Universidade Federal do Espírito Santo

Universidade Federal
do Espírito Santo

Programação II

INF09330

Departamento de Informática
Centro Tecnológico
Universidade Federal do Espírito Santo

Linguagem C

- Linguagem de médio nível, estruturada e flexível
- Geram programas objeto pequenos e eficientes
- É uma linguagem de uso genérico
- Surgiu nos anos 70, criada por Dennis Ritchie em um computador DEC PDP-11 que utilizava o sistema operacional UNIX

Dennis Ritchie

A estrutura de um bom programa deve conter...

Universidade Federal do Espírito Santo

- Cabeçalho
- Dicionário de dados
- Corpo
- Documentação
- Boa formatação (identação)

A estrutura de um bom programa deve conter...

Universidade Federal do Espírito Santo

- Cabeçalho

```
/*
 * media.c
 * criado em: 11/03/2019
 *
 * autora: Claudia Boeres
 *
 */

#include <stdio.h>
int main( )
{
        float a, b, media;
        // definição dos dados de entrada
        a = 10;
         b = 3;
        // cálculo da média
        media = (a + b)/2;

        return 0;
}
```

A estrutura de um bom programa deve conter...

Universidade Federal do Espírito Santo

- Cabeçalho
- Dicionário de dados

```
/*
 * media.c
 * criado em: 11/03/2019
 *
 * autora: Claudia Boeres
 *
 */

#include <stdio.h>
int main( )
{
        float a, b, media;
        // definição dos dados de entrada
        a = 10;
         b = 3;
        // cálculo da média
        media = (a + b)/2;

        return 0;
}
```

A estrutura de um bom programa deve conter...

Universidade Federal do Espírito Santo

- Cabeçalho
- Dicionário de dados
- Corpo

```
/*
 * media.c
 *
 * criado em: 11/03/2019
 *
 * autora: Claudia Boeres
 *
 */

#include <stdio.h>
int main( )
{
        float a, b, media;
        // definição dos dados de entrada
        a = 10;
         b = 3;
        // cálculo da média
        media = (a + b)/2;

        return 0;
}
```

A estrutura de um bom programa deve conter...

Universidade Federal do Espírito Santo

- Cabeçalho
- Dicionário de dados
- Corpo
- Documentação

```
/*
 * media.c
 *
 * criado em: 11/03/2019
 *
 * autora: Claudia Boeres
 *
 */

#include <stdio.h>
int main( )
{
        float a, b, media;
        // definição dos dados de entrada
        a = 10;
         b = 3;
        // cálculo da média
        media = (a + b)/2;

        return 0;
}
```

A estrutura de um bom programa deve conter...

Universidade Federal do Espírito Santo

- Cabeçalho
- Dicionário de dados
- Corpo
- Documentação
- Boa formatação (identação)

```
/*
 * media.c
 *
 * criado em: 11/03/2019
 *
 * autora: Claudia Boeres
 *
 */

#include <stdio.h>
int main( )
{
        float a, b, media;
        // definição dos dados de entrada
        a = 10;
        b = 3;
        // cálculo da média
        media = (a + b)/2;

        return 0;
}
```

Tipos de Dados

- Existem 5 tipos básicos em C:
  - **char, int, float, double e void**;

  - **char**: um único caracter. Ex: 'z';
  - **int**: número inteiro. Ex: 34;
  - **float**: número real. Ex: 7.98567;
  - **double**: número real com intervalo mais amplo.

- O padrão ANSI estipula apenas a faixa mínima de cada tipo de dado;

- O tipo **void** é um tipo especial: não é utilizado para definir variáveis;

Tipos de Dados

- O Padrão ANSI (89) define os seguintes tipos de dados:

Tipo	Num de bits	Intervalo	
		Inicio	Fim
char	8	-128	127
unsigned char	8	0	255
signed char	8	-128	127
int	16	-32.768	32.767
unsigned int	16	0	65.535
signed int	16	-32.768	32.767
short int	16	-32.768	32.767
unsigned short int	16	0	65.535
signed short int	16	-32.768	32.767
long int	32	-2.147.483.648	2.147.483.647
signed long int	32	-2.147.483.648	2.147.483.647
unsigned long int	32	0	4.294.967.295
float	32	3,4E-38	3.4E+38
double	64	1,7E-308	1,7E+308
long double	80	3,4E-4932	3,4E+4932

Tipos de Dados

- O tipo char, apesar de definir símbolos, é codificado por números inteiros.

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

Constantes

- Constantes não podem ser modificadas durante a execução do programa;
- Constantes diferem em relação a variáveis pois não utilizam posições de memória durante a execução do programa;

- Definição de constantes em C:

```
#define <identificador> <valor>
```

- Exemplo:

```
/*
 * areaCirculo.c
 * criado em: 11/03/2019
 * autora: Claudia Boeres
 *Utilização de constantes
 */

#include <stdio.h>
#define PI 3.141593

int main( )
{
        float area, raio;
        // definição dos dados de entrada
        raio = 3.5;
        // cálculo da área
        area = PI * raio * raio;

        return 0;
}
```

Expressões

- As variáveis e constantes podem ser combinadas com os operadores associados a cada tipo de dado, gerando expressões;
- Podem ser expressões aritméticas, relacionais e lógicas, dependendo do tipo de dado calculado e operadores utilizados;

Expressões aritméticas

Formadas por funções matemáticas e operadores aritméticos (Multiplicação (*), divisão (/) e resto (%) da divisão, adição (+) e subtração (-))

- Exemplo:

```c
#include <math.h>
#include <stdlib.h>
#define PI 3.1415

int main()
{
    float a , b , c , delta , raiz1 , raiz2;
    float x , y , z;

    a = 10;
    b = 50;
    c = 30;

    delta = b * b - 4 * a * c ;/*delta = ?*/
    raiz1 = -b + sqrt(delta)/(2*a);
    raiz2 = -b - sqrt(delta)/(2*a);

    x = sin( - PI / 2);
    y = fabs( x );
    z = pow(y ,2); // y elevado à potência 2

    return 0;
}
```

Expressões Relacionais

- retornam um valor booleano (verdadeiro ou falso), porém utilizam números para codificar esse tipo.

N diferente de 0 ≡ verdadeiro
N igual a 0 ≡ falso

Símbolo	Nome	Exemplo
<	menor que	a < 10
<=	menor ou igual a	x <= y
==	igual a	4 == 2*2
>	maior que	t > 0
>=	maior ou igual a	delta >= 0
!=	diferente de	x != 9+8*8

Tabela 2.3: Operadores relacionais.

Expressões lógicas

- relacionam os resultados de um conjunto de operações relacionais ou lógicas.

- "&&" (AND): Realiza o "E" lógico;
- "||" (OR): Realiza o "OU" lógico;
- "!" (NOT): Realiza a negação.

P	Q	P && Q	P || Q	!P	!Q
V	V	V	V	F	F
V	F	F	V	F	V
F	V	F	V	V	F
F	F	F	F	V	V

Tabela 2.4: Todas as combinações possíveis para os operadores lógicos.

Ordem de prioridade

- Aritméticas:
  - Funções matemáticas:
    - Exemplos: abs(), fabs(), sin(), cos(), sqrt(), abs(), pow(), ceil(), floor(), log() (neperiano), exp(), etc;
  - Multiplicação (*), divisão (/) e resto (%) da divisão;
  - Adição (+) e subtração (-)
- Relacionais:
  - Menor (<), menor ou igual (<=), maior (>), maior ou igual (>=), igual (==) e diferente (!=)
- Lógicas:
  - Negação (!)
  - Conjunção: e lógico (&&)
  - Disjunção: ou lógico (||)

Ordem de prioridade

- Aritméticas:
  - Funções matemáticas:
    - Exemplos: abs(), fabs(), sin(), cos(), sqrt(), abs(), pow(), ceil(), floor(), log() (neperiano), exp(), etc;
  - Multiplicação (*), divisão (/) e resto (%) da divisão
  - Adição (+) e subtração (-)
- Relacionais:
  - Menor (<), menor ou igual (<=), maior (>), maior ou igual (>=), igual (==) e diferente (!=)
- Lógicas:
  - Negação (!)
  - Conjunção: e lógico (&&)
  - Disjunção: ou lógico (||)

Exemplo:

2 + 1 * 2 → 4
(2 + 1) * 2 → 6

Exemplo:

10 > 5 || 3==3 && 7 != 7 → Verdadeiro
(10 > 5 || 3==3) && 7 != 7 → Falso

Comando de Atribuição

- Serve para alterar os valores (conteúdo) das variáveis.
- Exemplo:

```
/*
 * media.c
 *
 * criado em: 18/03/2019
 *
 * autora: Claudia Boeres
 */
#include <stdio.h>
int main( )
{ // declaração dos tipos e
  // definição dos dados de entrada

        int leituraAtual = 125;
        int leituraAnterior = 25;
        float valorUnitario = 2.5;
        int diferenca;
        float valorConta;

        diferenca = leituraAtual – leitura Anterior;
        valorConta  = diferença * valorUnitario;

        return 0;
}
```

- Qual o valor da variável **valorConta** ao final da execução do programa?

Entrada e Saída (I/O)

- Comando para entrada de dados:

```
scanf("<formato1> ... <formatoN>", &var1, ... , &varN);
```

- Para cada formato depende do tipo da variável ou da forma como se deseja visualizá-la;
- Os formatos básicos são: %d ou %i (inteiros com sinal), %f (reais), %e (reais mais longos) e %c (char);

Entrada e Saída (I/O)

- Exemplo:

```c
#include <stdlib.h>
#include <stdio.h>

int main()
{
    int leituraAtual , leituraAnterior , diferenca ;
    float valorUnitario , valorConta ;

    scanf ( "%d%d%f" , &leituraAtual , &leituraAnterior , &valorUnitario ) ;
    diferenca = leituraAtual - leituraAnterior ;
    valorConta = diferenca * valorUnitario ;

    return 0;
}
```

Entrada e Saída (I/O)

- Comando para saída de dados:

```
printf("<formato1> ... <formatoN>", exp1, ... , expN);
```

- O formato depende do tipo da variável ou da forma como se deseja visualizá-la;
- Os formatos básicos são: %d ou %i, %f, %e e %c;
- %% escreve o caracter %;

Entrada e Saída (I/O)

- Exemplo:

```c
#include <stdlib.h>
#include <stdio.h>

int main()
{
    int leituraAtual , leituraAnterior , diferenca ;
    float valorUnitario , valorConta ;

    printf ( "Digite o valor da leitura ATUAL : " ) ;
    scanf ( "%d" , & leituraAtual ) ;

    printf ( "Digite o valor da leitura ANTERIOR : " ) ;
    scanf ( "%d" , & leituraAnterior ) ;

    printf ( "Digite o preco do Quilowatt - hora : " ) ;
    scanf ( "%f" , & valorUnitario ) ;

    diferenca = leituraAtual - leituraAnterior ;
    valorConta = diferenca * valorUnitario ;

    printf ( "Valor da conta R$ : %f", valorConta ) ;


    return 0;
}
```

Entrada e Saída (I/O)

- Formatos de dados definidos no padrão ANSI :

Tipo	Num de bits	Formato para leitura com scanf	Intervalo	
			Inicio	Fim
char	8	%c	-128	127
unsigned char	8	%c	0	255
signed char	8	%c	-128	127
int	16	%i	-32.768	32.767
unsigned int	16	%u	0	65.535
signed int	16	%i	-32.768	32.767
short int	16	%hi	-32.768	32.767
unsigned short int	16	%hu	0	65.535
signed short int	16	%hi	-32.768	32.767
long int	32	%li	-2.147.483.648	2.147.483.647
signed long int	32	%li	-2.147.483.648	2.147.483.647
unsigned long int	32	%lu	0	4.294.967.295
float	32	%f	3,4E-38	3.4E+38
double	64	%lf	1,7E-308	1,7E+308
long double	80	%Lf	3,4E-4932	3,4E+4932

Comandos de Seleção

- O comando de seleção permite que um programa possa realizar diferentes alternativas de sequências de instruções durante sua execução;

- Dependendo do valor de uma expressão ou de uma variável, o programa segue executando uma ou outra sequência de comandos.

Comandos de Seleção

- Seleção simples:

```
if(<expressão lógica>)
{
    <sequência de comandos>
}
```

- Exemplo:

```c
#include <stdlib.h>
#include <stdio.h>

int main()
{
    int num1 , num2 , aux ;

    printf(" Digite dois numeros inteiros separados por espacos : " ) ;

    scanf("%d%d" , & num1 , & num2 ) ;
    printf("Valores digitados : %d %d \n" , num1 , num2 ) ;

    if ( num1 > num2 )
    {
        aux = num1 ;
        num1 = num2 ;
        num2 = aux ;
    }

    printf( "Valores ordenados : %d %d \n", num1 , num2 ) ;

    return 0;
}
```

Comandos de Seleção

- Seleção dupla:

```
if(<expressão lógica>)
{
    <sequência de comandos>
}
else
{
    <sequência de comandos>
}
```

Comandos de Seleção

- Exemplo:

```c
#include <stdlib.h>
#include <stdio.h>

int main()
{
    int n1 , n2 , n3 ;

    printf ( "Entre com os tres numeros separados por espacos : " ) ;
    scanf ( "%d%d%d", &n1, &n2, &n3 ) ;

    if ( n1 > n2 )
    {
        if ( n1 > n3 )
        {
            printf ( "O maior numero digitado foi: %d \n", n1 );
        }
        else
        {
            printf ( "O maior numero digitado foi: %d \n", n3 );
        }
    }
    else
    {
        if ( n2 > n3 )
        {
            printf ( "O maior numero digitado foi: %d \n", n2 );
        }
        else
        {
            printf ( "O maior numero digitado foi: %d \n", n3 );
        }
    }

    return 0;
}
```

Comandos de Seleção

- Seleção múltipla:

```
switch (<expressão>)
{
   case <valor1>: <sequência de comandos 1>
                  break;
   case <valor2>: <sequência de comandos 2>
                  break;
   ...
   ...
   case <valorN>: <sequência de comandos N>
                  break;
   default: <sequência de comandos>
}
```

Comandos de Seleção

- Exemplo:

```c
#include <stdlib.h>
#include <stdio.h>

int main()
{
    int numero ;

    printf ("URNA ELETRONICA - SEU VOTO PARA PREFEITO : " );
    scanf ("%d", &numero);

    switch(numero)
    {
        case 1: printf("Candidato escolhido : Hortencia da Silva.\n");
                break;
        case 2: printf("Candidato escolhido : Jose dos Cravos.\n");
                break;
        case 3: printf("Candidato escolhido : Margarida S. Pereira.\n");
                break;
        default:printf("Numero digitado invalido. Voto anulado.\n");
                break;
    }

    return 0;
}
```

Exercícios

1. Pesquise a história da linguagem C;

2. Crie programas em C que resolvam os seguintes problemas:

   - Calcular o volume de uma esfera tendo como entrada o valor do seu raio.
   - Dizer se um triângulo é retângulo, dados os três lados do mesmo;
   - Crie uma calculadora usando a instrução SWITCH, que pergunte qual das operações básicas quer fazer (+, -, * e /), em seguida peça os dois números e mostre o resultado da operação matemática entre eles.