Alunos: João Pedro Winckler Bernardi jpwbernardi@hotmail.com
Sival Francisco de Oliveira Junior jr_sival@hotmail.com

Linguagem C--

Comando para executar: java -jar CLL.jar *nome_do_arquivo*.cll
A extensão do arquivo deve ser .cll

1. Expressões:

A linguagem permite flexibilidade de espaços em uma mesma linha, ou entre um mesmo comando. Toda linha de código deve ser finalizada com ponto e vírgula, exceto quando o comando exige abertura de escopo. Nesse caso, termina com a abertura de chaves.

Exemplos	Equivalentes
j = 0;	j = 0;
i = 4 - 3 * (4 - 2);	i = 4 - 3 * (4 - 2);
p = 2 * 9;	p = 2 * 9;

2. Comentários:

Para fazer um comentário, basta colocar o símbolo #. Tudo que estiver depois do símbolo na linha, será desconsiderado em termos de código.

3. Declaração de variáveis:

Não existe declaração de variável, e todas as variáveis são double. Toda variável é considerada como declarada, sendo que quando não atribuída, seu valor é 0. Quando precisar de uma variável, apenas faça uma atribuição. Todo nome de variável deve começar com uma letra, conter apenas letras minúsculas e números. A atribuição é feita como da seguinte forma:

nome_variavel = expressão_matemática;

Exemplos:

```
x1y1 = 3;
i = (x1y1 + 4) * -3;
exemplo3 = x1y1;
```

4.Controle de fluxo:

O comando *if* verifica se dada expressão é verdadeira. Se for, executa os comandos que estão no escopo definido por chaves. É da forma:

```
if (condição) {
        comandos
}
```

É obrigatório o uso de chaves e a a linguagem permite aninhamento de controladores de fluxo. Na condição, as operações que não forem aritméticas devem ser obrigatoriamente desenvolvidas separadamente, sendo então necessária a utilização de parênteses. Exemplos:

Errado	Correto

if (abc IGUAL 2 EE cba DIF 4 EE outra MENORI 4)	if (((abc IGUAL 2) EE (cba DIF 4)) EE (outra MENORI 4))
	ou
	if ((abc IGUAL 2) EE ((cba DIF 4) EE (outra MENORI 4)))

5.Laços:

O comando *loop* verifica se a expressão é verdadeira, como no *if*. No entanto, enquanto ela for verdadeira, os comandos dentro do escopo definido serão executados. Funciona na forma:

```
loop (condição) {
        comandos
}
```

A resolução da condição segue o mesmo princípio do controlador de fluxo. Os comandos *break* e *continue* funcionam como na linguagem C. A linguagem permite aninhamento de laços.

6. Operadores aritméticos e lógicos:

Os comandos aritméticos permanecem os mesmos que o habitual, sendo + para soma, - para subtração, * para multiplicação, / para divisão, % para resto da divisão.

Aritméticos	
Operador	**Operação**
+	Soma
-	Subtração
*	Multiplicação
/	Divisão
%	Resto da divisão

Lógicos	
Operador	**Equivalente**
MAIOR	Maior (>)
MENOR	Menor (<)
IGUAL	Igual (==)
DIF	Diferente (!=)
MENORI	Menor ou igual (<=)
MAIORI	Maior ou igual (>=)
EE	AND (&&)
OU	OR (||)

7.Leitura de dados:

Para ler algum número do teclado, utiliza-se o comando *scan*. A linguagem fará com que o próximo valor digitado seja guardado na variável destino. É feito na forma:

scan(*variavel_destino*);

É possível a leitura de mais de um valor num mesmo comando, basta separar as variáveis destino por vírgula.

scan(*variavel1*, *variavel2*, ..., *variavelN*);

8.Impressão de dados na tela:

Para imprimir algo na tela é utilizado o comando *printa*. Para mostrar uma string, usa-se aspas duplas. Para mostrar uma variavel, simplesmente digita-se o nome dela, sem aspas ou qualquer outro símbolo. Tem forma:

printa(*informações_desejadas*);

Exemplos	**Saída**
printa("O valor da variável a é " a);	O valor da variável a é 0.0
printa(a " " b);	0.0 0.0
printa("a + b = " resp);	a + b = 0.0

A função só funciona com uma string(entre aspas) ou um nome de variável, não sendo possível qualquer tipo de operação dentro dela. Existem alguns símbolos especiais que devem ser escapados com \ para serem exibidos. Eles são:

Comando	**Saida**
\n	Quebra de linha
\#	#
\"	"
\;	;