

PROGRAMAÇÃO EM LINGUAGEM ESTRUTURADA: LINGUAGEM C

Introdução à Computação I

Estrutura Sequencial em Algoritmos

Bloco Principal

```
ALGORITMO  
    DECLARE  
    bloco de comandos  
FIM_ALGORITMO
```

Declaração de Variáveis

```
DECLARE  
X NUMÉRICO  
Y,Z LITERAL  
TESTE LÓGICO
```



Variável

- ❑ Um algoritmo e, posteriormente, um programa, recebem dados que precisam ser armazenados no computador para serem utilizados no processamento
- ❑ Esse armazenamento é feito na memória
- ❑ Uma variável representa uma posição de memória
- ❑ Toda variável possui nome e tipo e seu conteúdo pode variar ao longo do tempo, durante a execução de um programa
- ❑ Embora uma variável possa assumir diferentes valores, ela só pode armazenar um valor a cada instante

Variável

- Todo computador possui uma tabela de alocação que contém o nome da variável, seu tipo (para saber quantos bytes ocupará) e seu endereço inicial de armazenamento
- Quando queremos buscar algum dado na memória, basta sabermos o nome da variável que o computador, por meio da tabela de alocação, busca automaticamente

Tipos de Dados

- Tipos de dados mais utilizados:
 - ▣ Numéricos
 - Inteiros
 - -23, 98, 0, -2, 2, etc.
 - Reais
 - 23.45, -247.5, etc.
 - ▣ Lógicos
 - Dois possíveis valores: verdadeiro e falso
 - ▣ Literais ou Caracteres
 - “aluno”, “1234”, “@ internet”, ‘y’, ‘n’, etc.

Formação de Identificadores

- Os identificadores são os nomes das variáveis, dos programas, das constantes, das rotinas, das unidades, etc.
- Regras de formação:
 - ▣ Os caracteres que você pode utilizar são: os números, as letras maiúsculas, as letras minúsculas e o caractere sublinhado
 - ▣ O primeiro caractere deve ser sempre uma letra ou o caractere sublinhado
 - ▣ Não são permitidos espaços em branco e caracteres especiais (@, \$, +, -, %, !, etc.)
 - ▣ Não podemos usar as palavras reservadas nos identificadores, ou seja, palavras que pertençam a uma linguagem de programação

CUIDADO!!! A linguagem C é Case Sensitive

Formação de Identificadores

- Exemplos de identificadores válidos:
 - ▣ A, a, nota, X5, nota_1, etc.
- Exemplos de identificadores inválidos:
 - ▣ 5b, e 1 2, x-y, nota (2), case, etc.

Estrutura Sequencial em C

- ❑ Um programa C consiste em uma ou várias funções, onde uma das quais precisa ser denominada **main**
- ❑ O programa sempre começa executando a função **main**
- ❑ Definições de funções adicionais podem preceder ou suceder a função **main**

Estrutura Sequencial em C

Bloco Principal em C

```
#include <nome_da_biblioteca>
void main(void)
{
    declarações locais;
    bloco de comandos;
}
```

Cada instrução se encerra com um ; que faz parte do comando

Bibliotecas são arquivos contendo várias funções que podem ser incorporadas aos programas

A diretiva de pré-processamento #include faz com que o texto contido na biblioteca especificada seja inserido no programa

Estrutura Sequencial em C

□ Diretivas de pré-processamento (#)

▣ Inclusão de arquivos

- `#include <math.h>`
 - Editor de ligação (linguagem de máquina)
- `#include "sistemas.c"`
 - Pré-processador (linguagem de programação utilizada)

▣ Tradução de constantes simbólicas – diretiva `define`

- `#define <nome> <valor>`
 - `#define pi 3.14159`

Nome	Descrição das funções
stdio.h	Funções de entrada e saída (I/O)
string.h	Funções de tratamento de strings
math.h	Funções matemáticas
ctype.h	Funções de teste e tratamento de caracteres
stdlib.h	Funções de uso genérico
conio.h	Funções para controle da tela

Declaração de Variáveis em C

tipo nome_variavel;

Exemplos:

```
float x;  
float y,z;  
int aux=0;  
char sexo;
```

Observação:

1. A linguagem C não possui o tipo de dado lógico, pois considera verdadeiro qualquer valor diferente de zero
2. O tamanho máximo significativo para uma variável é de 31 posições

Modificadores: **short, long, unsigned**

TIPO	FAIXA DE VALORES	TAMANHO (aproximado)	
char	-128 a 127	8 bits	1 byte
unsigned char	0 a 255	8 bits	1 byte
int	-32.768 a 32.767	16 bits	2 bytes
unsigned int	0 a 65.535	16 bits	2 bytes
short int	-32.768 a 32.767	16 bits	2 bytes
long	-2.147.483.648 a 2.147.483.647	32 bits	4 bytes
unsigned long	0 a 4.294.967.295	32 bits	4 bytes
float	3.4×10^{-38} a 3.4×10^{38}	32 bits	4 bytes
double	1.7×10^{-308} a 1.7×10^{308}	64 bits	8 bytes
long double	3.4×10^{-4932} a 1.1×10^{4932}	80 bits	10 bytes

É importante ressaltar que, de acordo com o processador ou compilador C/C++ utilizado, o tamanho e a faixa de valores podem variar. A faixa apresentada está de acordo com o padrão ANSI e é considerada mínima.

Constantes

- Constantes são endereços de memória que guardam valores fixos, ou seja, não se alteram ao longo da execução de um programa
 - ▣ `#define <nome> <valor>`
 - ▣ `const <tipo> <nome> = <valor>`
 - `const float pi = 3.14159`

Comando de Atribuição em C

- Utilizado para conceber valores ou operações a variáveis, sendo representado por =
 - ▣ Em pseudocódigo a atribuição é expressa por ←

Portugol

```
x ← 4  
x ← x + 2  
y ← "aula"  
teste ← falso
```

C

```
x = 4;  
x = x + 2;  
Y = 2.5;  
sexo = 'F';
```

Observação: em C cada comando é finalizado com o sinal de ponto-e-vírgula (;)

Casting

Conversão de Tipos

□ Conversão Implícita (Automática)

- Quando dois ou mais operandos, de diferentes tipos, se encontram em uma mesma expressão, o conteúdo da variável de menor tamanho é convertido ao tipo de variável de maior tamanho
- O resultado da expressão é novamente convertido para o tipo da variável à esquerda do operador de atribuição

```
int Ndec=3;  
int NPoints=1 000;  
float Result;  
Result = Ndec/NPoints;
```

Casting

Conversão de Tipos

□ Conversão Explícita

- O operador de casting consiste em escrever o nome do tipo desejado e, em seguida, o valor ou a expressão a ser avaliada
- O resultado é a expressão convertida para o tipo especificado

```
int Ndec=3;  
int NPoints=1 000;  
float Result;  
Result = (float) Ndec/NPoints;
```

Casting

Conversão de Tipos

Exemplo

```
#include <stdio.h>

void main () {
    float x, y, z;
    x = 20 / 7;
    y = (float)20 / 7;
    z = (int)(2.5 * 4.3);
    printf ("x = %g; y = %g; z = %g", x, y, z);
}
```

Figura 3.14 Programa usando conversão de tipos.

O resultado desse programa é

x = 2; y = 2.85714; z = 10

Comentários

- Muitos programas, devido à sua importância, merecem estar bem documentados, para serem usados, corrigidos, aperfeiçoados pelo próprio programador ou por outras pessoas, em momentos próximos de sua confecção ou em datas bem variadas
- Comentários em C: `/* ... */` e `//`

Comando de Entrada em C

- O comando de entrada é utilizado para receber dados digitados pelo usuário
- Os dados recebidos são armazenados em variáveis

Portugol

```
LEIA X  
LEIA Y
```

C

```
#include <stdio.h>  
  
scanf(<string de controle>, [&]<lista de variáveis>);  
  
scanf("formato",&variável);
```

```
<conio.h>  
getch()  
getche()
```

Comando de Entrada em C

- ❑ `scanf("%d",&x);`
- ❑ `scanf("%d%f",&x,&y);`

Tabela 6.2 Tabela de formatos de leitura da Linguagem C

Formato	Tipos de conversões
<code>%c</code>	Caractere em código ASCII binário
<code>%d, %i</code>	Inteiro decimal em complemento de 2 de 2 bytes
<code>%ld</code>	Inteiro decimal em complemento de 2 de 4 bytes
<code>%o</code>	Inteiro octal em complemento de 2 de 2 bytes
<code>%x, %X</code>	Inteiro hexadecimal em complemento de 2 de 2 bytes
<code>%f, %g, %G, %e, %E</code>	Real decimal fracionário ou exponencial em ponto flutuante de 4 bytes
<code>%lf, %le</code>	Real decimal fracionário ou exponencial em ponto flutuante de 8 bytes
<code>%s</code>	Cadeia de caracteres em código ASCII binário, encerrado pelo código do caractere <code>'\0'</code>

Comando de Saída em C

- O comando de saída é utilizado para mostrar dados na tela ou na impressora

Portugol

```
ESCREVA X  
ESCREVA Y  
ESCREVA "Conteúdo de Y = ", Y  
ESCREVA "Olá Mundo"
```

C

```
#include <stdio.h>  
  
printf(<string de controle>, <lista de variáveis>);  
  
printf("formato",variável);  
printf("texto+formato",variável);  
printf("texto");
```

Comando de Saída em C

Tabela 3.2 Representação de alguns caracteres especiais e de controle na Linguagem C

Caractere	Representação em C	Significado
nl	'\n'	Iniciar nova linha (<i>new line</i>)
ht	'\t'	Tabulação horizontal
cr	'\r'	Voltar ao início da mesma linha (<i>carriage return</i>)
bs	'\b'	Voltar um espaço (<i>back space</i>)
bel	'\a'	Tocar a campainha
nul	'\0'	Caractere nulo
'	'\''	Apóstrofo
"	'\"'	Aspas
\	'\\'	Barra invertida

Tabela 6.1 Tabela de formatos de escrita da Linguagem C

Formato	Tipos das entidades escritas
%d, %i	Inteiro decimal de 2 bytes
%ld	Inteiro decimal de 4 bytes
%u	Inteiro decimal de 2 bytes, sem sinal
%o	Inteiro octal, sem sinal
%x, %X	Inteiro hexadecimal, sem sinal
%c	Caractere em ASCII
%s	Cadeia de caracteres em ASCII
%f	Real com 6 casas fracionárias
%g	Real com no máximo 6 casas fracionárias
%e, %E	Real decimal em notação exponencial
%%	Escreve apenas um caractere '%'

Comando de Saída em C

- `printf("Olá Mundo!!!");`
- `printf("%d",x);`
- `printf("O valor de x é: %d",x);`
- Tamanho do Campo
 - ▣ `printf("O valor de x é: %3d",10);` R: `_10`
 - ▣ `printf("O valor de x é: %-3d",10);` R: `10_`
 - ▣ `printf("O valor de x é: %03d",10);` R: `010`
 - ▣ `printf("O valor de x é: %.2f",4.78);` R: `4.78`
 - ▣ `printf("O valor de x é: %.1f",4.78);` R: `4.8`
 - ▣ `printf("Impressão do A: %d %c %x %o \n", 'A', 'A', 'A', 'A');` R: `65 A 41 101`
 - ▣ `printf("Letras A: %c %c %c %c \n", 'A', 65, 0x41, 0101);` R: `A A A A`

Operadores

Operadores Aritméticos

Operador	Descrição
=	atribuição
+	adição
-	subtração
*	multiplicação
/	divisão
%	resto da divisão
++	incremento
--	decremento

Operadores Combinados

Normal	Simplificada
$a=a+b;$	$a+=b;$
$a=a-b;$	$a-=b;$
$a=a*b;$	$a*=b;$
$a=a/b;$	$a/=b;$
$a=a\%b;$	$a\%=b;$

Operadores Lógicos

Operador	Descrição
&&	and lógico
	or lógico
!	not lógico

Operadores Relacionais

Operador	Descrição
>	maior que
>=	maior ou igual à
<	menor que
<=	menor ou igual à
==	igual à
!=	diferente de

Operadores

Tabela 3.6 Operadores da Linguagem C, sua precedência e associatividade

Precedência	Operadores	Associatividade
1	() [] . -> ++(posfixa) --(posfixa)	À esquerda
2	++(prefixa) -(prefixa) ! ~ sizeof (tipo) +(unário) -(unário) &(endereço) *(unário)	À direita
3	* / % (multiplicativos)	À esquerda
4	+ - (aditivos)	À esquerda
5	<< >> (entrada e saída)	À esquerda
6	< <= > >= (relacionais)	À esquerda
7	== != (de igualdade)	À esquerda
8	&(bit a bit)	À esquerda
9	^(bit a bit)	À esquerda
10	(bit a bit)	À esquerda
11	&& (lógico multiplicativo)	À esquerda
12	(lógico aditivo)	À esquerda
13	?: (expressão condicional)	À direita
14	= += -= *= /= %= >>= <<= &= ^= = (de atribuição)	À direita
15	, (vírgula)	À esquerda



Exercícios

Exercícios

- ❑ Faça um algoritmo que receba três notas e seus respectivos pesos, calcule e mostre a média ponderada. [Algoritmo → C]
- ❑ Faça um algoritmo que receba o salário de um funcionário, calcule e mostre o novo salário, sabendo-se que este sofreu um aumento de 25%. [Algoritmo → C]

Exercícios

□ Sabe-se que:

1 pé = 12 polegadas

1 jarda = 3 pés

1 milha = 1760 jardas

Faça um programa que receba uma medida em pés, faça as conversões a seguir e mostre os resultados.

(a) polegadas;

(b) jardas;

(c) milhas.

[Algoritmo → C]



Referências

Referências

- ASCENCIO, A. F. G.; CAMPOS, E. A. V. Fundamentos da programação de computadores: Algoritmos, Pascal, C/C++ e Java. Prentice Hall, 2007.
- Mokarzel, F. C. Introdução à ciência da computação. Elsevier, 2008.
- MEDINA, M.; FERTIG, C. Algoritmos e Programação: Teoria e Prática. Novatec, 2005.