

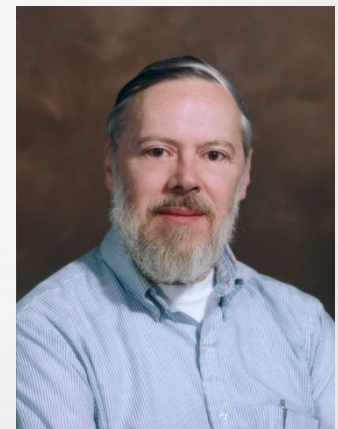
Programação II

Prof.^a Claudia Boeres (boeres@inf.ufes.br)

**CT IX - Sala 206
Departamento de Informática
Centro Tecnológico
Universidade Federal do Espírito Santo**

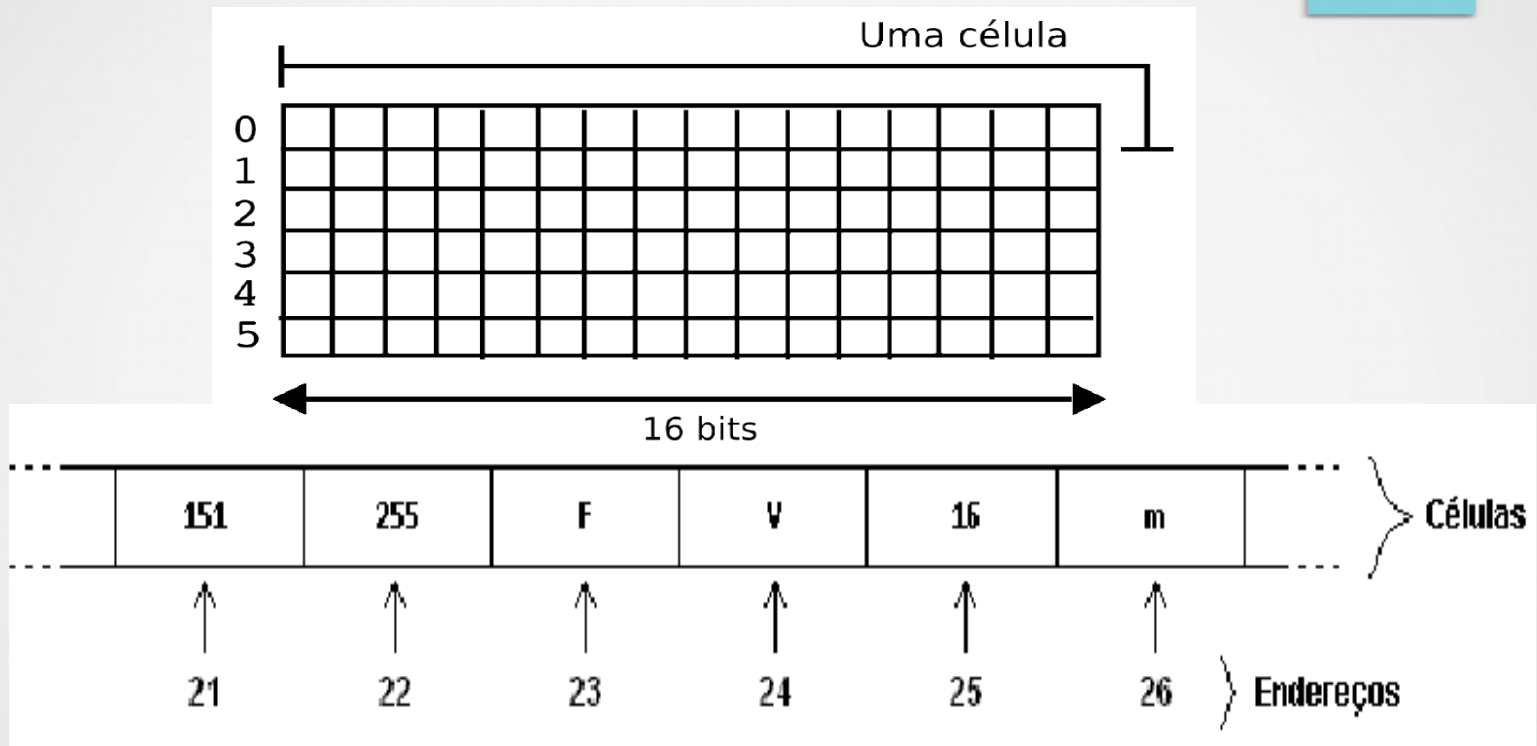
Linguagem C

- Linguagem de alto nível, estruturada e flexível
- Geram programas objeto pequenos e eficientes
- É uma linguagem de uso genérico
- Surgiu nos anos 70, criada por Dennis Ritchie



Dennis Ritchie

Células de Memória



Variáveis

- Nome dado ao local da memória capaz de armazenar um valor.
- No programa, através do nome da variável é possível acessar o valor (ou conteúdo) que lá está.
- Podemos dizer que uma variável nada mais é do que uma abstração para o endereço de memória.

Identificadores

- ▶ Em geral, as linguagens de alto nível possuem dois tipos de elementos: os elementos definidos pela própria linguagem (símbolos para operadores, nome de comandos, etc), e os elementos definidos pelo programador (identificadores, comentários, etc);
- ▶ Um identificador é um símbolo que pode representar alguma entidade criada pelo programador, como uma **variável** por exemplo;
- ▶ Cada linguagem define uma regra para formação de identificadores;

Identificadores e Variáveis

- ▶ Em geral, sempre é possível:
 - Utilizar uma sequência de caracteres alfanuméricos;
 - Os caracteres devem ser letras ou números sem acentos e sem cedilha;
 - O primeiro caractere de um identificador deve ser obrigatoriamente uma letra;

Identificadores e Variáveis

► Exemplos:

```
1    abc
2    x1
3    y2
4    letra
5    SOMA_TOTAL
6    B_32
```

Exemplo 2.1: Nomes válidos de variáveis, concordando com as regras de nomenclatura.

```
1    fim?           // '?' não é um caractere alfanumérico
2    %percentual%   // '%' não é um caractere alfanumérico
3    123quatro      // Iniciado por número
4    !hola!         // '!' não é um caractere alfanumérico
5    @ARROBA        // '@' não é um caractere alfanumérico
```

Exemplo 2.2: Nomes inválidos de variáveis.

Identificadores e Variáveis

- ▶ Algumas linguagens fazem diferenciação entre letras maiúsculas e minúsculas.
- ▶ Escolher nomes para as variáveis que sejam intuitivas quanto ao seu uso é uma boa política
- ▶ É recomendável adotar padrões para a escrita de identificadores
- ▶ Normalmente, em grandes projetos de software, são adotados padrões para a escrita dos identificadores a fim de que os programadores possam trocar seus códigos, entendê-los e alterá-los sem grande dificuldade.

Identificadores e Variáveis

- ▶ Padrões adotados na disciplina:
 - ▶ **Nomes simples:** começando com letra minúscula e demais caracteres minúsculos;
 - ▶ **Nomes compostos:** primeira parte iniciada por letra minúscula e as demais partes iniciadas por letra maiúscula. Os demais caracteres são minúsculos.
- ▶ Exemplo:

```
1  delta
2  raiz1
3  idade
4  letra
5  percentualDeLucro
6  primeiraLetra
7  indiceBovespa
```

Exemplo 2.3: Nomes significativos para variáveis.

Comando de Atribuição

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

```
1 /*  
2  * main.c  
3  *  
4  * Created on: 20/11/2012  
5  * Author: clebson  
6  */  
7  
8 #include <stdlib.h>  
9  
10 int main()  
11 {  
12     int leituraAtual = 125;  
13     int leituraAnterior = 25;  
14     float valorUnitario = 2.5;  
15     int diferenca;  
16     float valorConta;  
17  
18     diferenca = leituraAtual - leituraAnterior ;  
19     valorConta = diferenca * valorUnitario ;  
20  
21     return 0;  
22 }
```

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

Tipos de Dados

- ▶ Um tipo de dado delimita o conjunto de valores possíveis que uma determinada variável pode representar e suas operações básicas;
- ▶ São necessários porque uma única célula de memória representa um conjunto de dados muito limitado;
- ▶ Tipos de dados são abstrações sobre palavras de memória;
- ▶ **O tamanho de cada tipo de dado varia com a implementação do compilador/interpretador e com o tipo do processador utilizado;**

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	
		Início	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 letras 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	SYM (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		DEL

Source: www.LookupTables.com

Constantes

- ▶ 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:

```
2  * main.c
3  *
4  *   Created on: 20/11/2012
5  *   Author: clebson
6  */
7
8  /*Utilizacao de constantes*/
9
10 #include <stdlib.h>
11
12 # define PI 3.141593
13 # define FALSO 0
14 # define VERDADEIRO 1
15
16 int main()
17 {
18     float area = 0, raio = 3.5;
19
20     raio = PI*raio*raio;
21
22     return 0;
23 }
```


Expressões

- ▶ As variáveis e constantes podem ser combinadas com os operadores associados a cada tipo de dado, gerando expressões;
- ▶ Ordem de prioridade para avaliação:
 - Aritméticas:
 1. Funções matemáticas: necessário incluir a biblioteca `math.h` e adicionar o comando `-lm` à chamada do compilador. Ex: `abs()`, `fabs()`, `sin()`, `cos()`, `sqrt()`, `abs()`, `pow()`, `ceil()`, `floor()`, `log()` (`neperiano`), `exp()`, etc;
 2. Multiplicação (*), divisão (/) e resto (%) da divisão;
 3. Adição (+) e subtração (-).

Expressões

► Exemplo:

```
1 #include <math.h>
2 #include <stdlib.h>
3 #define PI 3.1415
4
5 int main()
6 {
7     float a , b , c , delta , raiz1 , raiz2;
8     float x , y , z;
9
10    a = 10;
11    b = 50;
12    c = 30;
13
14    delta = b * b - 4 * a * c ; /*delta = ?*/
15    raiz1 = -b + sqrt(delta)/(2*a);
16    raiz2 = -b - sqrt(delta)/(2*a);
17
18    x = sin( - PI / 2);
19    y = fabs( x );
20    z = pow(y ,2); // y elevado à potência 2
21
22    return 0;
23 }
24
```

Expressões

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

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

Símbolo	Nome	Exemplo
<	menor que	$a < 10$
<=	menor ou igual a	$x \leq y$
==	igual a	$4 == 2*2$
>	maior que	$t > 0$
>=	maior ou igual a	$\text{delta} \geq 0$
!=	diferente de	$x \neq 9+8*8$

Tabela 2.3: Operadores relacionais.

Expressões

- Lógicas: relacionam os resultados de um conjunto de operações relacionais.

- “&&” (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.

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:

```
1 #include <stdlib.h>
2 #include <stdio.h>
3
4 int main()
5 {
6     int leituraAtual , leituraAnterior , diferenca ;
7     float valorUnitario , valorConta ;
8
9     scanf ( "%d%d%f" , &leituraAtual , &leituraAnterior , &valorUnitario ) ;
10    diferenca = leituraAtual - leituraAnterior ;
11    valorConta = diferenca * valorUnitario ;
12
13    return 0;
14 }
15
```

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:

```
1 #include <stdlib.h>
2 #include <stdio.h>
3
4 int main()
5 {
6     int leituraAtual , leituraAnterior , diferenca ;
7     float valorUnitario , valorConta ;
8
9     printf ( "Digite o valor da leitura ATUAL : " ) ;
10    scanf ( "%d" , & leituraAtual ) ;
11
12    printf ( "Digite o valor da leitura ANTERIOR : " ) ;
13    scanf ( "%d" , & leituraAnterior ) ;
14
15    printf ( "Digite o preco do Quilowatt - hora : " ) ;
16    scanf ( "%f" , & valorUnitario ) ;
17
18    diferenca = leituraAtual - leituraAnterior ;
19    valorConta = diferenca * valorUnitario ;
20
21    printf ( "Valor da conta R$ : %f", valorConta ) ;
22
23
24    return 0;
25 }
26 |
```

Entrada e Saída (I/O)

- ▶ Formatos de dados definidos no padrão ANSI:

Tipo	Num de bits	Formato para leitura com scanf	Intervalo	
			Início	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:

```
1 #include <stdlib.h>  
2 #include <stdio.h>  
3  
4 int main()  
5 {  
6     int num1 , num2 , aux ;  
7  
8     printf(" Digite dois numeros inteiros separados por espacos : " ) ;  
9  
10    scanf("%d%d" , & num1 , & num2 ) ;  
11    printf("Valores digitados : %d %d \n" , num1 , num2 ) ;  
12  
13    if ( num1 > num2 )  
14    {  
15        aux = num1 ;  
16        num1 = num2 ;  
17        num2 = aux ;  
18    }  
19  
20    printf( "Valores ordenados : %d %d \n", num1 , num2 ) ;  
21  
22    return 0 ;  
23 }  
24
```

Comandos de Seleção

- Seleção dupla:

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

Comandos de Seleção

► Exemplo:

```
1 #include <stdlib.h>
2 #include <stdio.h>
3
4 int main()
5 {
6     int n1 , n2 , n3 ;
7
8     printf ( "Entre com os tres numeros separados por espacos : " ) ;
9     scanf ( "%d%d%d", &n1, &n2, &n3 ) ;
10
11     if ( n1 > n2 )
12     {
13         if ( n1 > n3 )
14         {
15             printf ( "O maior numero digitado foi: %d \n", n1 );
16         }
17         else
18         {
19             printf ( "O maior numero digitado foi: %d \n", n3 );
20         }
21     }
22     else
23     {
24         if ( n2 > n3 )
25         {
26             printf ( "O maior numero digitado foi: %d \n", n2 );
27         }
28         else
29         {
30             printf ( "O maior numero digitado foi: %d \n", n3 );
31         }
32     }
33
34     return 0;
35 }
36
```

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:

```
1 #include <stdlib.h>
2 #include <stdio.h>
3
4 int main()
5 {
6     int numero ;
7
8     printf ("URNA ELETRONICA - SEU VOTO PARA PREFEITO : " );
9     scanf ("%d", &numero);
10
11     switch(numero)
12     {
13         case 1: printf("Candidato escolhido : Hortencia da Silva.\n");
14                 break;
15         case 2: printf("Candidato escolhido : Jose dos Cravos.\n");
16                 break;
17         case 3: printf("Candidato escolhido : Margarida S. Pereira.\n");
18                 break;
19         default: printf("Numero digitado invalido. Voto anulado.\n");
20                 break;
21     }
22
23     return 0;
24 }
25 |
```

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.