Variáveis e Tipos de Dados

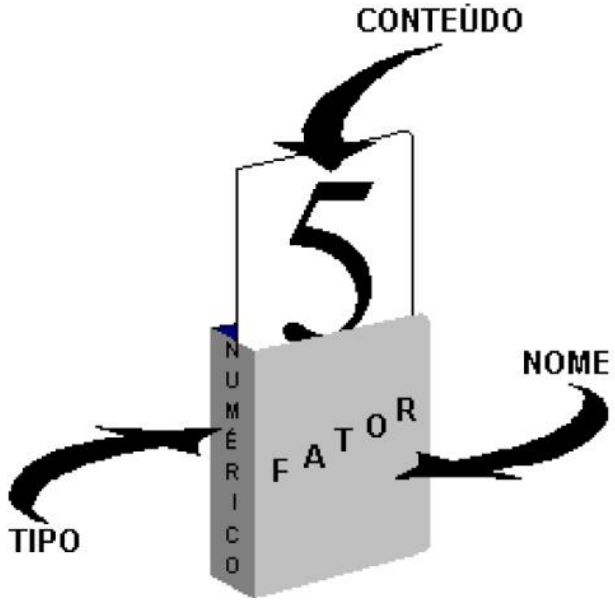
Profa. Elloá B. Guedes

www.elloaguedes.com

UEA
UNIVERSIDADE
DO ESTADO DO
AMAZONAS

Variáveis

- Já há uma familiaridade com o conceito de variável
- Na Programação, uma variável representa **uma posição na memória, possuindo um nome e um tipo**
- O conteúdo da memória indicado pela variável pode **mudar ao longo do tempo**
- Uma variável armazena **um valor de cada vez**



Variáveis em C

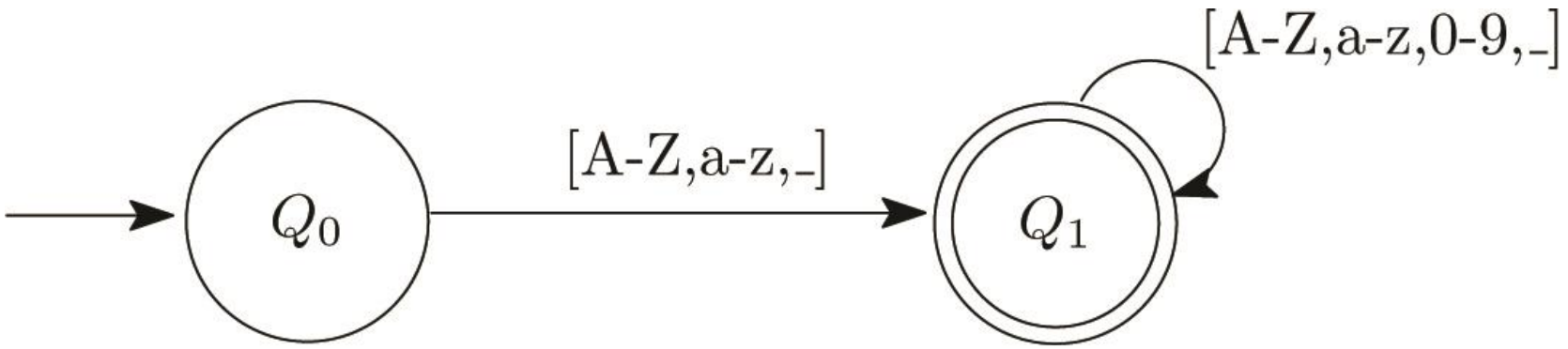
- Devem ser declaradas **antes** de serem utilizadas

- Identificadores de variáveis
  - Devem começar com uma letra ou com um sublinhado
  - Segue-se zero ou mais letras, sublinhados ou dígitos

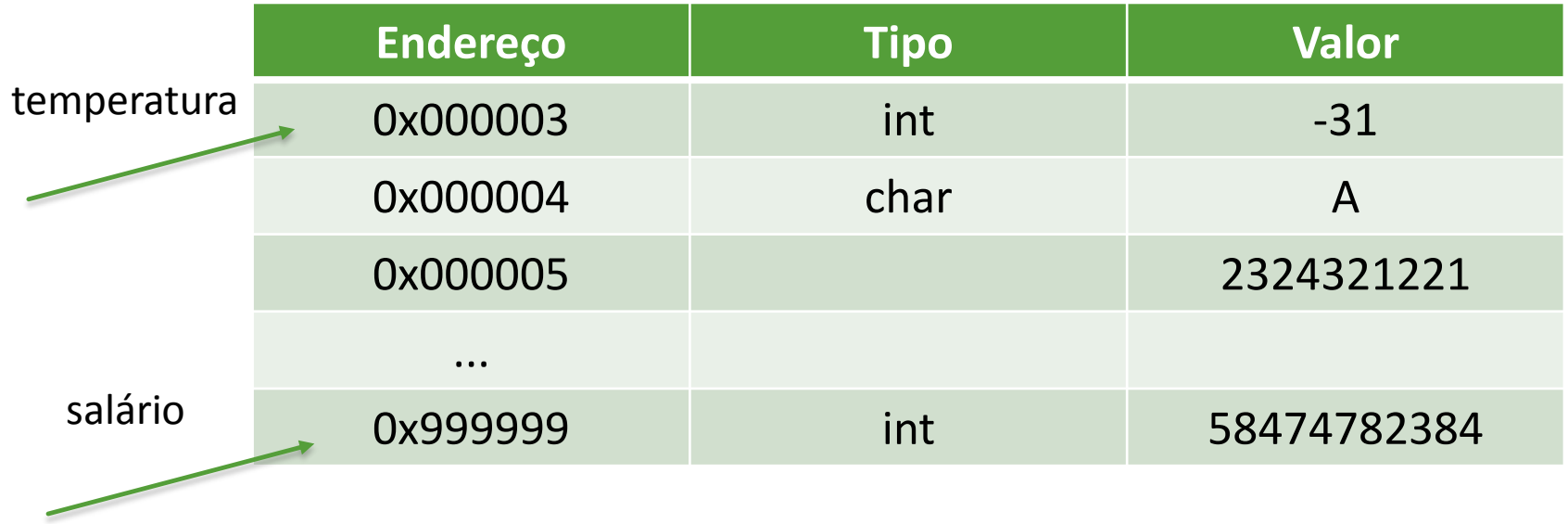
- Identificadores de variáveis válidos:
  - num,_nome, ID, nome_de_variavel, nomeDeVariavel

- Identificadores de variáveis inválidos
  - minha variável, uma$variável, 3numeros, int

Variáveis em C



Variáveis em C



Endereço	Tipo	Valor
0x000003	int	-31
0x000004	char	A
0x000005		2324321221
...		
0x999999	int	58474782384

Variáveis em C

- Declaração

```
tipo var1 | [; | var2, var3, ..., varx;]
```

- Estrutura geral de um programa

```
int main(){
  // Declaração de Variáveis
  ...
  // Instruções
  ...
  return 0;
}
```

Variáveis em C

- As **palavras reservadas** não podem ser utilizadas como identificadores de variáveis em C
  - Há muitas palavras reservadas: main, int, void, etc.

- A linguagem C é ***case sensitive***
  - Significa que há distinção entre maiúsculas e minúsculas
  - SOMA, sOmA, soma, SOMa podem ser variáveis válidas em um mesmo programa em C

- Nomes de variáveis devem ser **representativos**!
  - Facilitam o entendimento do código
  - Refletem o uso da variável

Comando de Atribuição

- Quando uma variável é declarada, o programador está solicitando ao compilador para reservar um espaço em memória para armazená-la

- O nome da variável referencia a totalidade do espaço ocupado pela variável

- Uma variável poderá ser iniciada com um valor por meio de uma operação de atribuição

```
variavel = expressao;
```

Tipos Básicos de Dados

- Em C há 5 tipos básicos de dados:
  - int
  - float
  - double
  - char
  - _Bool

Tipo int

- Consiste em uma sequência de um ou mais dígitos
- Podem ser positivos ou negativos

- Base decimal
  - Impressão com printf: %i ou %d

- Se inicia com 0, indica que o número está na **base octal**
  - Base 8
  - Impressão com printf: %o

- Se inicia com 0x, indica que o número está na **base hexadecimal**
  - Base 16
  - Impressão com printf: %x

Exemplos de Atribuição

```c
#include <stdio.h>

int main(){

    int num;
    int n = 1;
    int n1 = 3; n2 = 5;
    int a = b = c = d = 0;
    num = 17;


    return 0;
}
```

Operações sobre inteiros

Operação	Descrição	Exemplo	Resultado
+	Soma	`21 + 4`	25
–	Subtração	`21 - 4`	17
*	Multiplicação	`21 * 4`	84
/	Divisão Inteira	`21 / 4`	5
%	Resto da Divisão Inteira (**Módulo**)	`21 % 4`	1

Função scanf

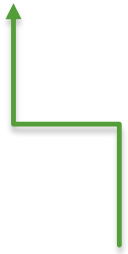
- Permite a leitura de valores do teclado e a sua respectiva atribuição à variáveis

- Sintaxe
  - Para leitura de inteiros
  - scanf(“%d”,&variavelInteira);

Função scanf

- Permite a leitura de valores do teclado e a sua respectiva atribuição à variáveis

- Sintaxe
  - Para leitura de inteiros
  - scanf(“%d”,&variavelInteira);



É um operador que permite que o valor lido seja atribuído à variável

Variações do tipo inteiro

Tipo de Variável	Nº de *Bytes*	Valor Mínimo	Valor Máximo
`int`	2	-32 768	32 767
`short int`	2	-32 768	32 767
`long int`	4	-2 147 483 648	2 147 483 647
`unsigned int`	2	0	65 535
`unsigned short int`	2	0	65 535
`unsigned long int`	4	0	4 294 967 295

Tipo float

- Armazena valores que possuem **casa decimal**
  - A casa decimal é indicada por um **ponto**
  - É possível omitir os dígitos antes do ponto, caso o número não tenha parte inteira (0.05 pode ser escrito como .05)
  - Impressão com printf: %f

- Utilização de notação científica
  - 1.7e4 = 1.7*(10^4)
  - Mantissa: 1.7
  - Expoente: 4
  - Impressão com printf: %e

Tipo float

- Impressão com %g
  - Representação mais adequada
  - Se o expoente é menor que -4 ou maior que 5, então %e
  - Senão, %f

Operações sobre float

Operação	Descrição	Exemplo	Resultado
`+`	Soma	`21.3 + 4.1`	`25.4`
`-`	Subtração	`21.7 - 4.8`	`16.9`
`*`	Multiplicação	`21.2 * 4.7`	`99.64`
`/`	Divisão Real	`21.0 / 4.0`	`5.25`
`%`	Não faz sentido aplicar a reais	`n.a.`	`n.a.`

Divisão com tipo float

```
21   / 4   ➔ 5    /* Divisão inteira */
21.0 / 4   ➔ 5.25 /* Como 21.0 é um real, o valor 4 é alterado para 4.0 */
21   / 4.  ➔ 5.25 /* Como 4. é um real, o valor 21 é alterado para 21.0 */
21.0 / 4.0 ➔ 5.25 /* Divisão real */
```

Tipo double

- Em algumas situações o tipo float pode não ser suficiente para armazenar um número
  - Aplicações Científicas
  - Aplicações da Engenharia
  - Cálculos Matemáticos
- Representa números reais positivos e negativos com 64 bits, normalmente o dobro do tipo float
- Impressão com printf: %f, %e e %g

Tipo char

- Armazena um único caractere por vez
  - Caractere deve ser armazenado entre aspas simples
  - ‘a’, ‘f’, ’\n’, ‘2’, ‘;’
  - Impressão com printf: %c
  - Armazenado em um byte

- Tipo string
  - Sequências de caracteres entre aspas duplas
  - printf(“Esta é uma string em C”);

Leitura de caracteres

- Pode ser feita com scanf e o símbolo %c
  - scanf(“%c”,&variavel);

- getchar() é uma função projetada especificamente para leitura de caracteres na linguagem C
  - variavel = getchar();

- Diferença entre scanf e getchar reside no buffer!
  - Se houver “lixo” no buffer, este deve ser limpo com fflush(stdin)

Leitura de caracteres

```c
#include <stdio.h>

int main(){

    char c, d;

    printf("Informe um caractere: ");
    scanf("%c",&c);
    //fflush(stdin);
    printf("\nInforme outro caractere: ");
    d = getchar();

    printf("O caractere c eh %c e o caractere d eh %c",c,d);


    return 0;
}
```

Caracteres e a Tabela ASCII

- Cada caractere em C corresponde a um valor de uma tabela chamada ASCII
  - Cada caractere corresponde a um número
  - Impressão com %d ou %i

Tabela ASCII

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

Caracteres e a Tabela ASCII

```
ch = 'A';      /* Formato tradicional */
ch = 65;       /* Caractere cujo código ASCII é 65 */
ch = '\101';   /* Caractere cujo código ASCII escrito em octal é 101 */
ch = '\x41';   /* Caractere cujo código ASCII escrito em hexa é   41 */
```

Tipo _Bool

- Representa valores booleanos ou lógicos
  - Verdadeiro e Falso
- Inclusão da biblioteca <stdbool.h>
- Ao criar uma variável do tipo _Bool
  - Falso = 0
  - Verdadeiro = Qualquer valor não-nulo
- Utilização de um valor do tipo int
  - Falso é representado por zero
  - Verdadeiro é representado por qualquer número não-negativo