Aula 03

MC 102 - Algoritmos e Programação de Computadores

Variáveis, Constantes e Expressões Numéricas

Armazenamento de Dados

Onde ficam os dados em um Computador?

- Na memória RAM (Principal)
- Na memória Secundária
  - HD, CD, DVD, Pen Drive, etc.

Armazenamento de Dados

Onde ficam os dados em um Computador?

- Na memória RAM (Principal) - Temporário
- Na memória Secundária - Definitivo
  - HD, CD, DVD, Pen Drive, etc.

Variáveis - Definição

Elementos de um programa que servem para o armazenamento temporário de dados quando este programa está em execução.

Objetivo: Armazenar valores na memória

Visibilidade: São locais, restritas ao programa

Valor: é dado por seu conteúdo -> é variável

Características: Devem possuir:

- Nome ou identificador
- Tipo do dado a ser armazenado

Variáveis - Exemplos

Analogia com a vida real

- Um Balde
  - Armazena até uma certa quantidade de líquidos
- Uma Gaveta
  - Armazena papéis.

Declaração: (define-se o tipo e o nome)

líquido Balde;

papel Gaveta;

Variáveis - Exemplos

Analogia com a vida real

- Um Balde
  - Armazena até uma certa quantidade de líquidos
- Uma Gaveta
  - Armazena papéis.

Declaração: (define-se o tipo e o nome)

líquido Balde;

papel Gaveta;

Tipo da variável

Variáveis - Exemplos

Analogia com a vida real

- Um Balde
  - Armazena até uma certa quantidade de líquidos
- Uma Gaveta
  - Armazena papéis.

Declaração: (define-se o tipo e o nome)

líquido Balde;

papel Gaveta;

Nome ou identificador

Variáveis - Exemplos

líquido Balde;

papel Gaveta;

Atribuição de valores

Balde = 5 litros água;

Gaveta = 500 folhas de papel;

Valor de Balde? Valor de Gaveta?

Balde = Balde + 12 litros água;

Gaveta = Gaveta - 215 folhas de papel;

Valor de Balde? Valor de Gaveta?

Variáveis - Tipos

Tipos de variáveis numéricas

- Dependentes da arquitetura ou do computador
  - Inteiros (**int**)
    - 32 bits (-2.147.483.648 a 2.147.483.647)
    - 64 bits (-4.294.967.296 a 4.294.967.295)
  - Inteiros não negativos (**unsigned int**)
    - 32 bits (0 a 4.294.967.295)
    - 64 bits (0 a 8.589.934.591)

Variáveis - Tipos

Tipos de variáveis numéricas

- Independentes do computador
  - Inteiros grandes (**long int**)
    - -2.147.483.648 a 2.147.483.647 (32 bits)
  - Inteiros grandes não negativos (**unsigned long int**)
    - 0 a 4.294.967.295 (32 bits)
  - Inteiros pequenos (**short int**)
    - -32.768 a 32.767 (16 bits)
  - Inteiros pequenos (**unsigned short int**)
    - 0 a 65.535 (16 bits)

Variáveis - Tipos

Tipos de variáveis reais (ponto flutuante)

$$(-1)^{sinal} \bullet mantissa \bullet 2^{expoente}$$

Exemplo:

$$0.5 = (-1)^0 \bullet 1 \bullet 2^{-1}$$

- Possuem problemas de precisão (arredondamento)
- São vistas como um número decimal para o programador

Variáveis - Tipos

Tipos de variáveis reais (ponto flutuante)

- 32 bits (**float**) $(-1)^{sinal} \bullet mantissa \bullet 2^{expoente}$
  - 1 bit p/ o sinal, 8 bits p/ o expoente e 23 bits p/ a mantissa
  - Valores aproximados entre -10^{-38} a $+10^{-38}$
- 64 bits (**double**)
  - 1 bit p/ o sinal, 11 bits p/ o expoente e 52 bits p/ a mantissa
  - Valores aproximados entre -10^{-308} a $+10^{-308}$

Variáveis - Tipos

Tipos de variáveis para caracteres

- Armazenam letras e outros símbolos em textos
- Na verdade, correspondem ao índice da tabela de símbolos utilizada nos computadores
  - A principal é a tabela ASCII – American Standard Code for Information Interchang
- 8 bits (**char**)
  - Inteiro correspondente ao símbolo (-128 a 127)
- 8 bits não negativos (**unsigned char**)
  - Inteiro correspondente ao símbolo (0 e 255)

Tabela ASCII

backspace	ctrl-H	8	08	BS	40	28	(	72	48	H	104	68	h
horizontal tab	ctrl-I	9	09	HT	41	29	)	73	49	I	105	69	i
line feed	ctrl-J	10	0A	LF	42	2A	*	74	4A	J	106	6A	j
vertical tab	ctrl-K	11	0B	VT	43	2B	+	75	4B	K	107	6B	k
form feed	ctrl-L	12	0C	FF	44	2C	,	76	4C	L	108	6C	l
carriage feed	ctrl-M	13	0D	CR	45	2D	-	77	4D	M	109	6D	m
shift out	ctrl-N	14	0E	SO	46	2E	.	78	4E	N	110	6E	n
shift in	ctrl-O	15	0F	SI	47	2F	/	79	4F	O	111	6F	o
data line escape	ctrl-P	16	10	DLE	48	30	0	80	50	P	112	70	p
device control 1	ctrl-Q	17	11	DC1	49	31	1	81	51	Q	113	71	q
device control 2	ctrl-R	18	12	DC2	50	32	2	82	52	R	114	72	r
device control 3	ctrl-S	19	13	DC3	51	33	3	83	53	S	115	73	s
device control 4	ctrl-T	20	14	DC4	52	34	4	84	54	T	116	74	t
neg acknowledge	ctrl-U	21	15	NAK	53	35	5	85	55	U	117	75	u
synchronous idel	ctrl-V	22	16	SYN	54	36	6	86	56	V	118	76	v
end of xmit block	ctrl-W	23	17	ETB	55	37	7	87	57	W	119	77	w
cancel	ctrl-X	24	18	CAN	56	38	8	88	58	X	120	78	x
end of medium	ctrl-Y	25	19	EM	57	39	9	89	59	Y	121	79	y
substitute	ctrl-Z	26	1A	SUB	58	3A	:	90	5A	Z	122	7A	z
escape	ctrl-[	27	1B	ESC	59	3B	;	91	5B	[	123	7B	{
file separator	ctrl-\	28	1C	FS	60	3C	<	92	5C	\	124	7C	|
group separator	ctrl-]	29	1D	GS	61	3D	=	93	5D	]	125	7D	}
record separator	ctrl-^	30	1E	RS	62	3E	>	94	5E	^	126	7E	~
unit separator	ctrl-_	31	1F	US	63	3F	?	95	5F	_	127	7F	DEL

Tabela ASCII estendida

128	Ç	144	É	160	á	176	▒	193	┴	209	╤	225	ß	241	±
129	ü	145	æ	161	í	177	▓	194	┬	210	╥	226	Γ	242	≥
130	é	146	Æ	162	ó	178	▓	195	├	211	╙	227	π	243	≤
131	â	147	ô	163	ú	179	|	196	─	212	╘	228	Σ	244	⌠
132	ä	148	ö	164	ñ	180	┤	197	┼	213	╒	229	σ	245	⌡
133	à	149	ò	165	Ñ	181	╡	198	╞	214	╓	230	µ	246	÷
134	å	150	û	166	ª	182	╢	199	╟	215	╫	231	τ	247	≈
135	ç	151	ù	167	º	183	╖	200	╚	216	╪	232	Φ	248	°
136	ê	152	_	168	¿	184	╕	201	╔	217	┘	233	Θ	249	·
137	ë	153	Ö	169	_	185	╣	202	╩	218	┌	234	Ω	250	·
138	è	154	Ü	170	¬	186	║	203	╦	219	▉	235	δ	251	√
139	ï	156	£	171	½	187	╗	204	╠	220	▄	236	∞	252	_
140	î	157	¥	172	¼	188	╝	205	═	221	▌	237	φ	253	²
141	ì	158	_	173	¡	189	╜	206	╬	222	▐	238	ε	254	■
142	Ä	159	ƒ	174	«	190	╛	207	╧	223	▀	239	∩	255	
143	Å	192	└	175	»	191	┐	208	╨	224	α	240	≡		

Variáveis - Regras

Para se definir os nomes ou identificadores

- **SEMPRE** começar com uma letra
  - Um número nunca pode ser o primeiro caracter
  - Ex: a, contador, entrada1, valor_1
- Podem conter letras e números e subscrito (_)
  - MAIÚSCULAS são diferentes de minúsculas
  - Ex: valorA, delta, Delta, _dado_1
- **NÃO** podem conter os caracteres:

  { } [ ] ( ) + - * = / \ ; . , @ # ~ ” ! `% $ £ & < > ' | €

  letras acentuadas e ç (Ç)

Palavras Reservadas

Algumas palavras possuem um significado nas linguagens e não podem ser utilizadas para nomes de variáveis. Na linguagem C, **NÃO** podemos usar:

```
auto      double    int       struct    break
enum      register  typedef   char      extern
return    union     const     float     short
unsigned  continue  for       signed    void
default   goto      sizeof    volatile  do
if        static    while
```


Exemplo usando variáveis

```
#include <stdio.h>
int main(int argc, char **argv) {
    int a, b;
    a = 10;
    b = 5;
    printf("a = %d, b = %d, a + b = %d\n", a, b, a + b);
}
```


Exercício

Retornando ao exercício de cálculo da média, escreva um programa em C que define 3 variáveis e atribui os valores 5, 7, 4 a cada uma respectivamente. Calcule a média destes números.

Operador de divisão: /

```
#include <stdio.h>
int main(int argc, char **argv) {
}
```


Exercício

```
#include <stdio.h>
int main(int argc, char **argv) {
    int a, b, c;
    float media, soma;

    a = 5;  b=7;  c=4;
    soma = a + b + c;
    media =  soma / 3;
}
```

Constantes

- Armazenam valores fixos
- São definidas utilizando #define antes do corpo do programa (int main...)
  - Por padrão, devem ser definidas em Maiúsculas
- Possuem os mesmos tipos das variáveis mais um tipo String (seqüência de caracteres)

Constantes

Exemplos:

- Ponto flutuante: **float**, **double**, **unsigned float** e **unsigned double**

  #define PI 3.1415
- Inteiro: **int**, **long int**, **shot int**, **unsigned int**, **unsigned long int**, **unsigned shot int**

  #define NULO 0
- Cadeia de caracteres: string

  #define AUTOR "Programa de Fulano de Tal."

Variáveis <-> Constantes

Quais as principais diferenças entre variáveis e constantes?

Porque usar uma ou outra?

Variáveis - Exemplos

Variáveis	Constantes
• Podem ter seus valores alterados	• Os valores são imutáveis
• O tipo precisa ser definido	• O tipo é definido automaticamente
• Existem durante a execução do programa	• São substituídas por seus valores pelo compilador
a = PI;	a = 3.1415;

Expressões Numéricas

Expressão simples

<operando_1> <operador> <operando_2>

<operando_1> é uma variável ou constante

<operando_2> é uma variável ou constante

<operador> (operadores matemáticos):

=	Atribuição (operando_1 deve ser variável)
+	Adição
-	Subtração
*	Multiplicação
/	Divisão
%	Resto da divisão inteira (entre 2 inteiros)

Expressões Simples

a = 10;	Atribuição de valor a variável
a + 15;	Adição entre variável e valor constante
raio * PI;	Multiplicação entre variável e constante
b / 2;	Divisão entre valor e constante
b – a;	Subtração entre variável e variável
a % 2;	Resto da divisão inteira de a por 2

Expressões Numéricas

Expressão composta

<expressão_1> <operador> <expressão_2>

<expressão_1> é qualquer expressão

<expressão_2> é qualquer expressão

Exemplos:

```
a + b + 5;
2 * PI * raio;
b * a / 10;
conta = raio + PI + b - a;
```


Precedência de Operadores

Existe uma ordem para o cálculo dos operadores da expressão quando o programa for executado.

- Ordem dos operadores na linguagem C

* / %	Na ordem que aparecem na expressão
+ -	Na ordem que aparecem na expressão, mas com precedência menor que os anteriores

- A ordem pode ser alterada
  - Utilize as expressões entre parênteses
    - Número de parênteses abertos = fechados!

Precedências - Exemplos

Qual o resultado das expressões abaixo?

10 + 4 / 2;	retorna 12
10 / 4 + 2;	retorna 4 (Divisão inteira)
(10 + 4)/2;	retorna 7
5 + 10 % 3;	retorna 6
(5 + 10) % 3;	retorna 0
5 * 10 % 3;	retorna 2
5 * (10 % 3);	retorna 5

Dois Operadores Especiais

Incremento (++) e Decremento (--)

- Servem para incrementar (adicionar 1) ou decrementar (subtrair 1) ao valor da variável, e atribuir à ela o resultado final.

  a++; é a mesma coisa que a = a + 1;

- Funciona também como uma expressão, mas a posição dos operadores define o que acontecerá primeiro

  Significa que a++ é diferente de ++a, mas o valor final de a será incrementado em 1

- Nas expressões, ++ e -- têm maior precedência

Incremento à Direita

Operador à direita

```
#include <stdio.h>

int main (int argc, char **argv) {
   int a = 10;
   printf ("%d\n", a++);
   printf ("%d\n", a);
}
```

Imprime como resultado 10 e depois 11

Incremento à Esquerda

Operador à esquerda

```
#include <stdio.h>

int main (int argc, char **argv) {
   int a = 10;
   printf ("%d\n", ++a);
   printf ("%d\n", a);
}
```

Imprime como resultado 11 e depois 11

Atribuições Simplificadas

Comando	Exemplo	Corresponde a:
+=	a += b;	a = a + b;
-=	a -= b;	a = a - b;
*=	a *= b;	a = a * b;
/=	a /= b;	a = a / b;
%=	a %= b;	a = a % b;