

C / C++

- Referências online

<http://www.cplusplus.com/>

<http://www.openbookproject.net/thinkcs/cpp/english/>

[http://pt.wikipedia.org/wiki/C_\(linguagem_de_programação\)](http://pt.wikipedia.org/wiki/C_(linguagem_de_programação))

<http://pt.wikipedia.org/wiki/C%2B%2B>

C / C++

- Referências online – não tenha medo de buscar!



YAHOO!

bing

C / C++

- Programação em ambiente Windows
- Ambiente de programação utilizado: Dev C++

<http://www.bloodshed.net/devcpp.html>

- Outros ambientes – GCC e G++

<http://gcc.gnu.org/>

C / C++

- Windows – Visual Studio

<http://msdn.microsoft.com/pt-br/vstudio/default.aspx>

- Linux - KDevelop

<http://www.kdevelop.org/>

- Mac – Xcode

<http://developer.apple.com/tools/xcode/>

Ementa do curso - C

- Introdução a algoritmos
- Estrutura de um programa
- Variáveis e constantes
- Operadores
- Estruturas de controle de fluxo
- Funções

Ementa do curso - C

- Vetores, *strings* e matrizes
- Ponteiros
- Memória dinâmica
- Dados definidos pelo usuário
- Diretivas de compilação
- Entrada e saída de dados / arquivos

Ementa do curso - C++

- Extensões ao C
- Orientação a objeto
- Classes
- Herança
- Polimorfismo
- Sobrecarga de operador
- Exceções

Introdução a algoritmos

- Algoritmos

- Sequência de instruções para resolver um problema;
- Sequência finita;
- Ordenada de maneira lógica;

- Um algoritmo não representa uma solução única; é um caminho para solução.

Introdução a algoritmos

- Perguntas pressupõem respostas

- Dois padres queriam saber se era permitido fumar e rezar ao mesmo tempo.
- O primeiro perguntou ao Papa: "É permitido fumar enquanto se reza?"
- O segundo perguntou: "É permitido rezar enquanto se fuma?"
- Fica claro que o Papa respondeu sim e não à mesma resposta!

Introdução a algoritmos

- Perguntas pressupõem respostas

- Uma aldeia na Lituânia sofreu uma epidemia em que a vítima entrava em coma mortal, e os médicos não sabiam dizer se o paciente estava morto ou não.
- Uma das soluções encontradas foi enterrar o suposto morto com comida e uma abertura para respirar e gritar por socorro.
- A outra solução foi afixar uma estaca no caixão, na altura do coração.

Introdução a algoritmos

- A primeira solução respondia à pergunta: Como ter certeza de que a pessoa enterrada está viva?
- A segunda solução respondia à pergunta: Como ter certeza de que a pessoa enterrada está morta?
- "Para quem tem um martelo, tudo parece prego."

Introdução a algoritmos

- Algoritmos computacionais e não-computacionais: passíveis ou não de serem resolvidos por computadores.
- Exemplos:
 - Fazer uma busca de um termo na internet;
 - Compor uma música;

Introdução a algoritmos

- Exemplo: Filmes de Hollywood (Star Wars, Karatê Kid, O Silêncio dos Inocentes)
 - Comece com um herói;
 - Forneça a este um chamado que apele à sua história;
 - Apresente um mentor;
 - Passe o herói por um desafio/provação;
 - Conclua com o aprendizado adquirido.

Introdução a algoritmos

- Exemplo: Fritar um ovo
 - Colocar um ovo na frigideira;
 - Esperar o ovo ficar frito;
 - Remover o ovo da frigideira;
- Detalhes são importantes, principalmente para computadores.

Introdução a algoritmos

- Exemplo: Fritar um ovo

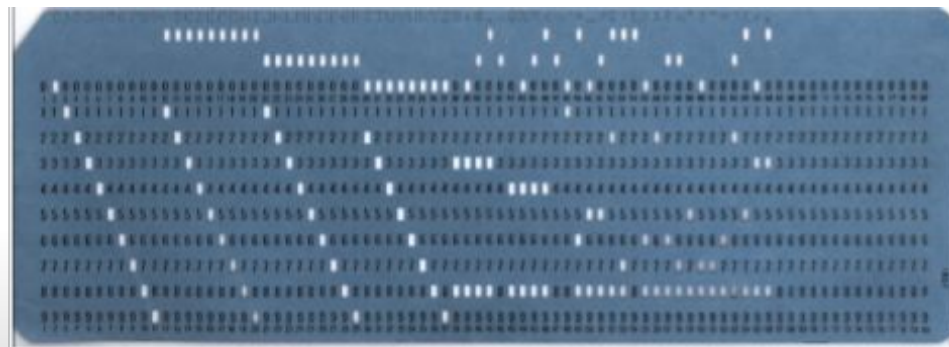
- Retirar um ovo da geladeira;
- Colocar a frigideira no fogo;
- Colocar óleo na frigideira e esperar o óleo ficar quente;
- Quebrar o ovo separando a casca;
- Colocar o conteúdo do ovo na frigideira;
- Esperar um minuto e retirar o ovo da frigideira;
- Apagar o fogo.

Introdução a algoritmos

- Para funcionarem, os algoritmos pressupõem o conhecimento da linguagem que está sendo utilizada.
- No caso de computadores, temos duas linguagens: a de programação e o código de máquina.
- Linguagem de programação: método para expressar instruções ao computador – C, C++, Pascal, Java etc.

Introdução a algoritmos

- Código de máquina: código executável pela máquina.
- Compilador: tradutor de linguagem de programação para o código de máquina.



- Programa de computador: implementação de um algoritmo.

Introdução a algoritmos

- Forma geral de um programa:

Algoritmo Nome_Do_Algoritmo

var

<Declaração de variáveis>

inicio

<Instruções>

fim_do_algoritmo

Introdução a algoritmos

- Forma geral de um programa:

Algoritmo Nome_Do_Algoritmo

Solução proposta

<Declaração de variáveis>

Manda reservar espaço
para guardar informações

inicio

<Instruções>

fim_do_algoritmo

- Obtém dados de entrada
- Processa informações
usando matemática, testes e
repetições
- Fornece solução
- Termina

Introdução a algoritmos

- Elementos básicos:

- Entrada
- Saída
- Matemática
- Testes
- Repetição

Introdução a algoritmos

- Elementos básicos:

- Entrada
- Saída
- Matemática
- Testes
- Repetição

Dados do problema

Solução

Cálculos que levam à solução

Condições que levam à solução

Iterações que levam à solução

Introdução a algoritmos

- Exemplo: Média de alunos diferentes que tiram dúvidas com o professor ao longo da semana

Dia	Aluno	Dúvida
Segunda-feira	João	Algoritmos
	Maria	Algoritmos
Terça-feira	Maria	Estrut. programa
	José	Estrut. programa
Quarta-feira	João	Algoritmos
	Raimundo	Operadores
Quinta-feira	Amanda	Funções
Sexta-feira	João	Ponteiros
	Maria	Ponteiros
	Amanda	Ponteiros
	Raimundo	Ponteiros

Introdução a algoritmos

- Teste: avaliar os dados de entrada.

Dia	Aluno	Dúvida
Segunda-feira	João Maria	Algoritmos Algoritmos
Terça-feira	Maria José	Estrut. programa Estrut. programa
Quarta-feira	João Raimundo	Algoritmos Operadores
Quinta-feira	Amanda	Funções
Sexta-feira	João Maria Amanda Raimundo	Ponteiros Ponteiros Ponteiros Ponteiros

Introdução a algoritmos

- Repetição – fazer a contagem todos os dias.

Dia	Aluno	Dúvida	Total
Segunda-feira	João Maria	Algoritmos Algoritmos	2
Terça-feira	Maria José	Estrut. programa Estrut. programa	1
Quarta-feira	João Raimundo	Algoritmos Operadores	1
Quinta-feira	Amanda	Funções	1
Sexta-feira	João Maria Amanda Raimundo	Ponteiros Ponteiros Ponteiros Ponteiros	0

Introdução a algoritmos

- Matemática – somar tudo e dividir pelo número de dias.

Dia	Aluno	Dúvida	Total
Segunda-feira	João Maria	Algoritmos Algoritmos	2
Terça-feira	Maria José	Estrut. programa Estrut. programa	1
Quarta-feira	João Raimundo	Algoritmos Operadores	1
Quinta-feira	Amanda	Funções	1
Sexta-feira	João Maria Amanda Raimundo	Ponteiros Ponteiros Ponteiros Ponteiros	0

Média = $(2+1+1+1+0)/5 = 1$ aluno diferente por dia

Estrutura de um programa

- Hello World

```
#include <stdio.h>

void main()
{
    printf("Hello world!");
    /*Comentario*/
}
```

Estrutura de um programa

- Hello World

```
#include <stdio.h>

void main()
{
    printf("Hello world!");
    /*Comentario*/
}
```

Estrutura de um programa

- Hello World

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

```
void main()
```

```
{
```

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

```
    /*Comentario*/
```

```
}
```



Hello world!

Estrutura de um programa

- Hello World

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

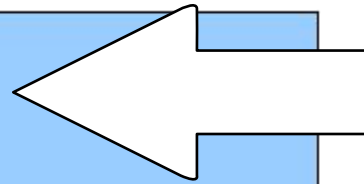
```
void main()
```

```
{
```

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

```
    /*Comentario*/
```

```
}
```



Biblioteca

STanDard

Input/Output

==>

Entrada e saída

padronizadas

Estrutura de um programa

- Hello World

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

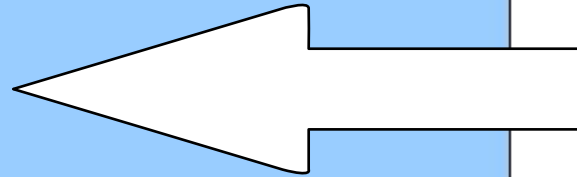
```
void main()
```

```
{
```

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

```
    /*Comentario*/
```

```
}
```

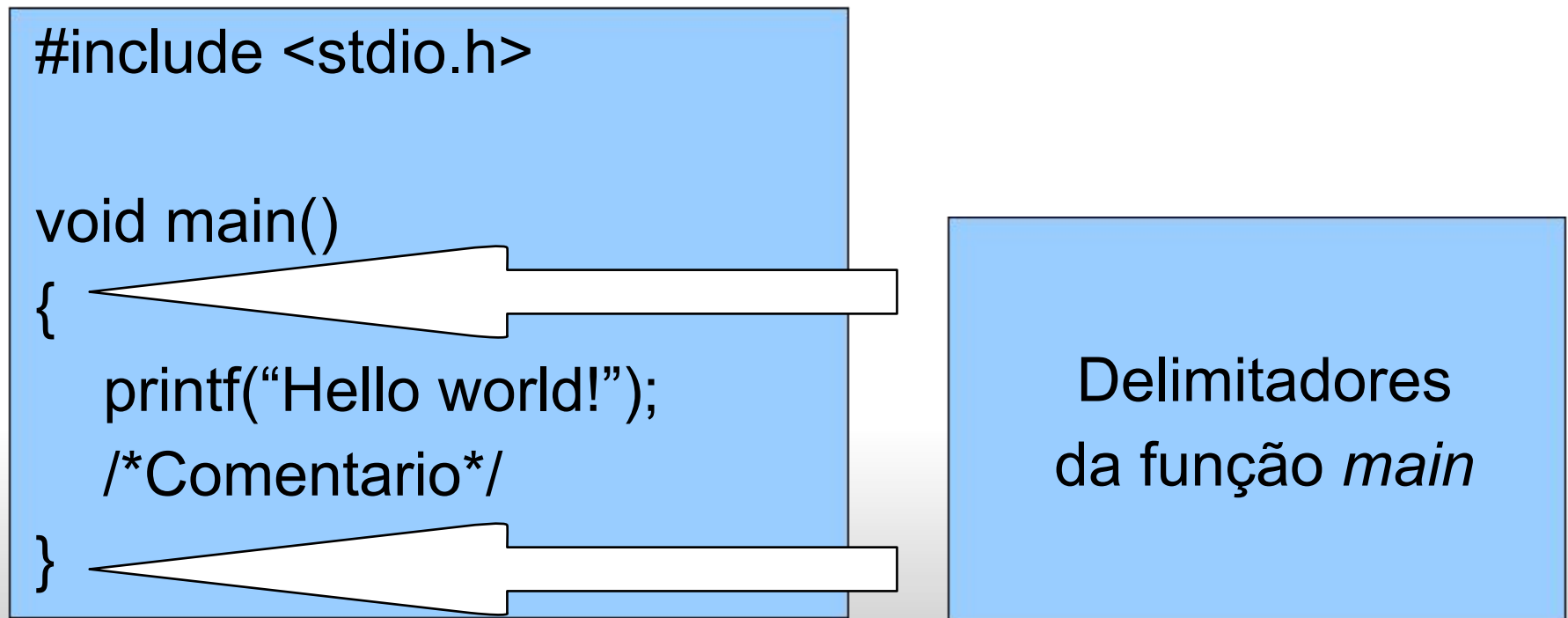


Declaração da
função principal
deste programa

Obrigatório
em qualquer
programa em C

Estrutura de um programa

- Hello World



Estrutura de um programa

- Hello World

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

```
void main()
```

```
{
```

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

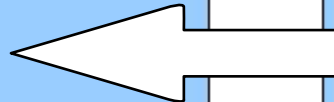
```
    /*Comentario*/
```

```
}
```

Chamada à
função printf()

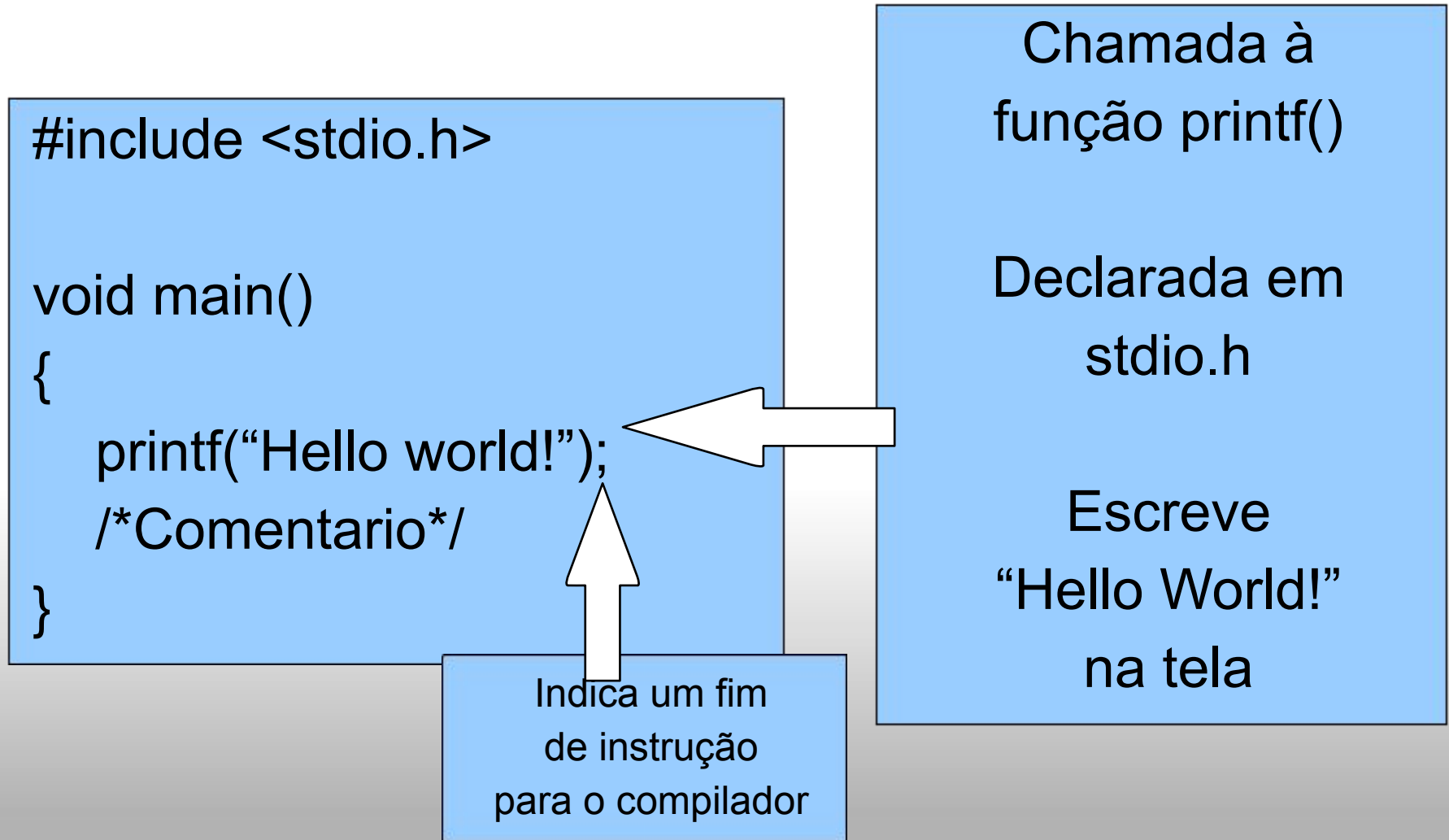
Declarada em
stdio.h

Escreve
"Hello World!"
na tela



Estrutura de um programa

- Hello World



Estrutura de um programa

- Hello World

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

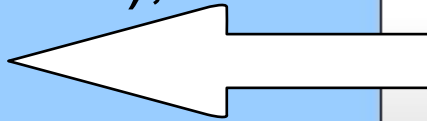
```
void main()
```

```
{
```

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

```
    /*Comentario*/
```

```
}
```



Comentário

Não afeta o
programa
em nada

Útil somente ao
programador

Estrutura de um programa

- Hello World

- Pode ser escrito na forma:

```
#include <stdio.h> void main(){printf("Hello world!");/*Comentario*/}
```

- Ilegível!

Estrutura de um programa

- Hello World

```
#include <stdio.h>

void main()
{
    printf("Hello world!");
    printf("Ola mundo!");
    /*Comentario*/
}
```

```
#include <stdio.h>


void main()
{
    printf("Hello world! Ola mundo!");
    /*Comentario*/
}
```

Estrutura de um programa

- Hello World

```
#include <stdio.h>

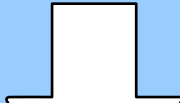
void main()
{
    printf("Hello world!");
    printf("Ola mundo!");
    /*Comentario*/
}
```



Hello world!Ola mundo!

```
#include <stdio.h>

void main()
{
    printf("Hello world! Ola mundo!");
    /*Comentario*/
}
```



Hello world! Ola mundo!

Estrutura de um programa

- Comentários

```
#include <stdio.h>

void main()
{
    printf("Hello world!");
    /*Comentario 1*/
    //Comentario 2
    printf("Ola mundo!");
}
```

Estrutura de um programa

- Comentários

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

```
void main()
```

```
{
```

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

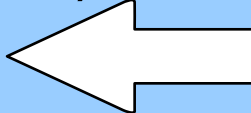
```
    /*Comentario 1*/
```

```
    //Comentario 2
```

```
    printf("Ola mundo!");
```

```
}
```

Compilador
ignora tudo entre
/* e */



Estrutura de um programa

- Comentários

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

```
void main()
```

```
{
```

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

```
    /*Comentario 1*/
```

```
    //Comentario 2
```

```
    printf("Ola mundo!");
```

```
}
```

Compilador
ignora tudo a
partir de //
ATÉ O FINAL
DA LINHA

Estrutura de um programa

- Comentários

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

```
void main()
```

```
{
```

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

```
    /*Comentario 1*/
```

```
    //Comentario 2 printf("Ola mundo!");
```

```
}
```

Estrutura de um programa

- Comentários

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

```
void main()
```

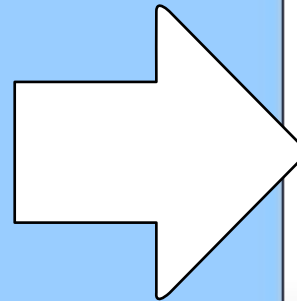
```
{
```

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

```
    /*Comentario 1*/
```

```
    //Comentario 2 printf("Ola mundo!");
```

```
}
```



Hello world!

Estrutura de um programa

- Comentários

```
#include <stdio.h>

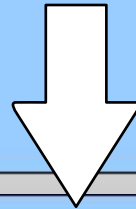
void main()
{
    printf("Hello world!");
    /*Comentario 1*/ printf("Ola mundo!");
    //Comentario 2
}
```

Estrutura de um programa

- Comentários

```
#include <stdio.h>

void main()
{
    printf("Hello world!");
    /*Comentario 1*/ printf("Ola mundo!");
    //Comentario 2
}
```



Hello world!Ola mundo!

Variáveis e constantes

- Variáveis

- Espaço na memória para armazenar informação;
- Utilizadas em cálculos matemáticos, testes e repetição;
- Memória binária: informação toda numérica.

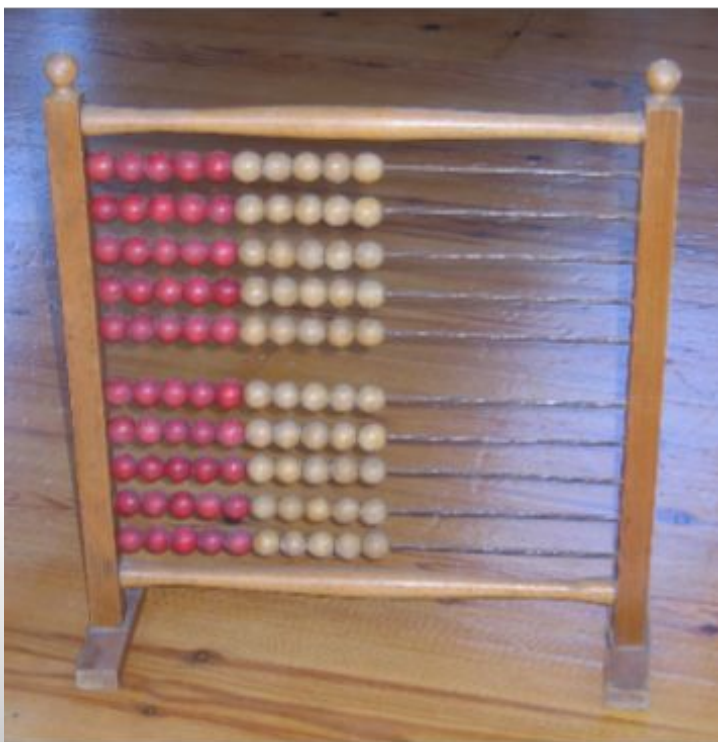
Variáveis e constantes

- Variáveis



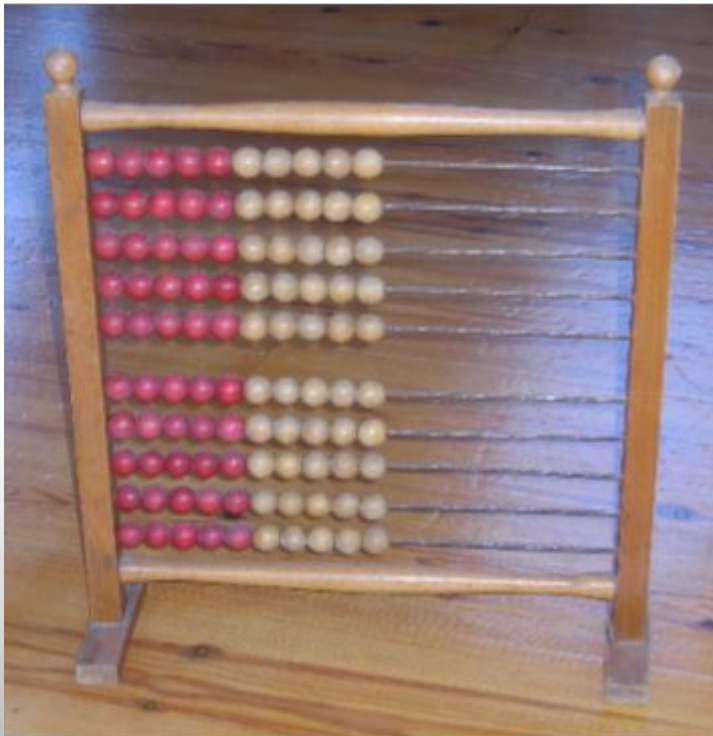
Variáveis e constantes

- Variáveis



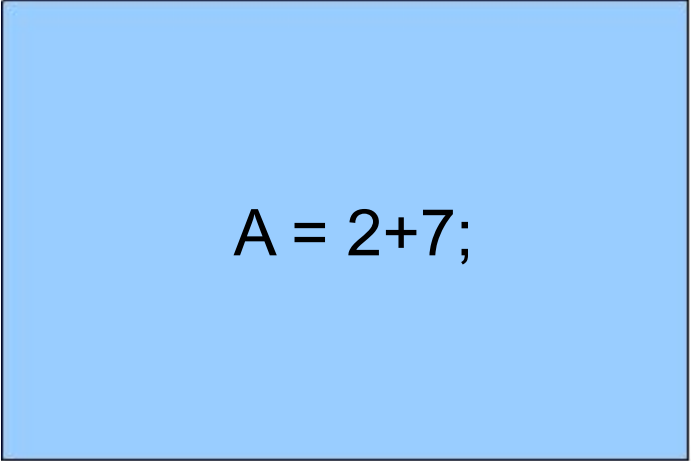
Variáveis e constantes

- Variáveis
 - Arquivo de ábacos binários



Variáveis e constantes

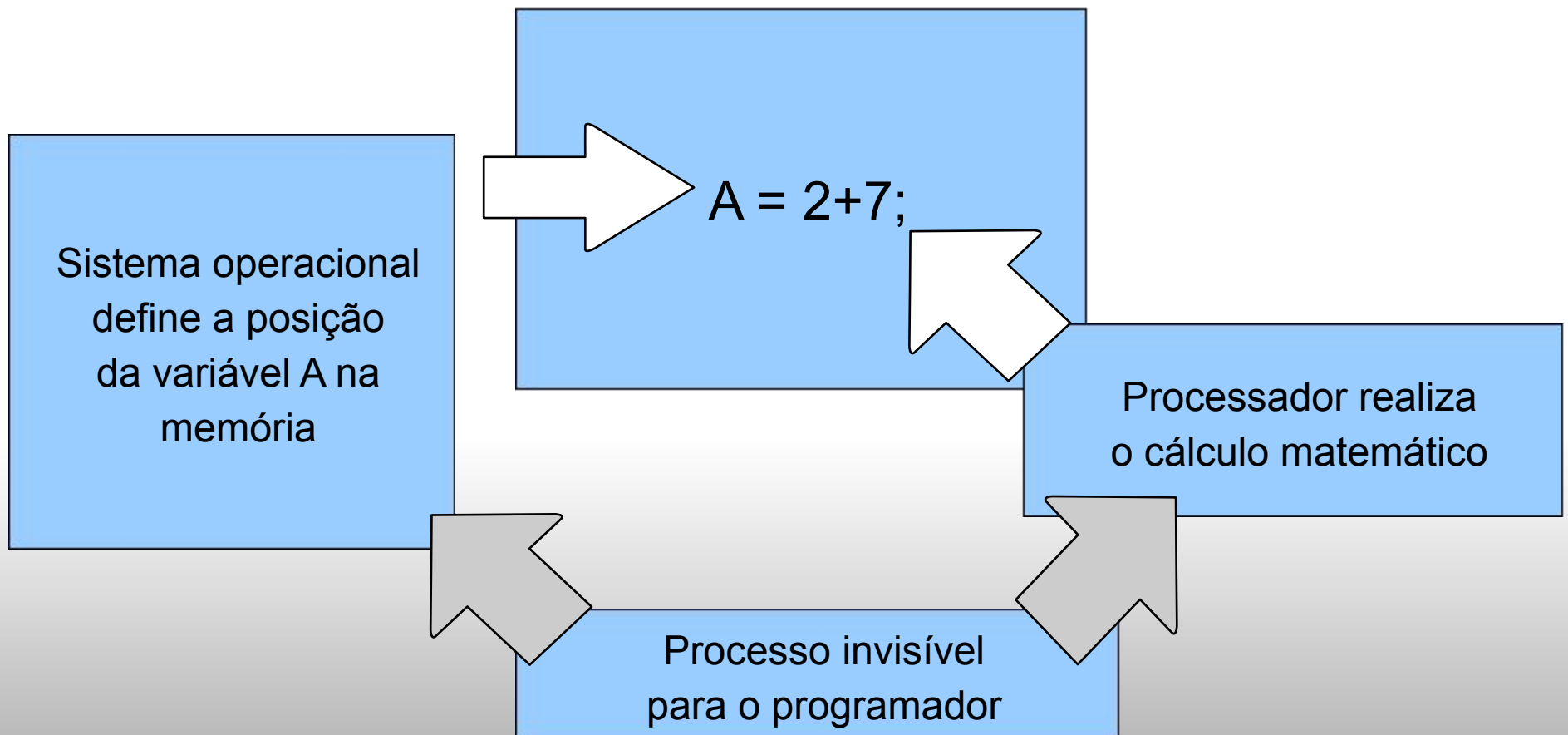
- Variáveis – Soma aritmética: 2+7



$A = 2+7;$

Variáveis e constantes

- Variáveis – Soma aritmética: $2+7$



Variáveis e constantes

- Variáveis – Soma aritmética: 2+7

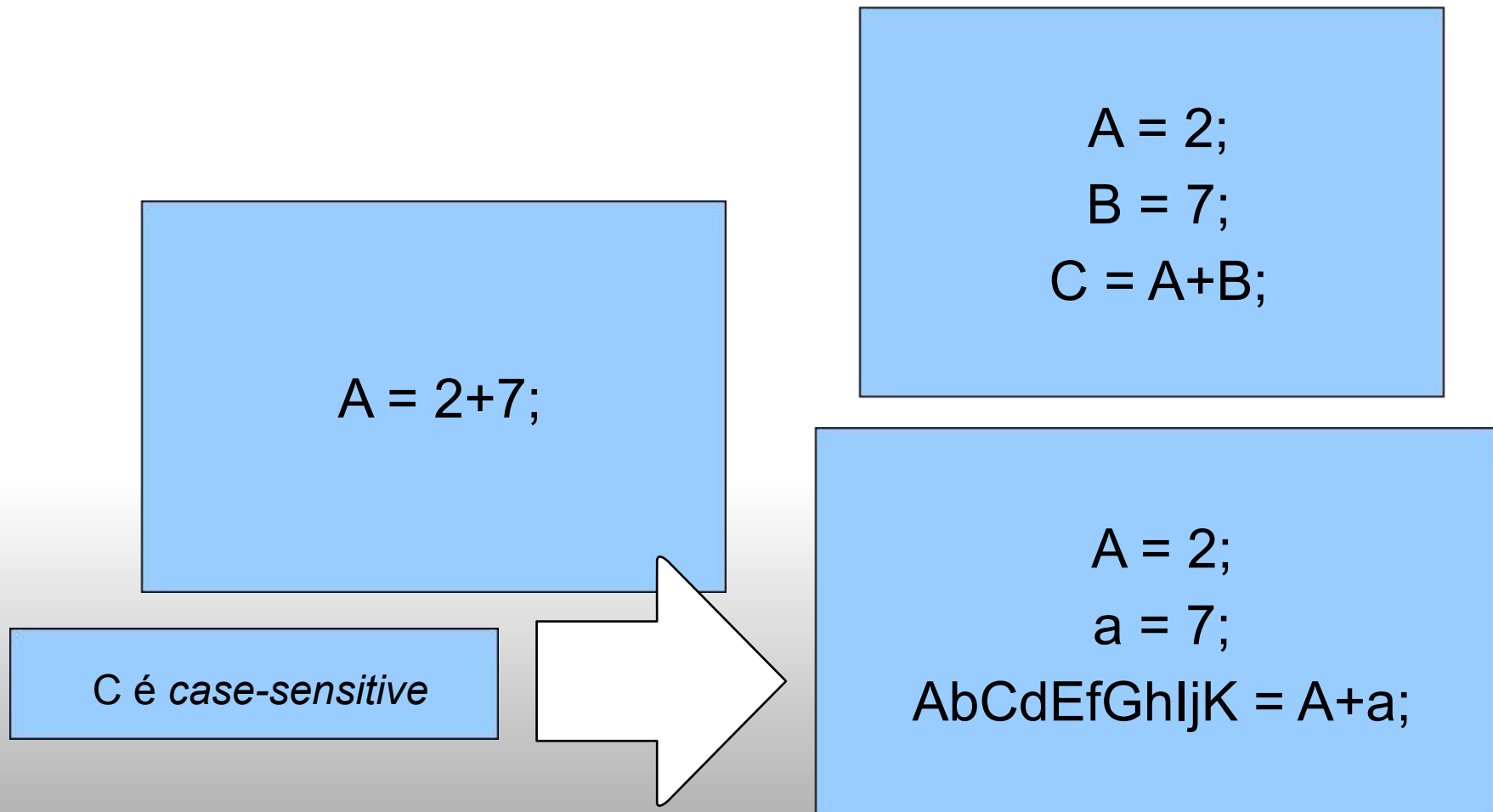
$A = 2+7;$

$A = 2;$
 $B = 7;$
 $C = A+B;$

$A = 2;$
 $a = 7;$
 $AbCdEfGhIjK = A+a;$

Variáveis e constantes

- Variáveis – Soma aritmética: 2+7



Variáveis e constantes

- Variáveis só podem ter nomes que contenham letras, números e o símbolo de *underscore* _
- Sempre só podem começar com letras ou com o *underscore*. Para não ter problemas, sempre comece com letras.

Variáveis e constantes

- Variáveis não podem ter os seguintes nomes, que fazem parte da sintaxe básica do C. Portanto, você se acostuma.

asm, auto, bool, break, case, catch, char, class, const, const_cast, continue, default, delete, do, double, dynamic_cast, else, enum, explicit, export, extern, false, float, for, friend, goto, if, inline, int, long, mutable, namespace, new, operator, private, protected, public, register, reinterpret_cast, return, short, signed, sizeof, static, static_cast, struct, switch, template, this, throw, true, try, typedef, typeid, typename, union, unsigned, using, virtual, void, volatile, wchar_t, while

Variáveis e constantes

- Tipos de variáveis

- O programador deve ter um conhecimento básico do tipo de variável que ele está trabalhando;

- Por exemplo:

- O número de alunos numa turma é representado por um inteiro;
- A média de alunos por aula é representada por um número decimal.

Variáveis e constantes

Variable Type	Keyword	Bytes Required	Range
Character	char	1	-128 to 127
Unsigned character	unsigned char	1	0 to 255
Integer	int	2	-32768 to 32767
Short Integer	short int	2	-32768 to 32767
Long Integer	long int	4	-2,147,483,648 to 2,147,438,647
Unsigned Integer	unsigned int	2	0 to 65535
Unsigned Short integer	unsigned short int	2	0 to 65535
Unsigned Long Integer	unsigned long int	4	0 to 4,294,967,295
Float	float	4	1.2E-38 to
Double	double	8	2.2E-308 to
Long Double	long double	10	3.4E-4932 to 1.1E+4932

O tipo da variável define o tamanho do “ábaco digital”

Variáveis e constantes

- Declaração de variáveis

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

```
void main()
```

```
{
```

```
    char A;
```

```
    char B;
```

```
    char C;
```

```
    A = 2;
```

```
    B = 7;
```

```
    C = A+B;
```

```
}
```

Variáveis e constantes

- Declaração de variáveis

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

```
void main()  
{
```

```
    char A;
```

```
    char B;
```

```
    char C;
```

```
    A = 2;
```

```
    B = 7;
```

```
    C = A+B;
```

```
}
```

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

```
void main()  
{
```

```
    char A, B, C;
```

```
    A = 2;
```

```
    B = 7;
```

```
    C = A+B;
```

```
}
```

Variáveis e constantes

- Declaração de variáveis

```
#include <stdio.h>

void main()
{
    unsigned int A, B, C;
    A = 234;
    B = 78;
    C = A*B;
}
```

```
#include <stdio.h>

void main()
{
    float A, B, C;
    A = -2.0;
    B = 77.987;
    C = A/B;
}
```

Variáveis e constantes

- Inicialização de variáveis

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

```
void main()
```

```
{
```

```
    float A = 2.0;
```

```
    float B = 77.987;
```

```
    float C;
```

```
    C = A/B;
```

```
}
```

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

```
void main()
```

```
{
```

```
    float A = 2.0, B = 77.987, C;
```

```
    C = A/B;
```

```
}
```

Variáveis e constantes

- Constantes

- Números inteiros:

- Base decimal: 17, -809, 75

- Base octal: 0113

==> Precedidos por um zero

- Base hexadecimal: 0x4b, 0x4B

==> Precedidos por um zero e um x

Variáveis e constantes

- Constantes

- Números em ponto flutuante:

- 3.14159

- 6.02e23

- 1.6e-19

- 3.0

Variáveis e constantes

- Constantes

- Caracteres e *strings*

- 't'

- "Hello World!"

- "t"

- '\n'

- "Hello World!\nOla Mundo!"

Variáveis e constantes

- Constantes – tabela ASCII

	0	1	2	3	4	5	6	7
0	NUL	DLE	space	0	@	P	`	p
1	SOH	DC1 XON	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3 XOFF	#	3	C	S	c	s
4	EOT	DC4	\$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(8	H	X	h	x
9	HT	EM)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[k	{
C	FF	FS	,	<	L	\	l	
D	CR	GS	-	=	M]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	del

Variáveis e constantes

- Constantes

- Caracteres e *strings*:

Constantes de barra

invertida

\n	Nova linha
\r	<i>Carriage return</i>
\t	Tabulação horizontal
\v	Tabulação vertical
\b	<i>Backspace</i>
\f	Alimentação de formulário
\a	<i>Beep</i>
\'	Aspas simples
\"	Aspas duplas
\\	Barra invertida

Variáveis e constantes

- Constantes definidas

- O programador pode definir nomes para valores muito usados, sem ocupar espaço na memória;
- O compilador simplesmente substitui o nome pelo valor na hora de compilar o código.

Variáveis e constantes

- Constantes definidas

```
#include <stdio.h>

#define PI 3.14159
#define PULO '\n'

void main()
{
    char p = PULO;
    float R = 5, circ;

    circ = 2*PI*R;
}
```

Variáveis e constantes

- Constantes declaradas

- Variáveis que não podem ser mudadas (ocupam espaço na memória);

```
#include <stdio.h>

void main()
{
    Const float PI = 3.14159;
    float R = 5, circ;

    circ = 2*PI*R;
}
```

Variáveis e constantes

- Visualização de variáveis

- Função *printf()*:

- `printf("Ola mundo");` → Escreve Ola mundo na tela

- Como escrever valores de variáveis na tela?

Variáveis e constantes

- Visualização de variáveis

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

```
void main()
```

```
{
```

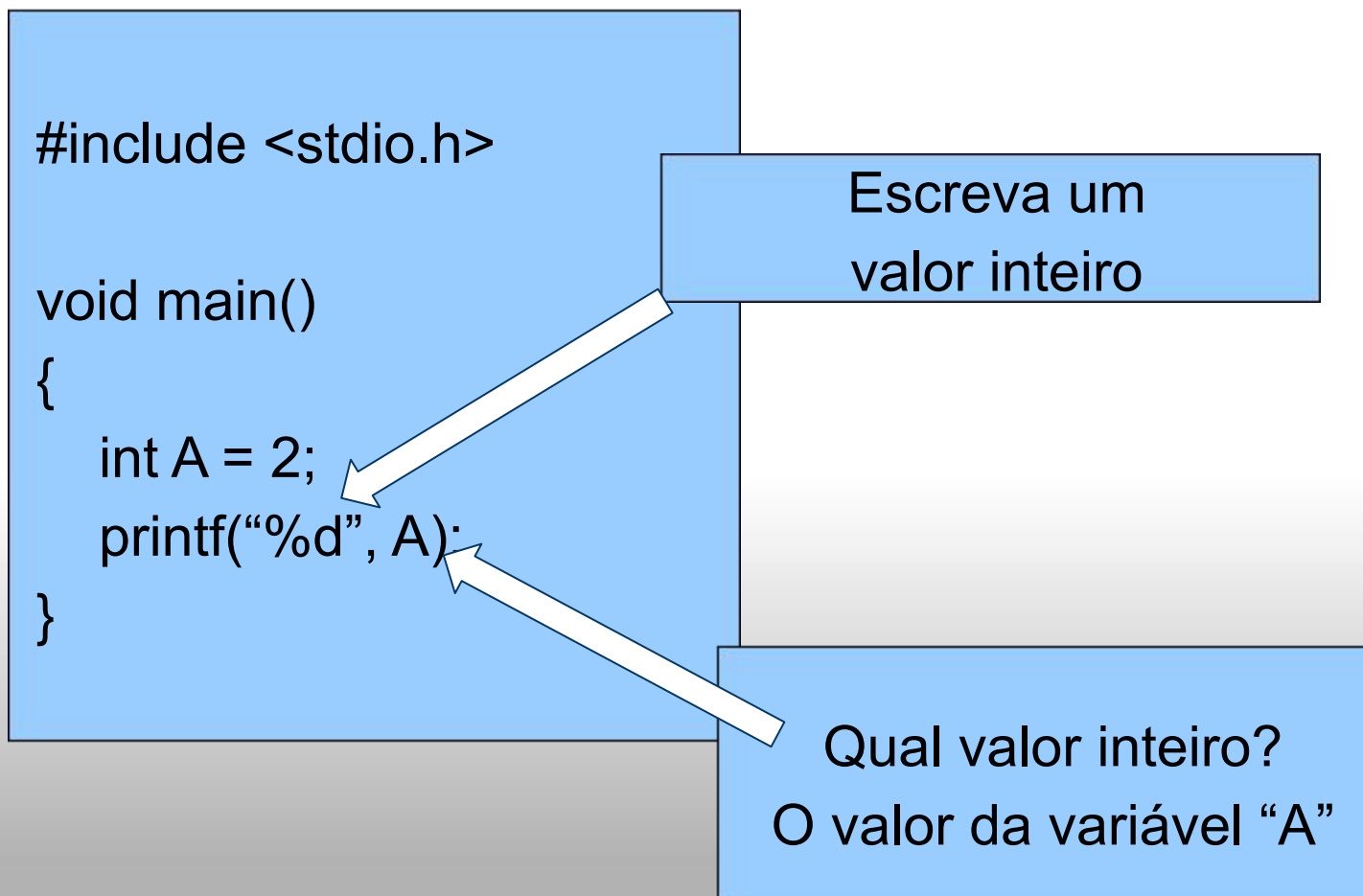
```
    int A = 2;
```

```
    printf("%d", A);
```

```
}
```

Variáveis e constantes

- Visualização de variáveis



Variáveis e constantes

- Visualização de variáveis

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

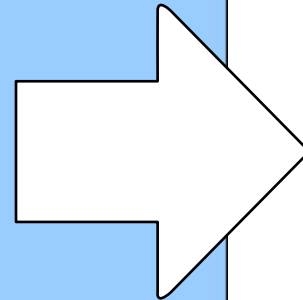
```
void main()
```

```
{
```

```
    int A = 2;
```

```
    printf("%d", A);
```

```
}
```



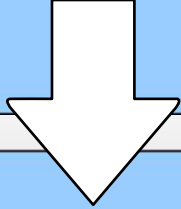
2

Variáveis e constantes

- Visualização de variáveis

```
#include <stdio.h>

void main()
{
    int A = 2;
    printf("O valor de A eh %d, tah entendendo?", A);
}
```



O valor de A eh 2, tah entendendo?

Variáveis e constantes

- Visualização de variáveis

```
#include <stdio.h>

void main()
{
    int A = 2;
    float B = 3.14;
    printf("%d %f", A, B);
}
```

Variáveis e constantes

- Visualização de variáveis

```
#include <stdio.h>

void main()
{
    int A = 2;
    float B = 3.14;
    printf("%d %f", A, B);
}
```

Escreva um valor
inteiro

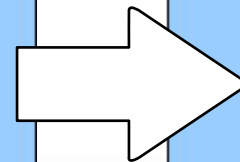
Escreva um valor
em ponto flutuante

Variáveis e constantes

- Visualização de variáveis

```
#include <stdio.h>

void main()
{
    int A = 2;
    float B = 3.14;
    printf("%d %f", A, B);
}
```



2 3.14

Variáveis e constantes

- Visualização de variáveis

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

```
void main()
```

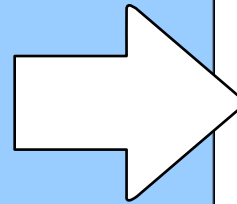
```
{
```

```
    int A = 2;
```

```
    float B = 3.14;
```

```
    printf("A vale %d, B vale %f", A, B);
```

```
}
```



A vale 2, B vale 3.14