Sumário

Parte I

Introdução

Breve histórico

O C++ é ...
O mínimo código em C++ é apresentado no código 1.

```
1 int main()
2 {
3     return 0;
4 }
```

Código 1: Código mínimo

Este é o código mais simples que um compilador C++ aceitará sem erros ou notas de atenção.

Capítulo 1

Sintaxe

1.1 O que é sintaxe

Sintaxe é o "componente do sistema linguístico que determina as relações formais que interligam os constituintes da sentença, atribuindo-lhe uma estrutura"[1]. Ou seja, é a maneira que as palavras de uma determinada linguagem se combinam formando sentenças.

No contexto de programação, sintaxe é o conjunto de regras estruturais que determinam o formato da linguagens. Como qualquer outra linguagem, o C++ tem um conjunto de regras a ser seguido para que possa ser entendido pelo compilador.

1.2 Os blocos de código e o ponto-e-vírgula

Qualquer linha do código que realize uma ação delimitada e determinada é denominada *comando*. Todo comando, que também pode ser denominado como *instrução*, e precisa de um caractere especial para ser dividido dos demais. Na criação da linguagem, foi escolhido o ponto-e-vírgula (`;`), provavelmente pelo lado poético de terminar uma oração delimitada sem finalizar a frase, o que é coerente, tendo em vista que o programa é uma série de comandos bem delimitadas que montam uma instrução mais complexa. É importante lembrar sempre: todo comando deve apresentar um ponto-e-vírgula (`;`) em sequência.

Em qualquer lugar é possível criar regiões de código com algumas propriedades especiais, estar regiões são denominadas *blocos de código*. Sua principal função é agrupar comandos, de tal forma que juntos formem uma instrução mais complexa. Pode-se dizer que os comando são as instruções passo-a-passo para a confecção de um bolo, e o bloco de código seja a própria expressão *faça um bolo*. Um bloco de código é delimitado pelos caracteres de chave esquerda (`{`) e direita (`}`).

```
{;}
```

1.3 A sensibilidade de caixa e os caracteres ASCII

A *sensibilidade de caixa* é uma características mais comuns e linguagens de programação modernas, embora possa parecer incoerente em linguagem coloquial, quando escrita pode ser a diferença entre um programa que é compilável ou não, essa sensibilidade é denominada *case sensitive*. De maneira simples, a caixa alta é o conjunto de caracteres de letras maiúsculas e a caixa baixa o conjunto de caracteres de letras minúsculas. Então, se em um lugar do código existir um termo como `palavra`, em outro o termo `Palavra` e em um terceiro lugar o termo `PALAVRA`, o compilador reconhecerá como três símbolos diferentes, completamente distintos e sem qualquer relação natural, mesmo que em um contexto coloquial sejam exatamente o mesmo termo.

Por se tratar de uma linguagem relativamente antiga, o C++ tem uma limitação nos caracteres processáveis, então alguns caracteres especiais não são aceitos pelo compilador, gerando um erro, mesmo que a sintaxe esteja perfeita e nenhum erro de lógica exista, caracteres como `ã` ou `ç` não são tolerados. Somente os caracteres da tabela ASCII são aceitos, esta pode ser consultada no apêndice A. Por isso ainda, é comum programadores escolherem escreverem seus programas usando termos em linguagens que não apresentem esse caracteres, como inglês, grego ou até latim.

$$A \neq a$$

1.4 Comentários

Ao longo do desenvolvimento de programas, é comum que o programador dê nome a coisas no programa, e, em programas maiores esses nomes podem se confundir. Para evitar isso, o programador pode criar uma tabela externa, como um arquivo de texto ou uma folha num caderno, mas isso se torna cansativo e não evolui caso o desenvolvedor não tome um tempo especial para isso. Para garantir que todos os nomes de coisas no programa sejam utilizados da forma correta, uma simples anotação no lugar onde são criados seria suficiente.

Todas as partes de processamento também ficam mais simples de entender quando há uma anotação a seu respeito explicando seu funcionamento. Tais anotações servem para explicar o programa, são consideradas uma boa prática de programação. Pode parecer algo desnecessário para programas pequenos, mas dutante a rotina de desenvolvimento de software, não é incomum passar um longo período de tempo sem editar alguns segmentos do programa e, depois de tanto tempo, é comum se esquecer pelo que esta parte é responsável e como ela o faz. Um simples comentário pode ser o suficiente para que não sejam perdidas horas de análise de código para simplesmente lembrar qual o objetivo deste segmento.

Em C++, existem duas formas de criar comentários, a mais comum utilizando duas barras (`//`), depois desta sequência, o restante da linha é considerada comentário.

```
//Linear
```

A segunda forma, muito utilizada para manter cópias de códigos alternativos no arquivo fonte (o arquivo de código), é semelhante a declaração de blocos, com um delimitador de direita e outro de esquerda, para abertura e fechamento respectivo. O início do bloco é delimitado com o par barra-asterisco (`/*`) e o fim por asterisco-barra (`*/`).

```
/*Blocular*/
```

Os comentários devem respeitar o uso de caracteres ASCII, caracteres especiais continuam proibidos, porém qualquer outra coisa pode ser escrita sem maiores problemas.

1.5 Um pequeno exemplo de sintaxe

O código 1.1 apresenta alguns exemplos de sintaxe. Note a forma que a palavra `int` não está com a coloração diferenciada como no código 1.

```
1  Int main()      //Esta linha nao sera aceita por causa da letra maiuscula
2  {               //O bloco principal inicia aqui
3      return 0;   //Este comando tem ; no final
4  }               //O bloco principa acaba aqui
5
6  /*
7  Aqui podemos escrever qualquer coisa maior, normalmente este tipo de
       comentario contem nome do programa, do programador, etc
8  int main()
9  {
10     return 0;
11 }
12 */
```

Código 1.1: Código mínimo com exemplos de sintaxe

Capítulo 2

Variáveis

Capítulo 3

Operadores Aritiméticos

Capítulo 4

Controladores de Fluxo

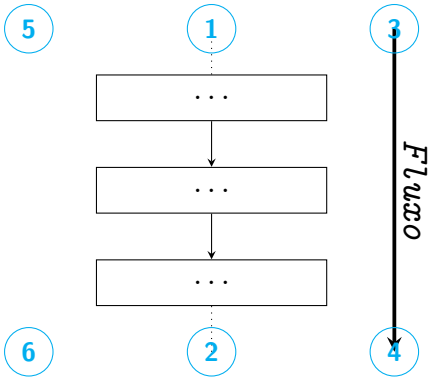

Figura 4.1: Fluxograma de fluxo simples

4.1 Decisões na direção do fluxo

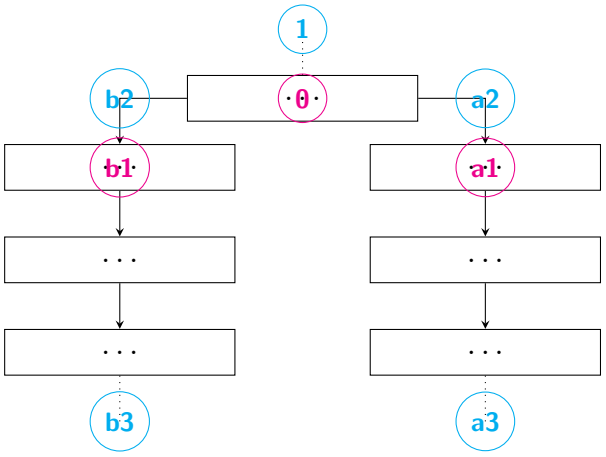

Figura 4.2: Fluxograma de fluxo ambíguo

4.2 O *if* e a estrutura básica de decisão

```
//...
if(<cond>) <comand>;
//...
```

Código 4.1: *if* simples linear

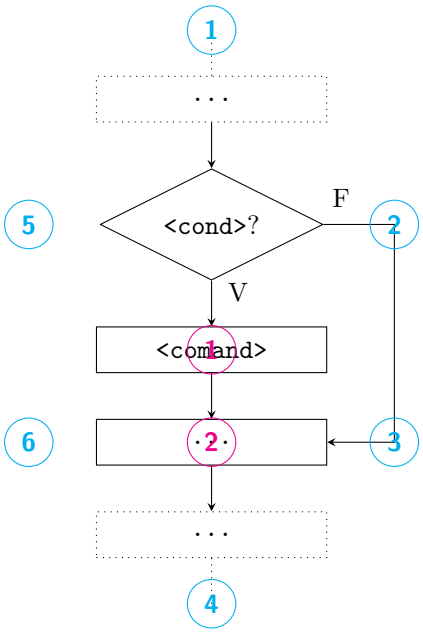

Figura 4.3: Fluxograma de *if* simples linear

```
//...
if(<cond>)
{
    <comand1>;
    <comand2>;
    //...
    <comandN>;
}
//...
```

Código 4.2: *if* simples blocular

4.3 O *else* e as estruturas duplas

```
//...
if(<cond>) <comandA>;
else <comandB>;
//...
```

Código 4.3: *if* composto linear

```
//...
if(<cond>)
{
    <comandA1>;
    <comandA2>;
    //...
```

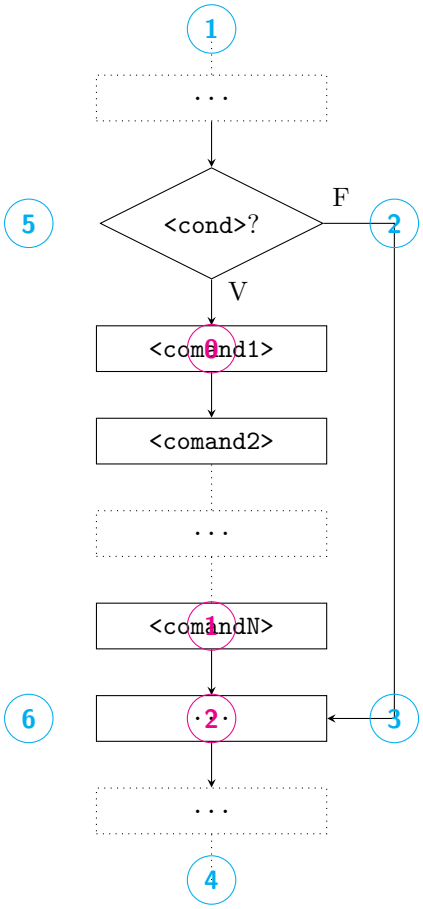

Figura 4.4: Fluxograma de *if* simples blocular

```
    <comandAN>;
}
else
{
    <comandB1>;
    <comandB2>;
    //...
    <comandBM>;
}
//...
```

Código 4.4: *if* composto blocular

```
//...
if(<cond0>)
{
    //...
    if(<condA>) <comandAA>;
    else <comandAB>;
    //...
}
else
{
    //...
    if(<condB>) <comandBA>;
    else <comandBB>;
```

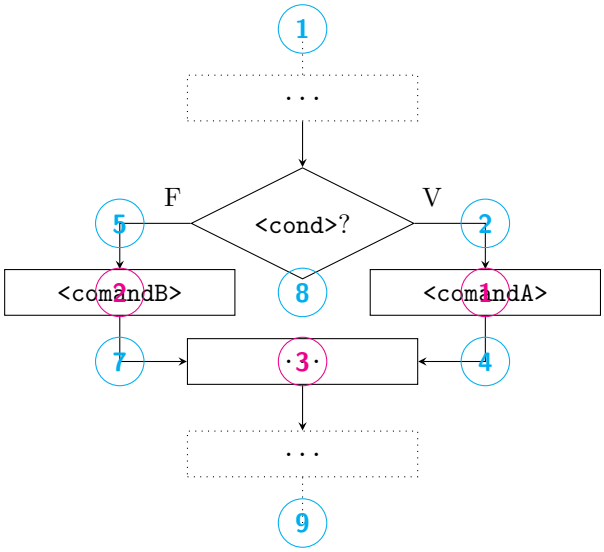

Figura 4.5: Fluxograma de *if* composto linear

```
14        //...
15    }
16    //...
```

Código 4.5: *if* aninhado

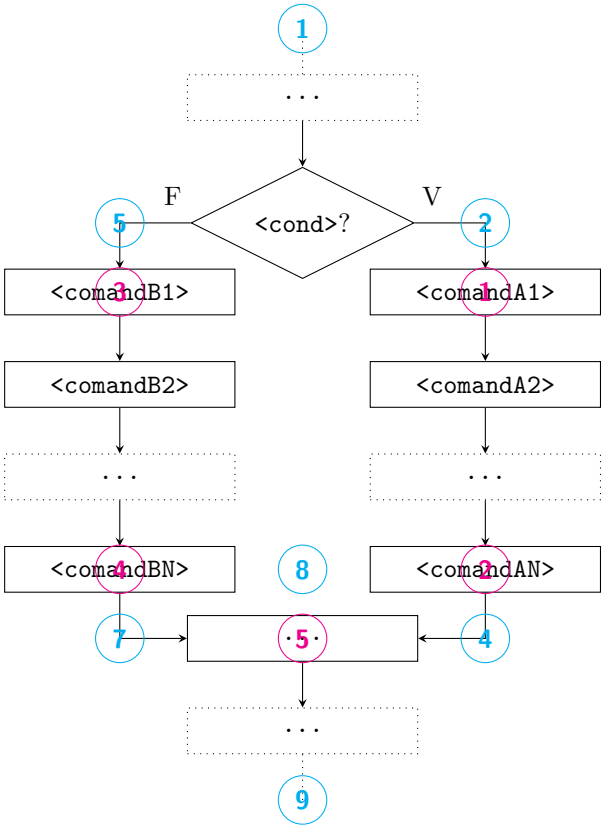

Figura 4.6: Fluxograma de *if* composto blocular

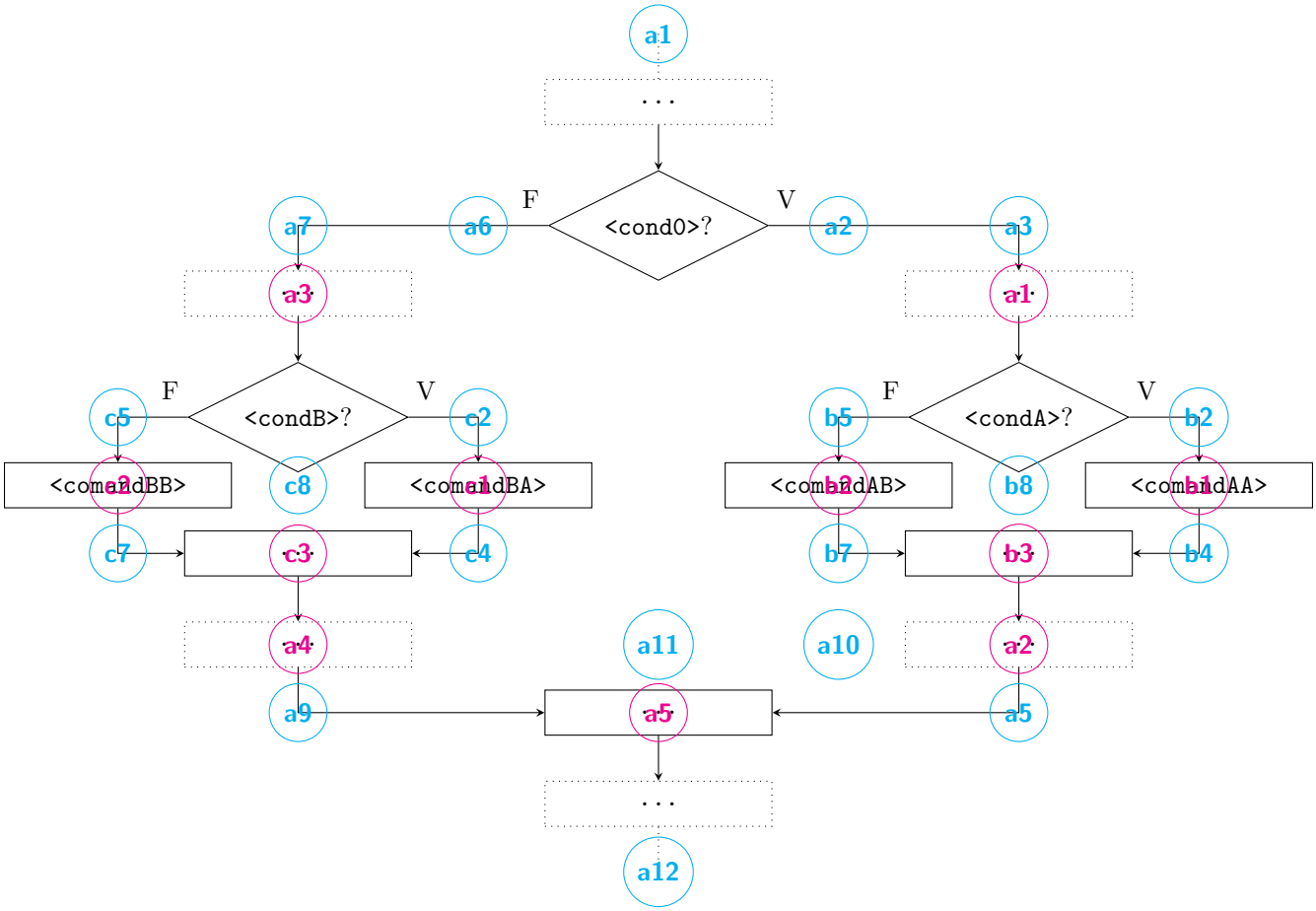

Figura 4.7: Fluxograma de *if* aninhado

Lista de Figuras

Códigos

Referências Bibliográficas

[1] A. Houaiss, M. Villar, F. de Mello Franco, and I. A. H. de Lexicografia, *Dicionário Houaiss da língua portuguesa*. Objetiva, 2009. [Online]. Available: https://books.google.com.br/books?id=1LQKqAAACAAJ

Apêndice A

Tabela ASCII

A tabela A.1 apresenta a codificação ASCII extendida, dividida em duas partes, na superio os tipos padrão, na inferior os tipos especiais.

A divisão da tabela é devida à codificação em bytes com um bit de paridade, o que diminui pela metade a capacidade de codificação de um byte.

Tabela A.1: Codificação ASCII extendida incompleta[1].

xX	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0X	NULL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1X	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2X		!	"	#	$	%	&	’	(	)	*	+	,	-	.	/
3X	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4X	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5X	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6X	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7X	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL
8X	Ç	ü	é	â	ä	à	å	ç	ê	ë	è	ï	î	ì	Ä	Å
9X	É	æ	Æ	ô	ö	ò	û	ù	ÿ	Ö	Ü	ø		Ø	×	f
AX	á	í	ó	ú	ñ	Ñ	ª	º	¿	®	¬	½	¼	¡		
BX	▒	▓	▓	│	┤	Á	Â	À	©	╣	║	╗	╝	¢	¥	┐
CX	└	┴	┬	├	─	┼	ã	Ã	╚	╔	╩	╦	╠	=	╬	¤
DX			Ê	Ë	È	ı	Í	Î	Ï	┘	┌	▓	▓	¦	Ì	▓
EX	Ó	ß	Ô	Ò	õ	Õ	µ			Ú	Û	Ù	ý	Ý	¯	´
FX		±		¾	¶		÷	¸	°	¨	·	¹	³	²		nbsp

[1]Problemas na codificação de caracteres impediram a renderização de alguns tipos, os espaços em branco na parte inferior são os destes tipos. Uma busca na internet pode localizar facilmente uma tabela completa.