	`x++`	Incrementa em 1 o valor de x
--	`x--`	Decrementa em 1 o valor de x

Real: n1, n2, n3, media

ler n1, n2, n3

media=(n1+n2+n3)/3

```
#include <stdio.h>
main()
{
    float n1, n2, n3, media;
   scanf ("%f %f %f",&n1, &n2, &n3);
   media=(n1+n2+n3)/3;

system("PAUSE");
}
```

Entrada e Saída

- Função **printf**

```
printf ("formatos", var1, var2,...)
```

Exemplos:

```
int i, j;
float x;
char c;
char* nome;
printf("%d", i);
printf("%i, %f", j, x);
printf("%c", c);
printf("%s", nome);
```

%d ou %i	inteiro
%f	float
%c	char
%s	string

Real: n1, n2, n3, media

ler n1, n2, n3

media=(n1+n2+n3)/3

exibir media

```
#include <stdio.h>
main()
{
    float n1, n2, n3, media;
    scanf ("%f %f %f",&n1, &n2, &n3);
    media=(n1+n2+n3)/3;
    printf (“%f”,media);

    system("PAUSE");
}
```

Mãos à Obra!

```
#include <stdio.h>
main()
{
    float n1, n2, n3,
    scanf ("%f %f %f",
          &n1, &n2, &n3);

    media=(n1+n2+n3)/3;
    printf (“%f”,media);


    system("PAUSE");
}
```

```
#include <stdio.h>
main()
{
    float n1, n2, n3, media;

    printf(“Digite 3 notas: ”);
    scanf ("%f %f %f",&n1, &n2, &n3);
    media=(n1+n2+n3)/3;
    printf (“A média é %0.2f”,media);


    system("PAUSE");
}
```

Exercicio

1) Escreva um programa em C que leia o valor da base e altura de um triângulo, calcule e escreva sua área.

área=(base * altura) / 2

Operadores de Atribuição

Operador	Exemplo	Comentário
=	x = y	Atribui o valor de y a x
+=	x += y	Equivale a x = x + y
-=	x -= y	Equivale a x = x – y
*=	x *= y	Equivale a x = x * y
/=	x /= y	Equivale a x = x / y
%=	x %= y	Equivale a x = x % y

Funções Matemáticas

Função	Exemplo	Comentário
`ceil`	`ceil(x)`	Arredonda o número real para cima; ceil(3.2) é 4
`cos`	`cos(x)`	Cosseno de x (x em radianos)
`exp`	`exp(x)`	**e** elevado à potencia x
`fabs`	`fabs(x)`	Valor absoluto de x
`floor`	`floor(x)`	Arredonda o número deal para baixo; floor(3.2) é 3
`log`	`log(x)`	Logaritmo natural de x
`log10`	`log10(x)`	Logaritmo decimal de x
`pow`	`pow(x, y)`	Calcula x elevado à potência y
`sin`	`sin(x)`	Seno de x
`sqrt`	`sqrt(x)`	Raiz quadrada de x
`tan`	`tan(x)`	Tangente de x

```
#include <math.h>
```

Exercício

2) Construa um algoritmo que tendo como entrada dois pontos quaisquer do plano P(x1,y1) e Q(x2,y2), imprima a distância entre eles.

A fórmula da distância é: $\sqrt{(x2-x1)^2 + (y2-y1)^2}$

3) Construa um algoritmo que calcule a quantidade de latas de tinta necessárias e o custo para pintar tanques cilíndricos de combustível, onde são fornecidos a altura e o raio desse cilindro.

Sabendo que:

- a lata de tinta custa R$20,00
- cada lata contém 5 litros
- cada litro de tinta pinta 3 metros quadrados.

Sabendo que:

Área do cilindro=3,14*raio^2 + 2*3,14*raio*altura

e que raio e altura são dados de entrada.

Operadores Relacionais

Operador	Exemplo	Comentário
==	`x == y`	O conteúdo de x é igual ao de y
!=	`x != y`	O conteúdo de x é diferente do de y
<=	`x <= y`	O conteúdo de x é menor ou igual ao de y
>=	`x >= y`	O conteúdo de x é maior ou igual ao de y
<	`x < y`	O conteúdo de x é menor que o de y
>	`x > y`	O conteúdo de x é maior que o de y

Operadores Lógicos

- **&&** (E lógico): retorna verdadeiro se ambos os operandos são verdadeiros e falso nos demais casos.
  Exemplo: if(a>2 && b<3).

- **||** (OU lógico): retorna verdadeiro se um ou ambos os operandos são verdadeiros e falso se ambos são falsos.
  Exemplo: if(a>1 || b<2).

- **!** (NÃO lógico): usada com apenas um operando. Retorna verdadeiro se o operando é falso e vice-versa.
  Exemplo: if(!var).

Operadores Lógicos

Tabela E	Tabela OU	Tabela NÃO
V e V → V	V ou V → V	Não V → F
V e F → F	V ou F → V	Não F → V
F e V → F	F ou V → V	
F e F → F	F ou F → F	

Estrutura condicional simples

- Comando **if**

```
if (condição)
    comando;

if (condição) {
    comando1;
    comando2;
    comando3;
}
```

```
if (a<menor)
   menor=a;


if (a<menor) {
    menor=a;
    printf (“%d”, menor);
}
```

Estrutura condicional composta

- Comando **if...else**

```
if (condição)
    comando;
else
    comando;
```

Executa o comando se a condição for qualquer coisa diferente de zero!

```
if (condição) {
    comando1;
    comando2;
} else {
    comando3;
    comando4;
}
```

```
if (peso == peso_ideal)
    printf ("Vc está em forma!");
else
    printf ("Necessário fazer dieta!");
```

Estrutura condicional SWITCH

```
switch (variável) {
        case constante1: {
                comandos;
                break;
        }
        case constante2: {
                comandos;
                break;
        }
        default: {
                comandos;
        }
}
```

Exercício

4) Dados os 3 valores A, B, C, verificar se eles podem ser os comprimentos dos lados de um triângulo e, se forem, verificar se compõem um triângulo equilátero, isósceles ou escaleno.

Informar se não compuseram nenhum triângulo.

- Triângulo: figura geométrica de 3 lados, onde cada um é menor do que a soma dos outros dois.
- Triângulo eqüilátero: Triângulo com 3 lados iguais.
- Triângulo isósceles: Triângulo com 2 lados iguais.
- Triângulo escaleno: Triângulo com todos os lados diferentes.

OBS.: Não existe lado com tamanho 0 (zero).

Exercício

5) Faça um programa que leia um número inteiro e mostre uma mensagem indicando se este número é par ou ímpar, e se é positivo ou negativo.

6) Explique porque está errado fazer if (num=10) ... O que irá acontecer?

Estrutura de repetição

- Comando **for**

```
for (var=valor inicial; condição; incremento)
    comando;

for (var=valor inicial; condição; incremento)
{
    comando1;
    comando2
    comando3;
}
```

Exemplo:

```
for (cont=3; cont<=11; cont++)
    printf ("%d",cont);
```

Exercícios

7)Sendo h= 1 + ½ + 1/3 + ¼ + ... + 1/N, prepare um algoritmo para calcular o número h, sendo o número N fornecido pelo usuário.

8) Elabore um algoritmo que calcule N! (fatorial de N), sendo que o valor inteiro de N é fornecido pelo usuário. Sabendo que:

- N! = 1 x 2 x 3 x ... x (N-1) x N;
- 0! = 1, por definição.

9)Faca um programa que apresente na tela a tabela de conversão de graus Celsius para Fahrenheit, de -100 C a 100 C. Use um incremento de 10 C.

- OBS: Farenheit = (9/5)*(Celsius) + 32

Exercícios

10)Escreva um programa que coloque os números de 1 a 100 na tela na ordem inversa (começando em 100 e terminando em 1).

11) Calcular e listar todos os múltiplos positivos do número 7 menores ou iguais a 100.

12) Calcular $$\sum_{i=0}^{i=7}\sum_{j=0}^{j=7}\frac{(2*j+1)*i}{2*j+5}$$

Pergunta: quantas vezes executa a fórmula?

Estrutura de repetição

- Comando **while**

```
while (condição)
    comando;

while (condição) {
    comando1;
    comando2
    comando3;
}
```

Exemplo:

```
while (N != 0) {
   scanf ("%d",&N);
   if (N > MAIOR) MAIOR = N;
 }
```

Exercício

12)Fulano tem 1,50 metro e cresce 2 centímetros por ano, enquanto Ciclano tem 1,10 e cresce tem 3 centímetros por ano. Construa um programa que calcule e imprima quantos anos serão necessários para que Ciclano seja maior que Fulano.

Estrutura de repetição

- Comando **do...while**

```
do {
    comando
} while (condição);

do {
    comando1;
    comando2
    comando3;
} while (condição);
```

Exemplo:

```
cont=0;
do {
    cont = cont + 1;
    printf("%d\n",cont);
} while (cont < 10);
```

Exercício

13) Escreva um programa que peça ao usuário que digite três números inteiros, correspondentes a dia , mês e ano. Teste os números recebidos, e em caso de haver algum inválido, repita a leitura até conseguir valores que estejam na faixa correta (dias entre 1 e 31, mês entre 1 e 12 e ano entre 1900 e 2100). Verifique se o mês e o número de dias batem. Se estiver tudo certo imprima o número que aquele dia corresponde no ano. Comente seu programa.

Exercício

14) Faça um programa que leia números digitados pelo usuário até ele digitar -1 (utilizando um laço while). No final, calcule a média dos números fornecidos.

15) Perguntar ao usuário quantos números deseja somar. Em seguida, ler estes N números e apresentar o valor da soma. (Fazer 3 versões deste programa: usando FOR, usando WHILE e usando DO...WHILE).

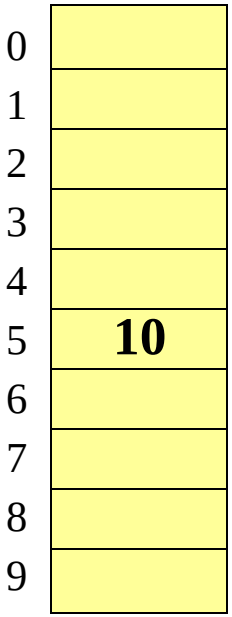
Vetores (array)

- Trata-se de automatizar a declaração de um grande número de dados de um mesmo tipo simples. As variáveis assim declaradas se acessam através de um índice de tipo int.

- Declaração:
  - int v[100];
  - primeira posição =0;
  - última posição=99;
- Atribuição:
  - v [9] = 87;

- Acessar um valor:
  - a = v[9];

- int v[10];



V[5]=10;

printf (“%d”,V[5]);

Quando usar?

- Calcular a média da nota de 5 alunos e verificar quantos conseguiram nota acima da média:

```
inteiro: cont, soma;
real: media, num
para cont=1 até 5 repetir
        receber num
        soma = soma + num
media = soma/cont
```

Como verificar se a nota de cada aluno é maior que a média??

```
#include <stdio.h>
main()
{
 int contA=0;
 double media, n1, n2, n3, n4, n5;
 scanf ("%lf %lf %lf %lf %lf",
            &n1, &n2, &n3, &n4, &n5);
 media = (n1+ n2 + n3 + n4 + n5)/5;
 if (n1>media) contA=contA+1;
 if (n2>media) contA=contA+1;
 if (n3>media) contA=contA+1;
 if (n4>media) contA=contA+1;
 if (n5>media) contA=contA+1;
 printf ("%lf %d", media, contA);
 system("pause");
}
```

```c
#include <stdio.h>
main()
{
 int contA=0;
 double media, n1, n2, n3, n4, n5;
 scanf ("%lf %lf %lf %lf %lf",
              &n1, &n2, &n3, &n4, &n5);
 media = (n1+ n2 + n3 + n4 + n5)/5;
 if (n1>media) contA=contA+1;
 if (n2>media) contA=contA+1;
 if (n3>media) contA=contA+1;
 if (n4>media) contA=contA+1;
 if (n5>media) contA=contA+1;
 printf ("%lf %d", media, contA);
 system("pause");
}
```

```c
#include <stdio.h>
main()
{
  int i, contA=0;
  double soma=0, media;
  double v[5];

  for (i=0;i<5;i++) {
     scanf ("%lf", &v[i]);
     soma = soma + v[i];
  }
  media = soma/i;
  for (i=0;i<5;i++) {
      if (v[i]>media) contA=contA+1;
  }
  printf ("%lf %d\n", media, contA);
  system("PAUSE");
}
```

Exercícios

16) Dada duas seqüências de 5 números, calcule a soma de cada número com o seu correspondente.

17) Ler 5 números, armazenando-os no vetor X[5]. Calcular a soma destes 5 números e mostrá-la na tela.

18) Ler 5 números, armazenando-os no vetor X[5]. Copiar este vetor, de trás para frente (na ordem inversa de leitura), em um segundo vetor Y[5]. Mostrar o vetor Y.

19) Gerar um vetor com 10 números da seguinte forma: cada número guardado no vetor será o valor da sua própria posição no vetor (seu índice) menos 2 (ou seja, Xi = i-2). Mostrar na tela este vetor.

Exercícios

20) Leia um vetor com N elementos, encontre e escreva o maior e o menor elemento e suas respectivas posições no vetor.

Structs

Variável que contém outras variáveis de tipos diferentes.

Equivalente ao que se denomina “registro” em outras linguagens de programação.

Cria-se um novo tipo de dado.

SINTAXE:

struct <identificador> {

<listagem dos tipos e campos>;

}

EXEMPLO:

```
struct aluno {
        char nome[50];
        int matricula;
        float media;
};
aluno escola[5];
```

Strings

- Não existe um tipo String em C.
- Strings em C são uma array do tipo char que termina com ‘\0’.
- Para literais String, o próprio compilador coloca ‘\0’.

```
#include <stdio.h>
main(){
    char re[] = "lagarto";
    printf ("%s", re);
    system("pause");
}
```

Para ler uma String

- Comando gets

```
#include <stdio.h>
main(){
    char re [80];
    printf ("Digite o seu nome: ");
    gets(re);
    printf ("Oi %s\n", re);
    system("pause");
}
```

Para comparar duas strings

- strcmp (s1, s2); strcmp retorna 0 se as duas strings são iguais.
- Precisa do #include <string.h>

```
#include <stdio.h>
#include <string.h>
main( ){
    char re[80];
    printf ("Digite a senha: ");
    scanf ("%s", &re);
    if (strcmp(re,"laranja")==0) {
        printf ("Senha correta\n");
    }
    else {
        printf ("Senha invalida\n");
    }
    system("pause");
}
```

Pode utilizar scanf no lugar do gets, só que o scanf não lê espaços em branco.
Se o usuário digitar:
>“Alexandre Costa e Silva”
o scanf vai pegar apenas “Alexandre”.
O gets pega tudo.

Para saber o tamanho de uma string

- `int size = strlen(str);`
  - Retorna um valor inteiro com o número de caracteres da String.
  - Precisa do #include <string.h>

```
#include <stdio.h>
#include <string.h>
main(){
    char re[80];
    printf ("Digite a palavra: ");
    scanf ("%s", &re);
    int size=strlen(re);
    printf ("Esta palavra tem %d caracteres.\n", size);
    system("pause");
}
```

Para copiar o conteúdo de uma string para outra

- strcpy(para, de);
- Precisa do #include <string.h>

```
#include <stdio.h>
#include <string.h>
main()
{
 char str[80];
 strcpy (str, "Alo");
 printf ("%s", str);
 system("pause");
}
```

Exercícios

20) Escreva um programa que leia duas strings e as coloque na tela. Imprima também a segunda letra de cada string.

21) Escreva um programa que leia uma string, conte quantos caracteres desta string são iguais a 'a' e substitua os que forem iguais a 'a' por 'b'. O programa deve imprimir o número de caracteres modificados e a string modificada.

22) Faça um programa que verifique se uma palavra é simétrica. Por exemplo: “arara” é uma palavra simétrica. Podemos lê-la normalmente ou de trás para a frente e sempre obteremos a mesma palavra.

Funções

- São estruturas que permitem ao programador separar o código do seu programa em blocos.
- Uma função tem a seguinte forma :

```
tipo_de_retorno Nome_da_funcao (parâmetros){
  /*corpo da função */
}
```

Funções que não retornam valor

```
#include <stdio.h>
void ehPar (int x){
    if (x % 2) {
        printf ("O numero nao eh par!\n");
    }
    else {
        printf ("O numero eh par!\n");
    }
}
int main(){
    char re[80];
    ehPar (3);
    system("pause");
    return (0);
}
```

Funções que retornam valor

```
#include <stdio.h>
int ehPar (int x){
    int z = 0;
/* o operador % retorna o resultado da divisão por 2 */
    if (x % 2) return 1;
}
int main(){
    char re[80];
    int i = ehPar (3);
    if (i!=0){
        printf ("O numero eh par!");
    }
    system("pause");
    return (0);
}
```

Exercícios

23) Construa um programa que tenha uma função que verifica se um número inteiro, passado como parâmetro, e exiba na tela se o número é negativo ou positivo.

24) Elabore um programa que tenha uma função que retorne o reverso de um número inteiro. Por exemplo, 932-> 239.

Exercício

25) Construa um programa que possua uma função que dado um número de conta corrente com cinco dígitos, retorne o seu dígito verificador, o qual é calculado da seguinte maneira:

Exemplo: número da conta: 25678

a) somar número da conta com seu inverso: 25678+87652=113330

b) multiplicar cada dígito pela sua ordem posicional e somar este resultado: 1 1 3 3 3 0

$$
\begin{array}{cccccc}
1 & 1 & 3 & 3 & 3 & 0 \\
+ & & & & & \\
0 & 1 & 2 & 3 & 4 & 5 \\
\hline
\end{array}
$$

$$1 + 1 + 6 + 9 + 12 + 0 = 29$$