



Algoritmos 1

Introdução à Linguagem C



Um pouco de história...

- Bell Labs;
- Desenvolvimento de tecnologias:
 - Fax;
 - Televisão;
 - Unix;
 - Linguagem C;



Bell Telephone Laboratories

Um pouco de história...

- Criador da Linguagem C;
- Contribuição no UNIX;
- Vencedor do prêmio Turing;



Dennis Ritchie



Linguagem de Programação

Linguagem C

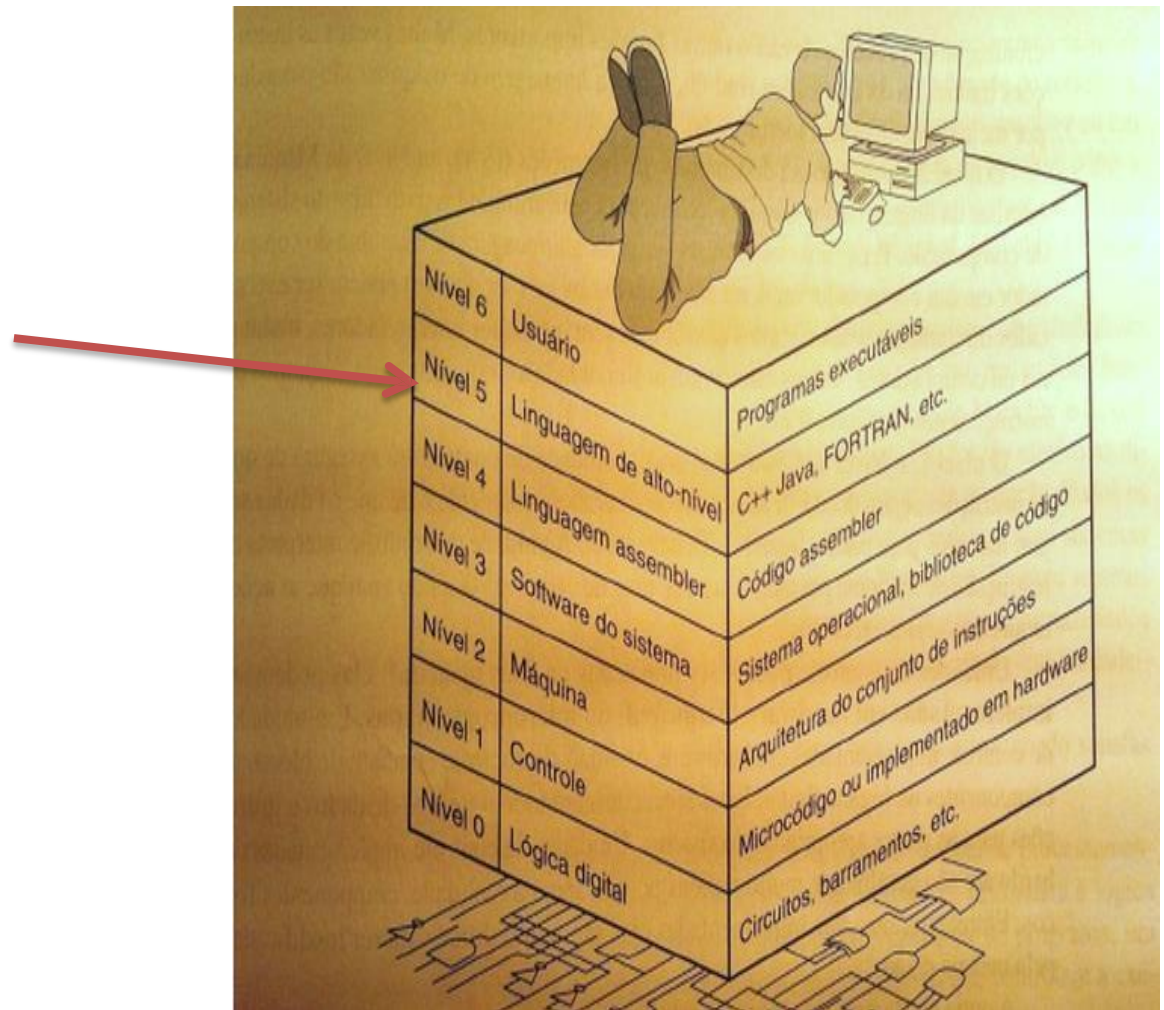
- Derivada da Linguagem B;
- Desenvolvida por programadores e para programadores;
- Alto Nível: Nível de abstração elevado, (mais próximo da linguagem humana);



Linguagem de Programação

Linguagem C

Hierarquia de níveis de
Arquitetura
de Computadores





Linguagem de Programação

Linguagem C

- Vantagens:
 - Flexibilidade: diversos tipos de aplicação, de jogos eletrônicos a controladores de satélites (propósitos gerais – *general purpose*);
 - Portabilidade, os programas codificados em C podem ser executados em diversas plataformas, praticamente, sem nenhuma alteração;
 - Eficiência proporciona alta velocidade de execução e economia de memória;





Linguagem de Programação

- O compilador da linguagem gera códigos mais enxutos e velozes;
- Precursora de linguagens como: C#, Java, e PHP;





Etapas de desenvolvimento

Edição do Código Fonte

Ambiente de Desenvolvimento Integrado



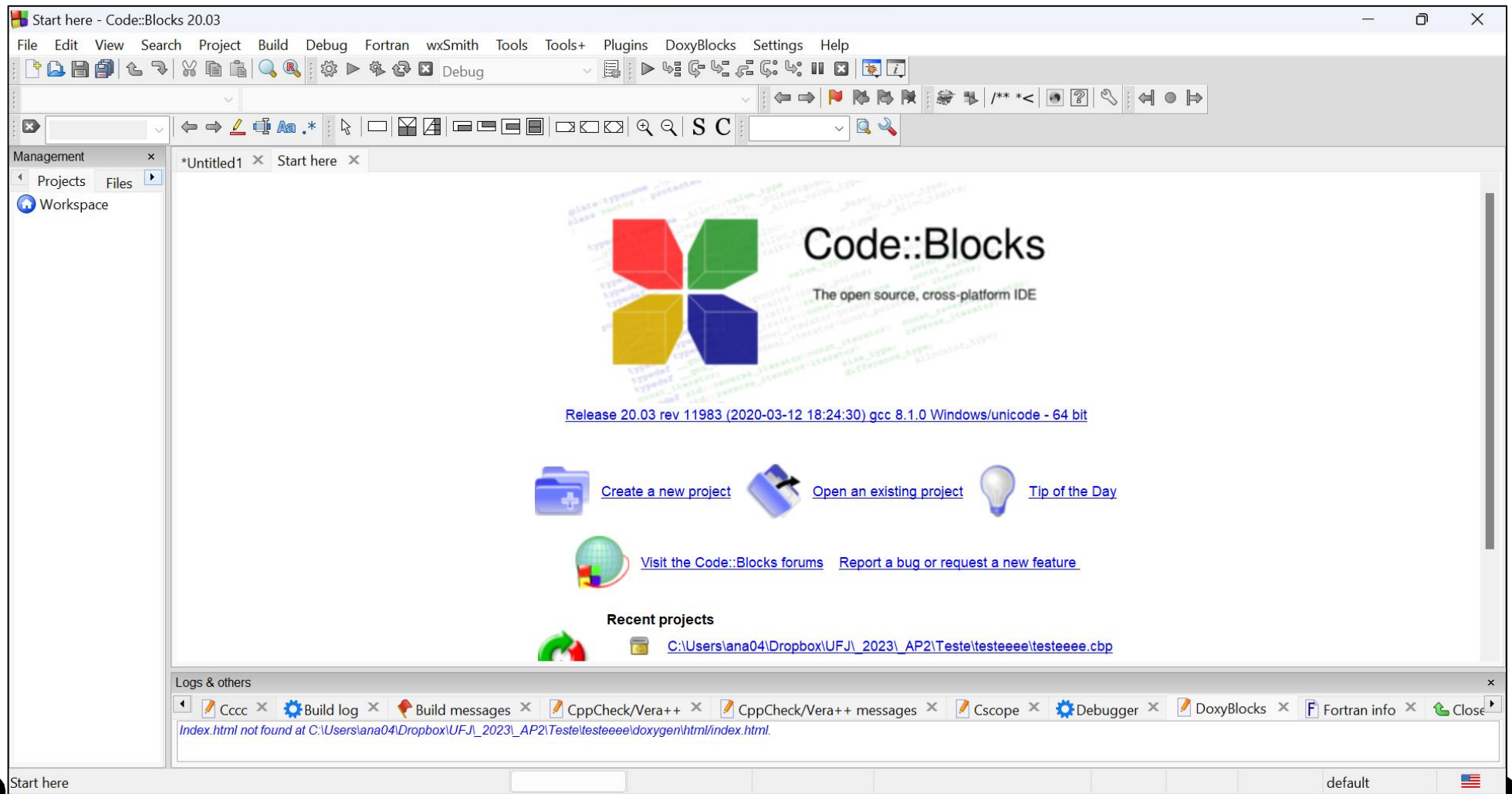


Ambiente de Desenvolvimento Integrado

- Download: www.codeblocks.org;
- Windows (Podem ocorrer problemas)
- Possível baixar apenas com a IDE;
- Possível baixar com o compilador do GCC do MinGW:



Ambiente de Desenvolvimento Integrado



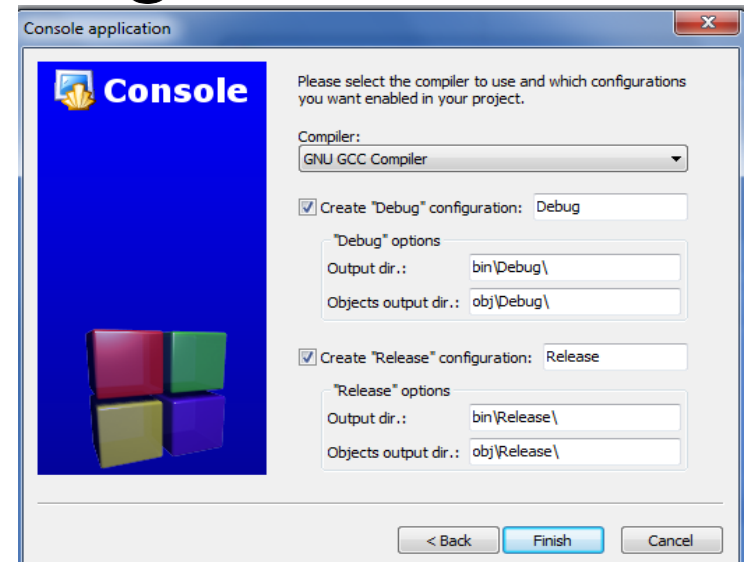
Ambiente de Desenvolvimento Integrado

- Criando um novo projeto;



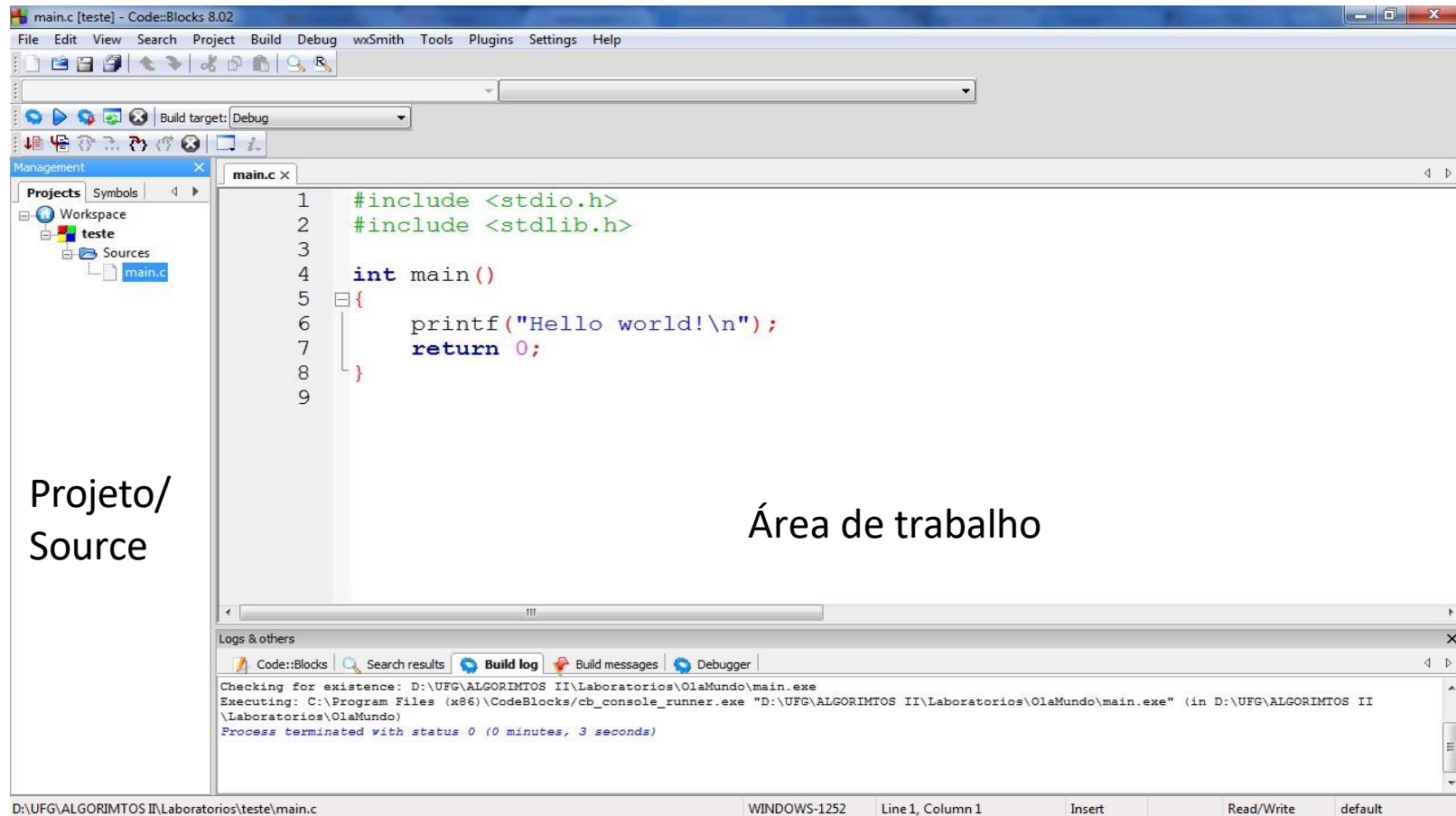
Ambiente de Desenvolvimento Integrado

“Project title”,
“Folder to create project in”



Configurações do
compilador que
poderão ser
modificadas

Ambiente de Desenvolvimento Integrado



Estrutura de um programa em C

```
#include <stdio.h>           //Biblioteca de Entrada e Saída
#include <stdlib.h>          //Biblioteca padrão

int main()
{
    printf("Hello world!\n"); // Escreve uma mensagem na tela
    return 0;                // Retorna zero
}
```

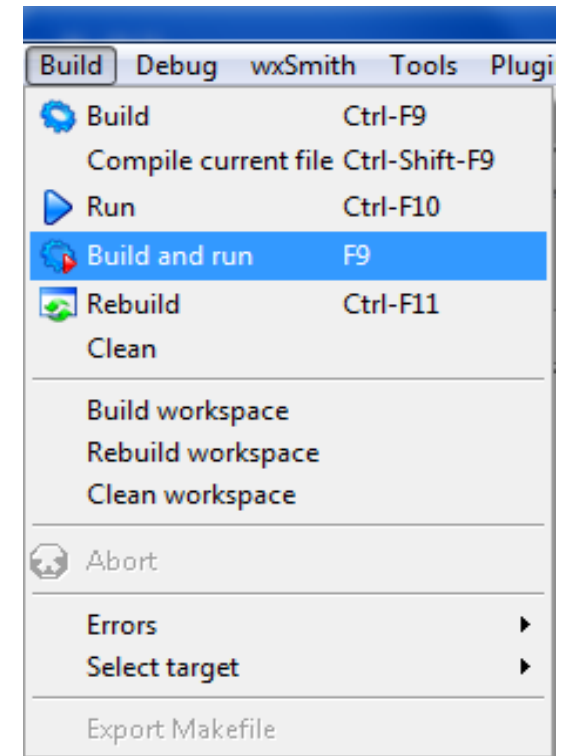
// Início do programa
// Fim do programa

Ambiente de Desenvolvimento Integrado

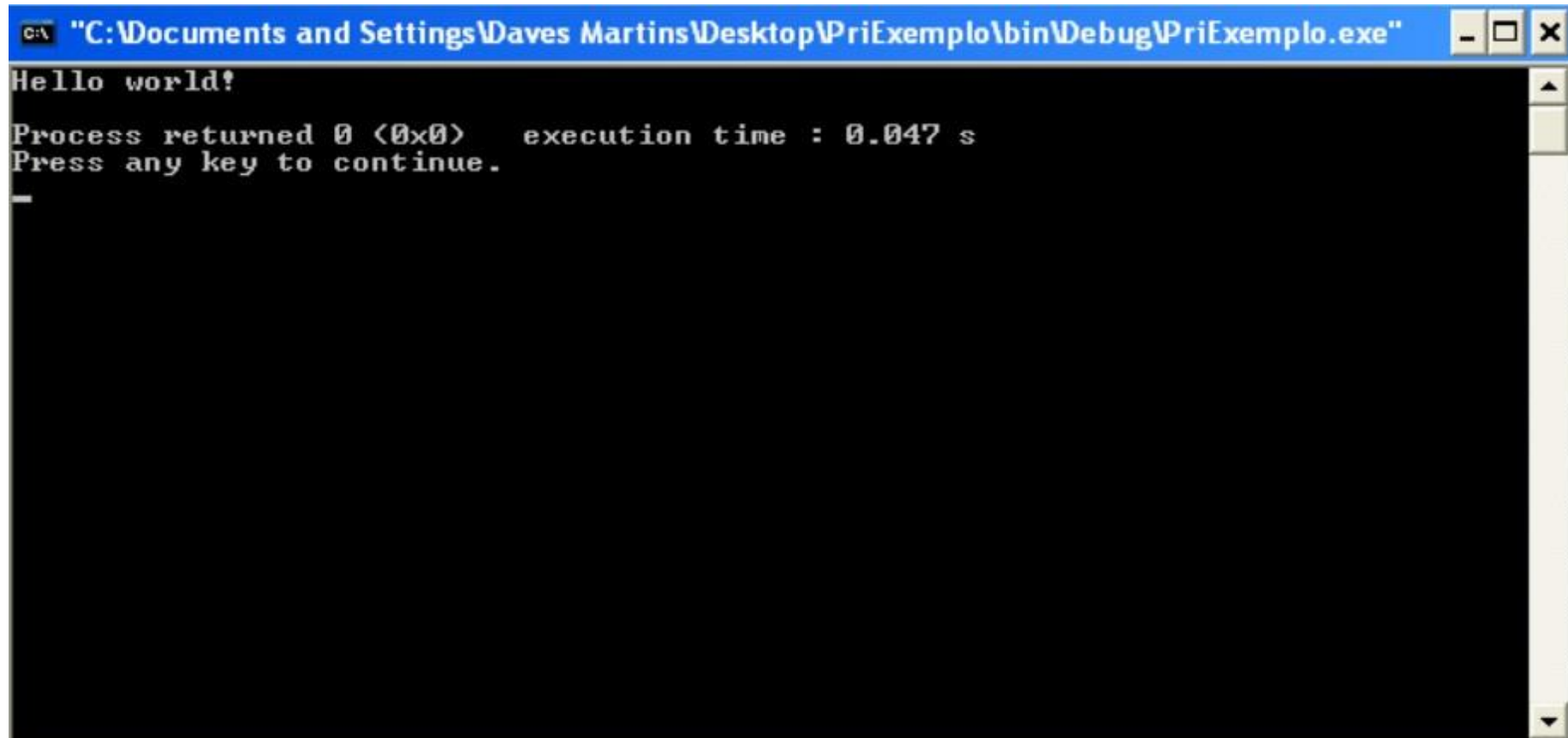
Build (Ctrl+F9): Compila todos os arquivos do projeto e faz o processo de linkagem com tudo que é necessário para gerar o executável do seu programa;

Run (Ctrl + F10): Executa a última versão do código compilado;

Build and run (F9): Além de gerar o executável, executa o programa gerado.



Ambiente de Desenvolvimento Integrado



A screenshot of a Windows command prompt window. The title bar at the top reads "C:\ "C:\Documents and Settings\Daves Martins\Desktop\PriExemplo\bin\Debug\PriExemplo.exe". The window has standard Windows window controls (minimize, maximize, close) on the right. The command prompt area is black with white text. It displays "Hello world!" on the first line. The second line shows "Process returned 0 (0x0) execution time : 0.047 s". The third line says "Press any key to continue." followed by a cursor on the fourth line.

```
C:\ "C:\Documents and Settings\Daves Martins\Desktop\PriExemplo\bin\Debug\PriExemplo.exe"
Hello world!
Process returned 0 (0x0) execution time : 0.047 s
Press any key to continue.
_
```




Geração de Código

- A geração de programa executável a partir do programa fonte obedece uma sequência de operações:
 1. Editor (Módulo fonte em C). Ex.:First.c;
 2. Compilador (Gera o arquivo objeto). Ex.: first.o;
 3. Lincador (Gera o executável). Ex.: first.exe;





Etapas de desenvolvimento

Edição do Código Fonte

- Para cada comando serão apresentados:

Sintaxe: Formato geral do comando (padrão);

Semântica: Significado da ação;



Etapas de desenvolvimento

- Programa mínimo em C;
- Deve conter a função *main* (principal), que será sempre a primeira função a ser executada

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

```
//Primeiro código em C
```

```
void main( )
```

```
{
```

```
    int a;
```

bloco

```
    printf("Hello world");
```

```
}
```


Executa comando por comando, de cima para baixo e sai



Etapas de desenvolvimento

Edição do Código Fonte

Sintaxe:

- Diretivas de Compilação
 - Comentários
 - Definição de tipos
 - Entrada e Saída
 - Operadores
- 



Etapas de desenvolvimento

Edição do Código Fonte – Diretivas de Compilação

- Comandos que instruem o compilador a realizar determinadas tarefas antes de iniciar a compilação de todos ou parte do programa;
- Conhecidas como diretivas de pré-processamento;
- Todas as diretivas iniciam com `#include` e os arquivos tem uma extensão `.h`

Ex.: *`#include <stdio.h>`*





Etapas de desenvolvimento

Edição do Código Fonte – Diretivas de Compilação

- Acentuação de caracteres em C;

<locale.h>: Adaptação às características de um determinado idioma;

```
#include <locale.h>           //necessário para usar setlocale  
setlocale(LC_ALL, " ");       //linguagem do SO  
setlocale(LC_ALL, "Portuguese"); //Português
```





Etapas de desenvolvimento

Edição do Código Fonte – Comentários

- Os comentários podem ser escritos:

`/* */` O que estiver escrito dentro desse intervalo, será comentário para o compilador;

`//` Comentário em apenas uma linha.

- Em C existe a necessidade de um programa principal `main()`;
- A cada comando (não função) é finalizado com um ponto e vírgula `(;)`





Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos

Variável:

- Declarada no início do programa;
Sintaxe: <tipo> nome;
- Nome que corresponde a uma posição da memória e que contém o seu valor;
- Em tempo de execução, o nome da variável permanece sempre o mesmo e seu valor pode ser modificado;





Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos

- Nome de Variável: Criado pelo programador e deve ser iniciado por uma letra;
- Deve ter o nome sugestivo;
- Exemplos:
 - ✓ Certos: nome, telefone, salario_func, x1;
 - ✓ Errado: 1no, sal/hora, _nome





Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos

- Tipos de Dados Básicos da variáveis;
- **inteiro, real, caracter, vazio.**
- (Em C: int, float/double, char, void);





Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos – char

- ASCII - (*American Standard Code for Information Interchange*);
- Código comum, possibilitando a comunicação entre os computadores;
- Divisão: Tabela de caracteres de controle (funções ou equipamentos) – [0-31] decimal; Tabela com caracteres (ASCII Normal) – [32-127]; Tabela ASCII Estendida [128 – 255];



Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos - char

Decimal	Binário	Hex	Referência
0	00000000	00	Null - NUL
1	00000001	01	Start of Heading - SOH
2	00000010	02	Start of Text - STX
3	00000011	03	End of Text - ETX
4	00000100	04	End of Transmission - EOT
5	00000101	05	Enquiry - ENQ
6	00000110	06	Acknowledge - ACK
7	00000111	07	Bell, rings terminal bell - BEL
8	00001000	08	BackSpace - BS
9	00001001	09	Horizontal Tab - HT
10	00001010	0A	Line Feed - LF
11	00001011	0B	Vertical Tab - VT
12	00001100	0C	Form Feed - FF
13	00001101	0D	Enter - CR
14	00001110	0E	Shift-Out - SO
15	00001111	0F	Shift-In - SI
16	00010000	10	Data Link Escape - DLE
17	00010001	11	Device Control 1 - D1
18	00010010	12	Device Control 2 - D2
19	00010011	13	Device Control 3 - D3
20	00010100	14	Device Control 4 - D4

Decimal	Binário	Hex	Referência
21	00010101	15	Negative Acknowledge - NAK
22	00010110	16	Synchronous idle - SYN
23	00010111	17	End Transmission Block - ETB
24	00011000	18	Cancel line - CAN
25	00011001	19	End of Medium - EM
26	00011010	1A	Substitute - SUB
27	00011011	1B	Escape - ESC
28	00011100	1C	File Separator - FS
29	00011101	1D	Group Separator - GS
30	00011110	1E	Record Separator - RS
31	00011111	1F	Unit Separator - US
32	00100000	20	Space - SPC
33	00100001	21	!
34	00100010	22	"
35	00100011	23	#
36	00100100	24	\$
37	00100101	25	%
38	00100110	26	&
39	00100111	27	'
40	00101000	28	(

Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos - char

41	00101001	29)
42	00101010	2A	*
43	00101011	2B	+
44	00101100	2C	,
45	00101101	2D	-
46	00101110	2E	.
47	00101111	2F	/
48	00110000	30	0
49	00110001	31	1
50	00110010	32	2
51	00110011	33	3
52	00110100	34	4
53	00110101	35	5
54	00110110	36	6
55	00110111	37	7
56	00111000	38	8
57	00111001	39	9
58	00111010	3A	:
59	00111011	3B	;
60	00111100	3C	<

61	00111101	3D	=
62	00111110	3E	>
63	00111111	3F	?
64	01000000	40	@
65	01000001	41	A
66	01000010	42	B
67	01000011	43	C
68	01000100	44	D
69	01000101	45	E
70	01000110	46	F
71	01000111	47	G
72	01001000	48	H
73	01001001	49	I
74	01001010	4A	J
75	01001011	4B	K
76	01001100	4C	L
77	01001101	4D	M
78	01001110	4E	N
79	01001111	4F	O
80	01010000	50	P

Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos - char

81	01010001	51	Q
82	01010010	52	R
83	01010011	53	S
84	01010100	54	T
85	01010101	55	U
86	01010110	56	V
87	01010111	57	W
88	01011000	58	X
89	01011001	59	Y
90	01011010	5A	Z
91	01011011	5B	[
92	01011100	5C	\
93	01011101	5D]
94	01011110	5E	^
95	01011111	5F	_
96	01100000	60	`
97	01100001	61	a
98	01100010	62	b
99	01100011	63	c
100	01100100	64	d

101	01100101	65	e
102	01100110	66	f
103	01100111	67	g
104	01101000	68	h
105	01101001	69	i
106	01101010	6A	j
107	01101011	6B	k
108	01101100	6C	l
109	01101101	6D	m
110	01101110	6E	n
111	01101111	6F	o
112	01110000	70	p
113	01110001	71	q
114	01110010	72	r
115	01110011	73	s
116	01110100	74	t
117	01110101	75	u
118	01110110	76	v
119	01110111	77	w
120	01111000	78	x

Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos - char

121	01111001	79	y
122	01111010	7A	z
123	01111011	7B	{
124	01111100	7C	
125	01111101	7D	}
126	01111110	7E	~
127	01111111	7F	Delete
128	10000000	80	Ç
129	10000001	81	ü
130	10000010	82	é
131	10000011	83	â
132	10000100	84	ä
133	10000101	85	à
134	10000110	86	ã
135	10000111	87	ç
136	10001000	88	ê
137	10001001	89	ë
138	10001010	8A	è
139	10001011	8B	ï
140	10001100	8C	î

141	10001101	8D	ì
142	10001110	8E	Ã
143	10001111	8F	Å
144	10010000	90	É
145	10010001	91	æ
146	10010010	92	Æ
147	10010011	93	ô
148	10010100	94	ö
149	10010101	95	ò
150	10010110	96	û
151	10010111	97	ù
152	10011000	98	ÿ
153	10011001	99	Ö
154	10011010	9A	Û
155	10011011	9B	ø
156	10011100	9C	£
157	10011101	9D	Ø
158	10011110	9E	×
159	10011111	9F	f
160	10100000	A0	á

Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos - char

161	10100001	A1	ù
162	10100010	A2	ó
163	10100011	A3	ú
164	10100100	A4	ñ
165	10100101	A5	Ñ
166	10100110	A6	ª
167	10100111	A7	º
168	10101000	A8	¿
169	10101001	A9	®
170	10101010	AA	¬
171	10101011	AB	½
172	10101100	AC	¼
173	10101101	AD	¡
174	10101110	AE	«
175	10101111	AF	»
176	10110000	B0	▒
177	10110001	B1	▒
178	10110010	B2	▒
179	10110011	B3	
180	10110100	B4	┘

181	10110101	B5	À
182	10110110	B6	Â
183	10110111	B7	Ã
184	10111000	B8	©
185	10111001	B9	⌌
186	10111010	BA	⌌
187	10111011	BB	⌌
188	10111100	BC	⌌
189	10111101	BD	¢
190	10111110	BE	¥
191	10111111	BF	⌌
192	11000000	C0	┐
193	11000001	C1	┐
194	11000010	C2	┐
195	11000011	C3	┐
196	11000100	C4	—
197	11000101	C5	+
198	11000110	C6	ä
199	11000111	C7	Ä
200	11001000	C8	ℓ

Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos - char

201	11001001	C9	Œ
202	11001010	CA	ℒ
203	11001011	CB	Ƨ
204	11001100	CC	℔
205	11001101	CD	=
206	11001110	CE	℔
207	11001111	CF	□
208	11010000	D0	δ
209	11010001	D1	Ð
210	11010010	D2	Ê
211	11010011	D3	Ë
212	11010100	D4	È
213	11010101	D5	ı
214	11010110	D6	Í
215	11010111	D7	Î
216	11011000	D8	Ï
217	11011001	D9	Ɔ
218	11011010	DA	Ɔ

219	11011011	DB	■
220	11011100	DC	■
221	11011101	DD	ı
222	11011110	DE	İ
223	11011111	DF	■
224	11100000	E0	Ó
225	11100001	E1	β
226	11100010	E2	Ô
227	11100011	E3	Ò
228	11100100	E4	ø
229	11100101	E5	Õ
230	11100110	E6	μ
231	11100111	E7	þ
232	11101000	E8	Ɔ
233	11101001	E9	Ú
234	11101010	EA	Û
235	11101011	EB	Ü
236	11101100	EC	ý
237	11101101	ED	Ý
238	11101110	EE	—
239	11101111	EF	,

Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos - char

240	11110000	F0	
241	11110001	F1	±
242	11110010	F2	—
243	11110011	F3	³ / ₄
244	11110100	F4	¶
245	11110101	F5	§
246	11110110	F6	÷
247	11110111	F7	•
248	11111000	F8	°
249	11111001	F9	~
250	11111010	FA	•
251	11111011	FB	1
252	11111100	FC	3
253	11111101	FD	2
254	11111110	FE	■
255	11111111	FF	



Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos – int

- Um número inteiro é um número sem vírgula, que pode ser expresso em diferentes bases:
 - ✓ **Base decimal:** o número inteiro é representado por uma sequência de unidades (de 0 a 9).
 - ✓ **Base hexadecimal:** o número inteiro é representado por uma sequência de unidades (de 0 a 9 ou de A a F (ou de a a f)), que deve começar com 0x ou 0X.
 - ✓ **Base octal:** o número inteiro é representado por uma sequência de unidades (incluindo apenas os dígitos de 0 a 7).





Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos – Números reais – float e double

- Número com vírgula e pode ser representado:
 - ✓ um inteiro decimal : 895
 - ✓ um número com um ponto (ou vírgula, dependendo da configuração da máquina) : 845.32
 - ✓ uma fração: 27/11
 - ✓ um número exponencial, ou seja, um número (possivelmente com vírgula) seguido da letra e (ou E) e de um inteiro correspondente à potência de 10 (assinado ou não, isto é, precedido por um "+" ou um "-"): 2.75e-2; 35.8E+10; .25e-2





Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos – Números reais – float e double

- $\pm M \times B^{+e}$

M é a mantissa (parte fracionária)

B é a base

e é o expoente

- Os números reais são números com vírgula flutuante, isto é, números em que a posição da vírgula não é fixa, e é identificada por uma parte dos seus bits;





Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos

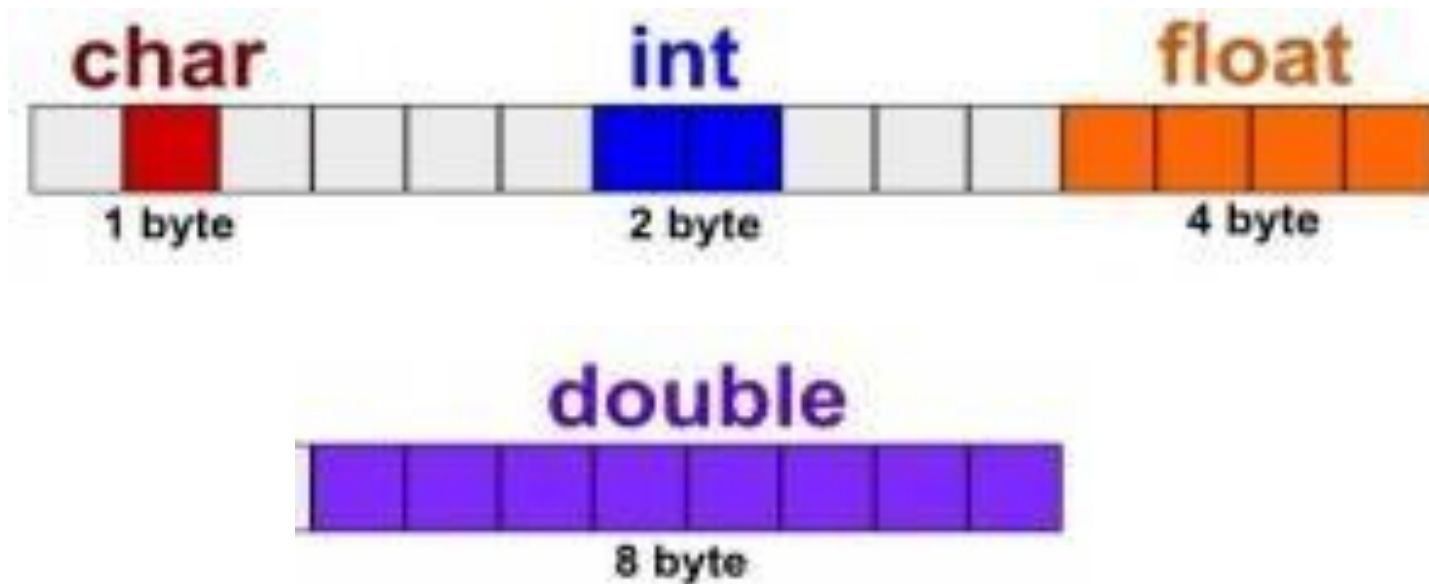
Tipo de Dado	Espaço	Escala
char	1 byte	-128 a +127
int	2 bytes	-32.768 a +32.767
float	4 bytes	-3,4e38 a +3,4e38
double	8 byte	-1,7e308 a +1,7e308
void	Nenhum	Nenhum

Tipos de dados em C



Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos





Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos

Exemplo:

```
char tecla, opcao;  
int x, y, z;  
float media, porcentagem;  
double potencia;
```





Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos

Modificadores de tipo



Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos

signed / unsigned

- Bit mais a esquerda em uma variável do tipo Char ou Int;
- Bit de Sinal;
- Valores positivos ou negativos;



Bit de Sinal: Se bit=0, então o valor é positivo, caso contrário é negativo.



Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos

signed / unsigned

- Unsigned: A memória armazenará apenas valores sem sinal;
- Desnecessário reservar este espaço;



Sem reservar o Bit de Sinal, a capacidade de armazenamento dobra.





Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos

Long

- Faz com que o espaço de memória, reservado para uma variável do tipo int seja duplicado;
- Aumenta a capacidade de armazenamento da variável;





Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos

Short

- Em algumas máquinas, faz com que esse espaço caia para a metade





Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos

Tipo	Num. bits	Intervalo inicial	Intervalo final
char	8	-128	127
unsigned char	8	0	255
signed char	8	-128	127

Modificadores de dados: char



Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos

Tipo	Num. bits	Intervalo inicial	Intervalo final
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

Modificadores de dados: int



Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos

Tipo	Num. bits	Intervalo inicial	Intervalo final
float	32	3,4e-38	3,4e+38
double	64	1,7e-308	1,7e+308
long double	80	3,4e-493	3,4e+493

Modificadores de dados: Reais





Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos

Tipos de dados e Manipulação de variáveis





Etapas de desenvolvimento

Edição do Código Fonte – Entrada e Saída de dados formatada

Leitura

- Similar ao comando `leia(var);`
- Especificadores de formato: quantidade e os tipos de dados;
- Exemplo:

```
scanf("%d %c",&idade,&sexo); //leia(idade, sexo);
```



Etapas de desenvolvimento

Edição do Código Fonte – Entrada e Saída de dados formatada

Especificador	Representa
%c	Único caracter
%i,%o,%d,%x	Número inteiro, octal, decimal e hexadecimal, respectivamente
%u	Número inteiro em base decimal sem sinal
%ld	Número inteiro longo em base decimal
%f, %lf	Número real de precisão simples ou dupla
%s	Cadeia de caracteres (string)

Etapas de desenvolvimento

Edição do Código Fonte – Entrada e Saída de dados formatada

Escrita

- Similar ao comando escreva(“O resultado é “,var);
printf(“formatação”,arg1,arg2....,argn);
- Especificadores de formato: quantidade e os tipos de dados;
- Exemplo:

```
printf(“O resultado é %d e %c”, idade, sexo);
```



Etapas de desenvolvimento

Edição do Código Fonte – Entrada e Saída de dados formatada

Caracter de controle	Efeito
<code>\a</code>	Soa alarme
<code>\n</code>	Salta linha
<code>\t</code>	Tabulação
<code>\"</code>	Exibe uma única aspa
<code>\\</code>	Exibe uma única barra invertida

Os principais caracteres de controle utilizados com a função `printf()`





Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos

Atribuição em VisuAlg '←'
Atribuição em C '='





Etapas de desenvolvimento

Edição do Código Fonte – Definição de Tipos

Operadores Aritméticos






Etapas de desenvolvimento

Edição do Código Fonte – Operadores Aritméticos

Operador	Resultado
+	Soma
-	Subtração
*	Produto
/	Divisão
%	Resto da divisão





Etapas de desenvolvimento

Edição do Código Fonte

Estruturas Condicionais





Etapas de desenvolvimento

Edição do Código Fonte – Estruturas Condicionais

- Não existe um tipo específico para a representação de valores lógicos.
- O valor 0 representa o valor lógico "falso" e 1 representa o valor lógico "verdade".
- Exemplo: `printf("%d %d", 5<6, 5>6);`



Etapas de desenvolvimento

Edição do Código Fonte – Estruturas Condicionais

Operador relacional	Resultado
$x == y$	verdade se x for igual a y
$x != y$	verdade se x for diferente de y
$x < y$	verdade se x for menor que y
$x > y$	verdade se x for maior que y
$x \leq y$	verdade se x for menor ou igual a y
$x \geq y$	verdade se x for maior ou igual a y


Operadores relacionais e valores lógicos



Condicionais

Edição do Código Fonte – Estruturas Condicionais Simples

```
if (condição)
{ //comandos caso condição seja verdadeira
}
else //opcional
{ //comandos caso condição seja falsa
}
```





Condicionais

Edição do Código Fonte – Estruturas Condicionais Simples

A linguagem C possui um operador que proporciona uma forma mais compacta de se representar decisões simples.

Sintaxe: <condição> ? <expressão1> : <expressão2>

Exemplo:

```
x>3?printf("Acertou!"):printf("Errou!");
```





Condicionais Compostas

Edição do Código Fonte – Estruturas Condicionais Compostas

```
if ((condição1&&condição2)||condição3)
{ //comandos caso condição seja verdadeira
}
else
{ //comandos caso condição seja falsa
}
```






Condicionais aninhadas

Em C

```
if (condição){  
    if (condição2){  
    }  
}  
else{  
    if (condição3){  
    }  
}
```






Condicionais: Switch-Case x Escolha Caso

Switch-Case

```
switch(n) {  
    case 1: putchar('A');  
        break;  
    case 3: putchar('B');  
        break;  
    case 4: putchar('C'); break;  
    default: printf('*');break;  
}
```





Laços de Repetição While

While

```
a=1;  
while (a<10)  
{  
    printf("%d\n",a);  
    a++;  
}
```

Enquanto

```
a<-1;  
enquanto (a<10) faça  
    escreval(a);  
    a<-a+1;  
fimenquanto
```





Laços de Repetição do while

Do while

```
a=1;  
do  
{  
    printf("%d\n",a);  
    a++;  
} while (a<10);
```

Repita até

```
a<-1;  
repita  
    escreval(a);  
    a<-a+1;  
ate (a>=10)
```





Laços de Repetição for

for

```
para (i=1;i<10;i++)  
{  
    printf("%d\n",i);  
}
```

Para

```
para i de 1 ate 10 faça  
    escreval(i);  
fimpara
```






Vetor em C

Vetor em C

```
Int v[10],i;  
para (i=0;i<10;i++)  
{  
    scanf("%d",&v[i]);  
}  
para (i=0;i<10;i++)  
{  
    printf("%d",v[i]);  
}
```

Vetor em pseudocódigo

```
V: vetor[1..10] de inteiro  
i: inteiro  
para i de 1 ate 10 faça  
    leia(v[i]);  
fimpara  
  
para 1 de 1 ate 10 faça  
    leia(v[i]);  
fimpara
```



Matriz em C

Matriz em C

```
int v[10][3],i;  
for(i=0;i<10;i++){  
    for (j=0;j<3;j++){  
        scanf("%d",&v[i][j]);  
    }  
}  
for(i=0;i<10;i++){  
    for (j=0;j<3;j++){  
        scanf("%d",&v[i][j]);  
    }  
}
```

Matriz em pseudocódigo

V: vetor[1..10,1..3] de inteiro

l,j: inteiro

para i de 1 ate 10 faça

para j de 1 ate 3 faça

leia(v[i,j]);

fimpara

fimpara

para i de 1 ate 10 faça

para j de 1 ate 3 faça

leia(v[i,j]);

fimpara

fimpara