TEORIA DE CONTROLADOR LÓGICO PROGRAMÁVEL

Programação Instruction List (IL) – 1.a Parte

Prof. Dr. Cesar da Costa

E-mail: ccosta@ifsp.edu.br

Site: www.professorcesarcosta.com.br

INSTITUTO FEDERAL
SÃO PAULO
Campus São Paulo

Lógica de Programação - Fluxograma

- Fluxograma é uma ferramenta utilizada para representar *graficamente* o algoritmo, isto é, a sequência lógica e coerente do fluxo de dados. Composta por desenhos geométricos simples conhecidos como diagrama de bloco.

Lógica de Programação - Fluxograma

❖Diagrama de Blocos

- Ferramenta utilizada para representar o *método* do fluxograma, isto é, o que significa cada símbolo geométrico utilizado no fluxo, sua função, sempre desenvolvida no nível de detalhe necessário.

Lógica de Programação - Fluxograma

❖Simbologia

TERMINAL

Indica o inicio ou o fim do fluxo de um programa.

SETA

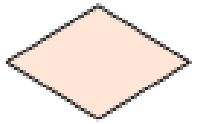
Indica o sentido do fluxo de dados, serve exclusivamente para ligar os diagramas.

Lógica de Programação - Fluxograma

❖Simbologia

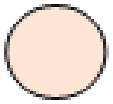
PROCESSAMENTO

Indica cálculo, atribuições ou manipulação de dados.



DECISÃO

Indica tomada de decisão, separação de fluxo de dados.



CONECTOR

Indica conexão de fluxo na página

Lógica de Programação - Fluxograma

❖Simbologia

CONECTOR

Indica conexão de fluxo em outra página

TECLADO

Indica que a informação será digitada via teclado.
Diagrama que representa entrada de dados.

DISPLAY

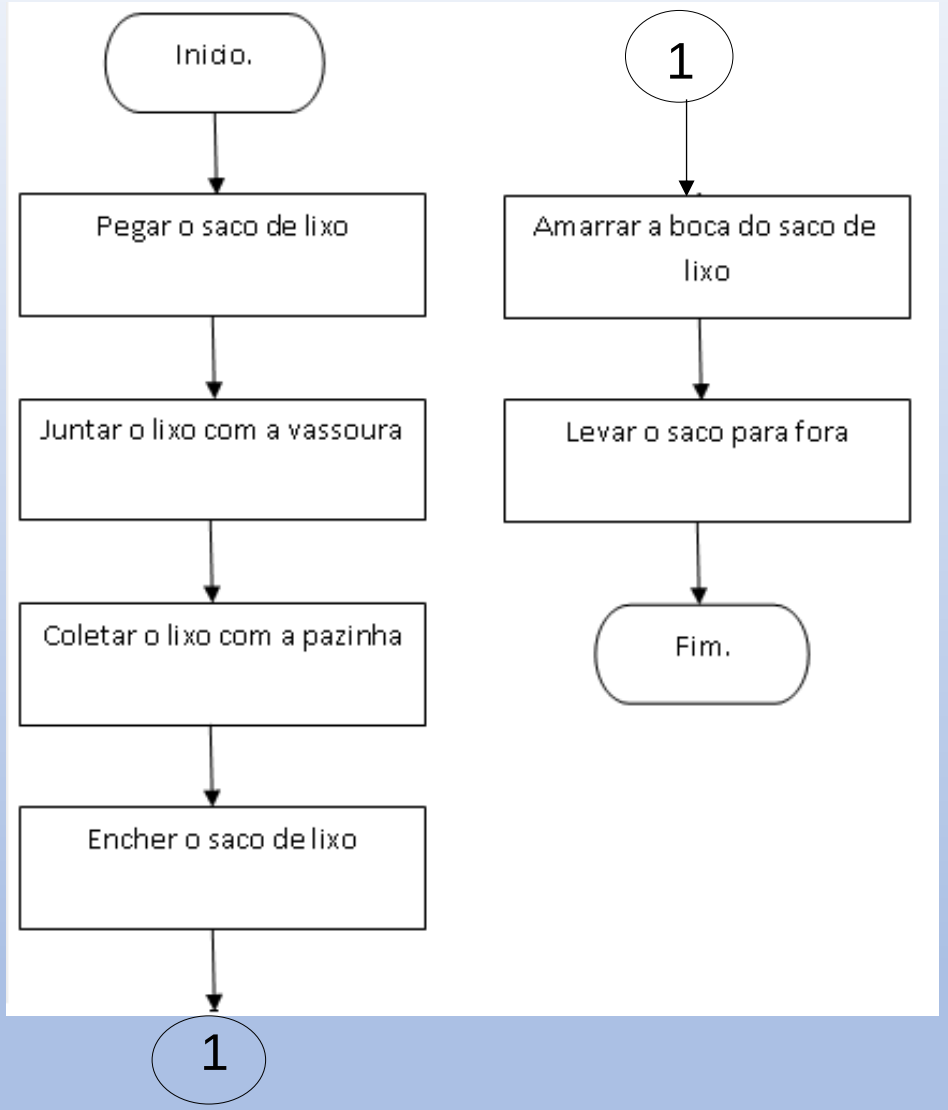
Indica que a informação será exibida no monitor. Diagrama que representa saída de dados.

Lógica de Programação - Fluxograma

1. Exemplo de fluxograma

- No Algoritmo simples teríamos:

1- Pegar o saco de lixo;
2- Juntar o lixo com a vassoura;
3- Coletar com a pazinha;
4- Encher o saco de lixo;
5- Amarrar a boca do saco de lixo;
6- Levar o saco para fora;

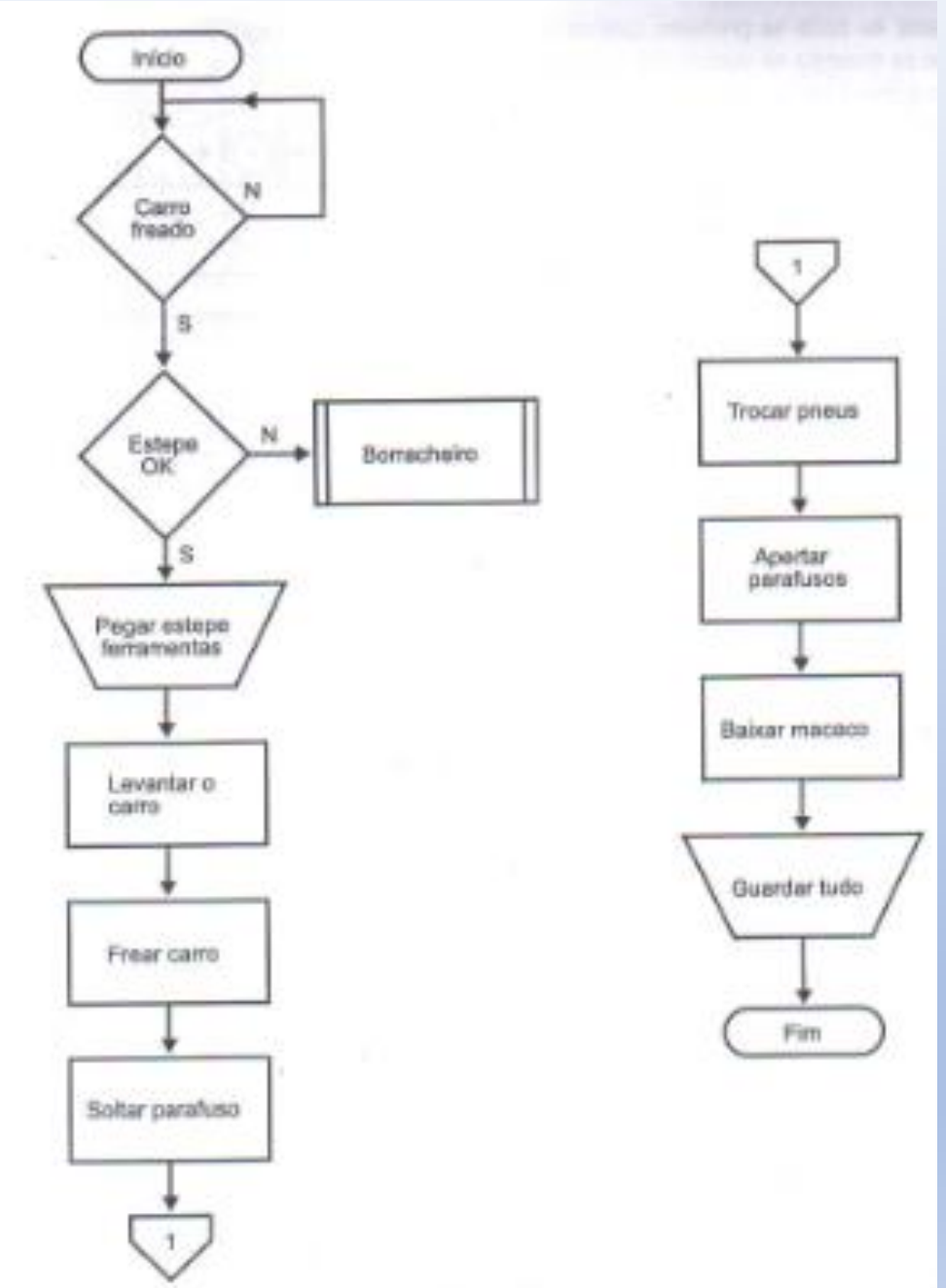

Inicio.
Pegar o saco de lixo
Juntar o lixo com a vassoura
Coletar o lixo com a pazinha
Encher o saco de lixo
1
1
Amarrar a boca do saco de lixo
Levar o saco para fora
Fim.

Lógica de Programação - Fluxograma

- **Exercício 1:**

Desenvolva um fluxograma correspondente a troca de um pneu em uma estrada.

Solução:


Início
Carro freado
N
S
Estepe OK
N
Borracheiro
S
Pegar estepe ferramentas
Levantar o carro
Frear carro
Soltar parafuso
1
1
Trocar pneus
Apertar parafusos
Baixar macaco
Guardar tudo
Fim

Exercício 2:

O sistema de corte se processa quando o carrinho se encontra na posição C2 e o tubo alcança a chave fim de curso C1. O carrinho, por meio de um acionamento, atinge a velocidade v em C3, quando a morsa fecha e a serra circular baixa, serrando o tubo (a serra circular funciona constantemente).

O carrinho retorna à sua posição inicial e, antes de alcançar esta posição, o seu acionamento é desligado por C7, atingindo pela inércia a chave C2.

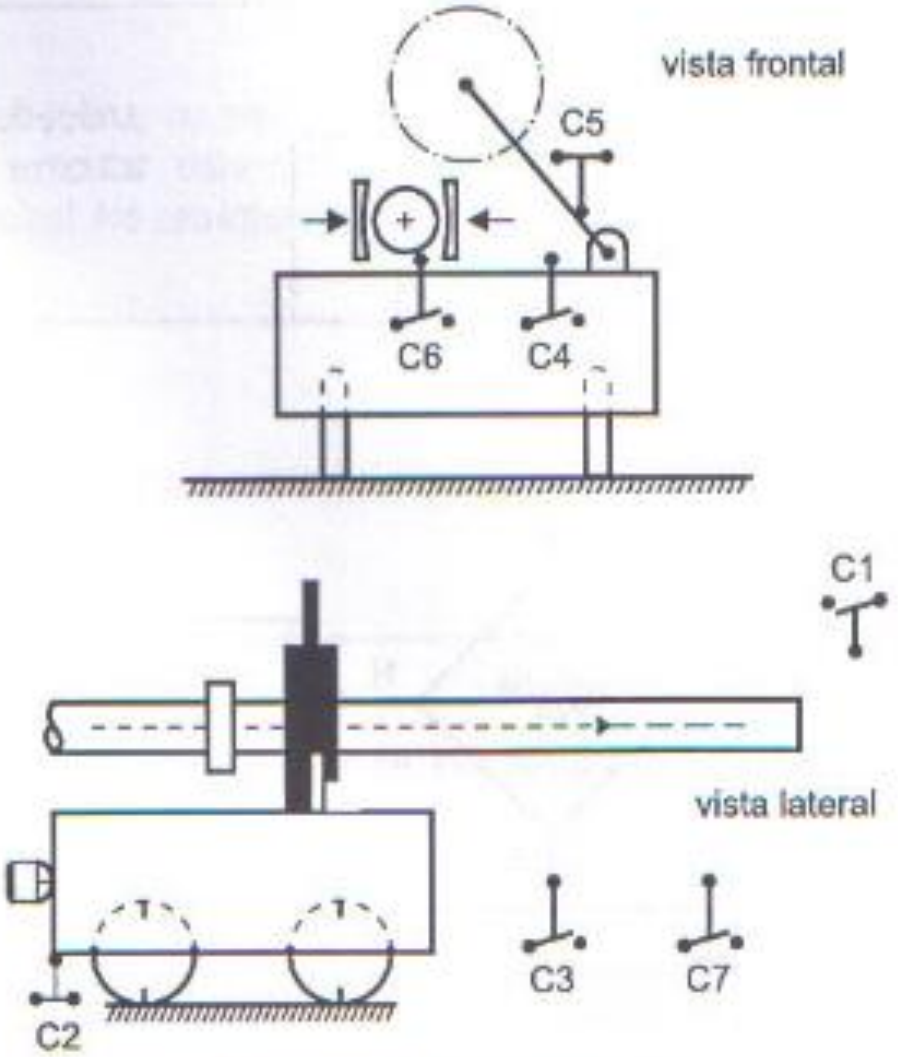


Figura 8.4 - Sistema de corte.

- Desenvolver o fluxograma para a automação do processo.

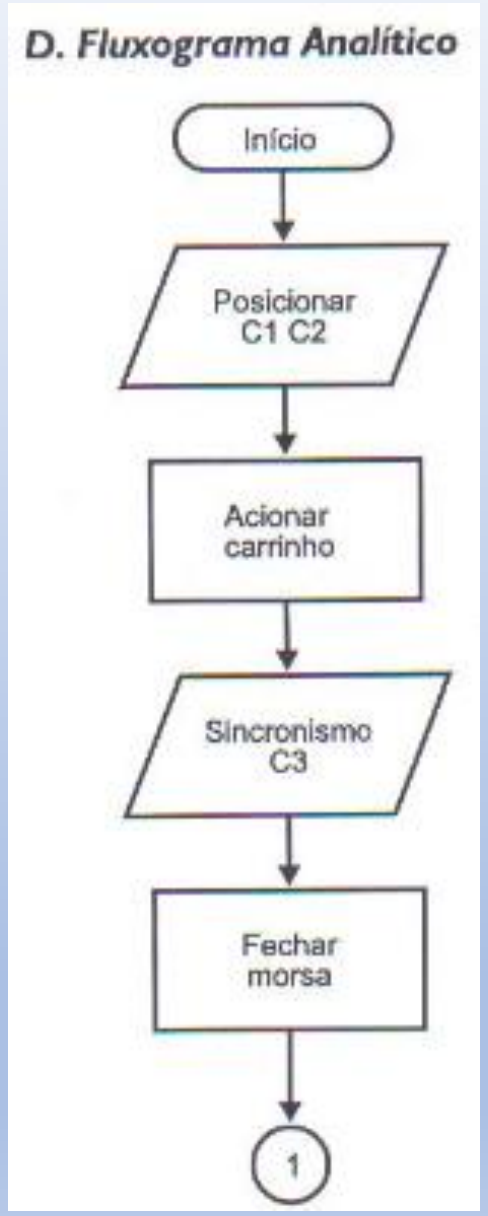
B. Análise

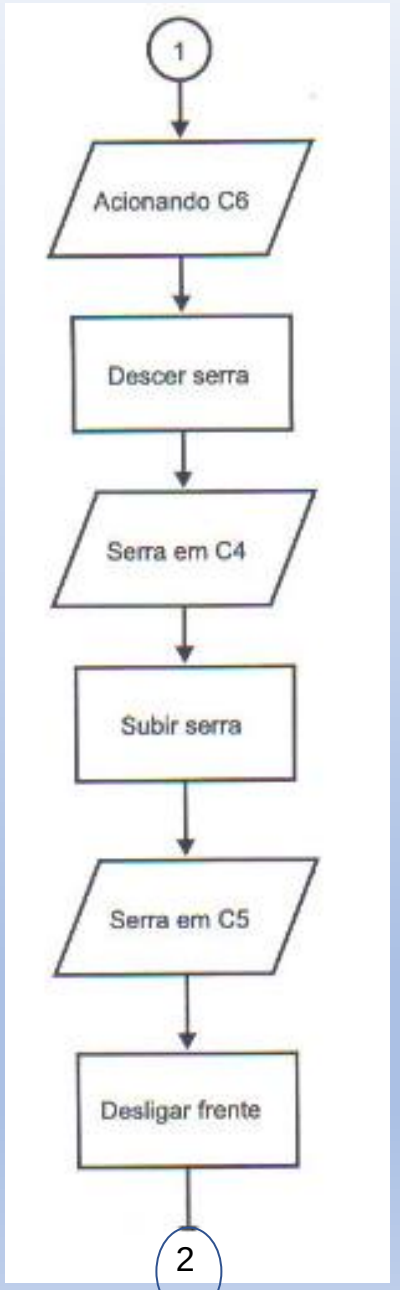
- Posicionamento do tubo - C1;
- Posição inicial do carrinho - C2;
- Sincronismo do carrinho com o tubo - C3;
- Fechar a morsa - C6;
- Baixar serra - C4;
- Subir a serra - C5;
- Retornar carrinho - C7.

C. Algoritmo

1. Posicionamento do tubo em C1 e posição do carrinho em C2;
2. Acionar carrinho para frente;
3. Sincronizar carrinho com o tubo chave C3;
4. Fechar morsa até C6;
5. Descer serra até C4;
6. Subir serra até C5;
7. Desligar carrinho para frente;
8. Abrir morsa:

D. Fluxograma Analítico





Atividade 3:

1) O desenho a seguir representa uma instalação para carga e descarga de vagões, que deverá ser automatizada. O cliente enviou as especificações do projeto, que são apresentadas a seguir. Elabore um fluxograma functional.

1. A chave S1 liga o sistema, a chave S2 desliga o sistema e H1 indica o sistema ligado;
2. Quando o vagão está posicionado o sensor S3 é atuado;
3. Quando o vagão é posicionado, ele é travado por Y2;
4. Após o vagão ser travado, a esteira (motor M1) é ligada e inicia o ciclo de enchimento do vagão;
5. A válvula YA abre o silo;
6. O pressostato P por meio do sensor S4, quando é atuado, indica que o vagão está cheio;
7. Então, fecha a válvula Yb, fechando o silo;
8. Espera 10 segundos;
9. Destrava o vagão, liberando Y2;
10. Supervisiona o sensor S3 por 20 segundos, se ele não for acionado, ultimo vagão, desliga a esteira Motor M1.

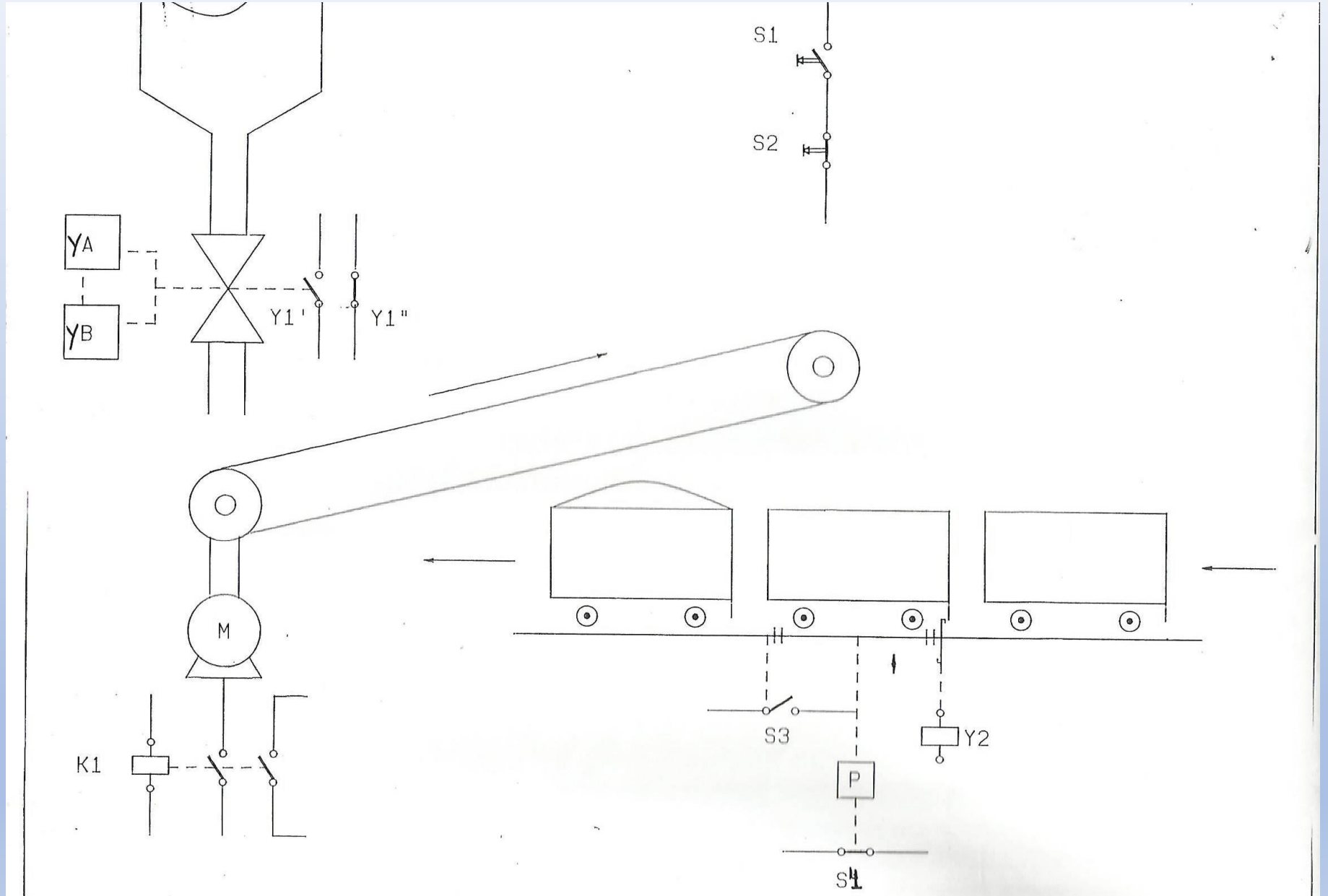


FIGURA 2.1 – INSTALACAO PARA CARGA E DESCARGA DE VAGOES

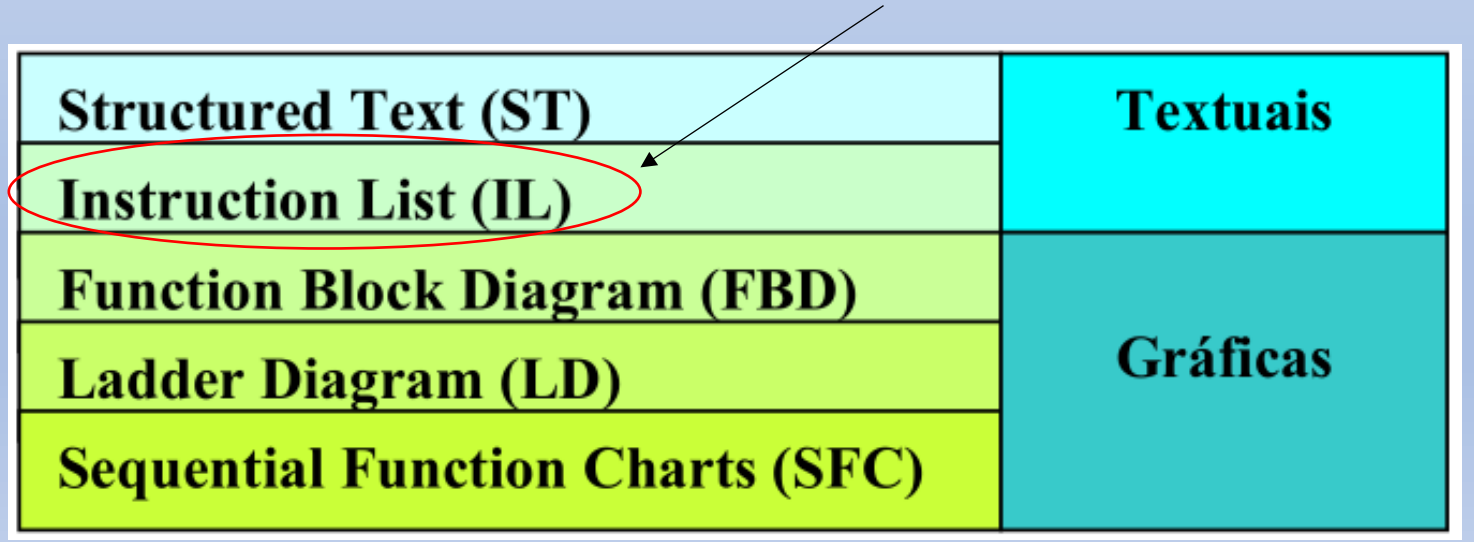
QUADRO RESUMO

ITEM	DESCRIÇÃO	TIPO	SIMBOLO
01	CHAVE PULSO	ENTRADA	S1
02	CHAVE PULSO	ENTRADA	S2
03	SENSOR 24 VDC	ENTRADA	S3
04	CONTATO (N /F)	ENTRADA	S4
05	CONTATO (N /A)	ENTRADA	K1
06	CONTATO (N /A)	ENTRADA	Y1
07	CONTATO (N /F)	ENTRADA	Y1"
08	LAMPADA 110 VCA	SAIDA	H1
09	CONTATOR 110 VCA	SAIDA	K1
10	VALVULA 110 VCA	SAIDA	Y1 (A)
11	VALVULA 110 VCA	SAIDA	Y1 (B)
12	VALVULA 110 VCA	SAIDA	Y2

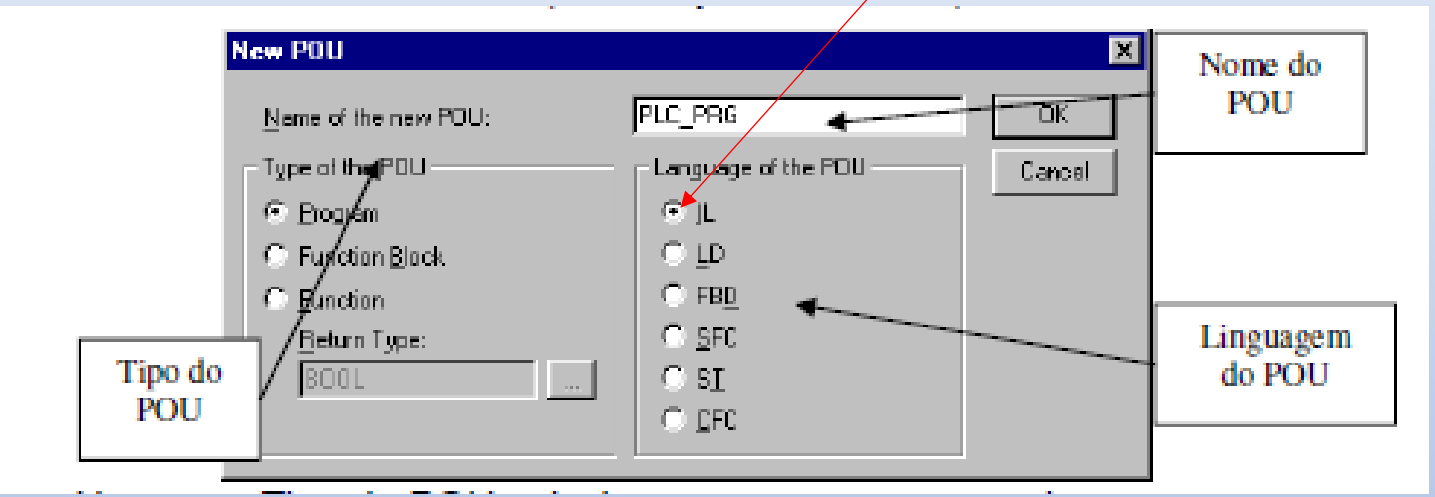
7 entradas	24 VDC
5 saidas	110 VAC

Linguagem de Programação Instruction List

- A Organização Internacional IEC (*International Electrotechnical Committee*) é a responsável pela padronização das linguagens de programação para CLP, sendo a norma IEC 61131-3 *Programing Languages* a responsável pela classificação dessas linguagens.



Software CODESYS - Criando um novo Projeto



- Na opção Tipo do POU seleciona-se criar um programa, em *Linguagem em Lista de Instruções* – Instruction List *(IL)*

Linguagem em Lista de Instruções (IL)

- A linguagem de programação em Lista de Instruções, ou Instruction List (IL) é uma linguagem de programação textual de baixo nível semelhante ao assembly. Muito utilizada para resolver problemas simples e pequenos.

- Como o próprio nome diz, o programa se resume a uma listagem de comandos que o CLP executa um atrás do outro. Sendo uma linguagem menos amigável e pouco flexível é também utilizada para produzir códigos otimizados em programas, ou trechos onde a performance da execução é crítica.

Linguagem em Lista de Instruções (IL)

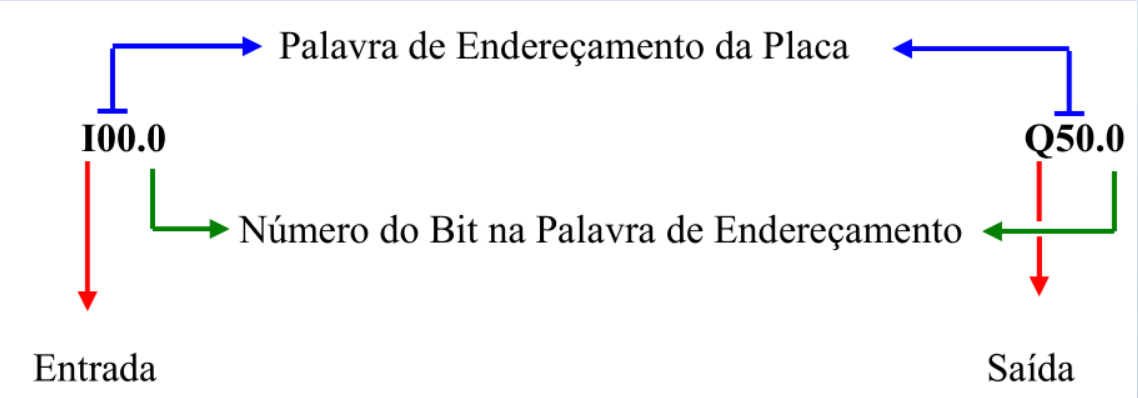
- Esta linguagem é a preferida por quem vem do setor de Informática.

  - Exemplo de programa em IL (Instruction List)

```
0001   LD    I1   (*LE ENTRADA 1*)
0002   ANDN  I2   (*LÓGICA E NÃO*)
0003   ST    Q1   (*COLOCA RESULTADO NA SAÍDA*)
0004
0005   LD    I3
0006   OR    ( IW4
0007   GE    500 )
0008   ST    Q2
0009
```

Linguagem em Lista de Instruções (IL)

- Exemplo de um programa em Ladder.



- Seu equivalente em Lista de Instruções.

LD	I00.0	→ carrega acumulador com valor de entrada	A = I00.0
ST	Q50.0	→ Armazena valor do acumulador na saída	Q50.0 = A

❑ Introdução

- A linguagem de Lista de Instruções (IL), também comumente referenciada pelo nome original da língua inglesa, Instruction List (IL), define mnemônicos como na linguagem assembly, utilizada nos microprocessadores e microcontroladores.

- Os mnemônicos representam operações lógicas booleanas, comparações e comandos de transferência de dados.

- Por exemplo: LD, LDN, AND, OR, GE, MOVE, etc.

❑ Introdução

❖ Em relação as demais linguagens, apresenta as seguintes características:

❑ **Vantagens**

- Correspondência entre comandos da linguagem e as instruções assembly do CLP, facilitando a estimativa do tempo de execução do programa.

- Documentação mais compacta do que a equivalente com relés.

❑ Introdução

❑ Desvantagens

- Necessidade de familiarização do operador com álgebra booleana;
- Necessidade de uma certa noção de programação em assembly;
- É normalmente difícil e trabalhoso realizar eventuais alterações no código já implementado.

❑ Introdução

- A IL é uma linguagem ideal para resolver problemas simples e pequenos em que existem poucas quebras no fluxo de execução do programa. É, portanto, particularmente adequada para CLPs de pequeno porte

- Essa linguagem pode ser usada para descrever o comportamento de:
  - Funções;
  - Blocos de funções;
  - Programas

❑ Princípios Básicos

❑ A linguagem de Lista de Instruções é semelhante ao código assembly com comandos load e store. Ela usa o conceito de acumulador para armazenar os resultados intermediários.

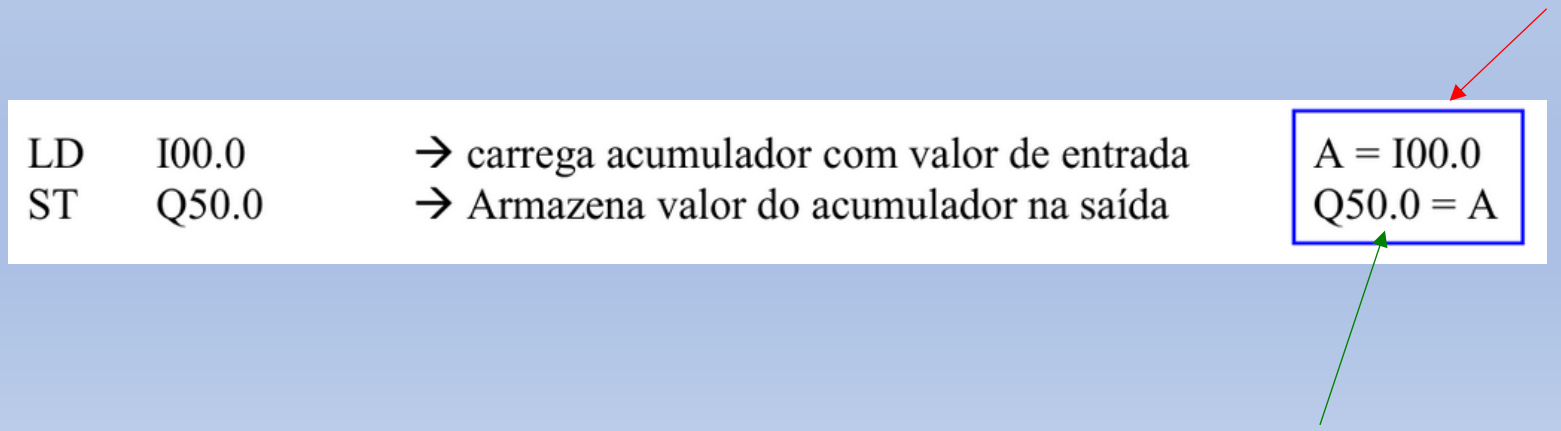
- Cada instrução utiliza ou modifica o valor de um único registrador denominado registro de resultado ou acumulador.

Princípios Básicos

- As instruções são executadas no conteúdo do acumulador.

- O operador indica o tipo de operação a ser feito entre o resultado atual contido no acumulador e o operando.

- O resultado da operação é armazenado no próprio acumulador.



❑ Sintaxe

❑ As regras principais de formação de um programa em linguagem de Lista de Instruções são:

- Cada instrução deve começar em uma nova linha;

- Cada instrução pode ser precedida por um rótulo (elemento opcional) que é indicado com um nome seguido de dois pontos “:”;

❑ Sintaxe

- Uma instrução é composta de operador e operandos (instrução = operador + operandos);
- O operador pode ou não incluir um modificador;
- Caso seja necessária a inclusão de mais de um operando, estes devem ser separados por vírgulas;

❑ Sintaxe

- Se for desejada a inclusão de comentário, ele deve ser o último elemento da linha;
- Um comentário é iniciado pela sequência de caracteres (* e terminado pela sequência*);
- Linhas em branco podem ser inseridas entre instruções;
- Um comentário pode ser colocado em linha sem instruções.

❑ Sintaxe

❖ A estrutura pode ser verificada na Figura 1

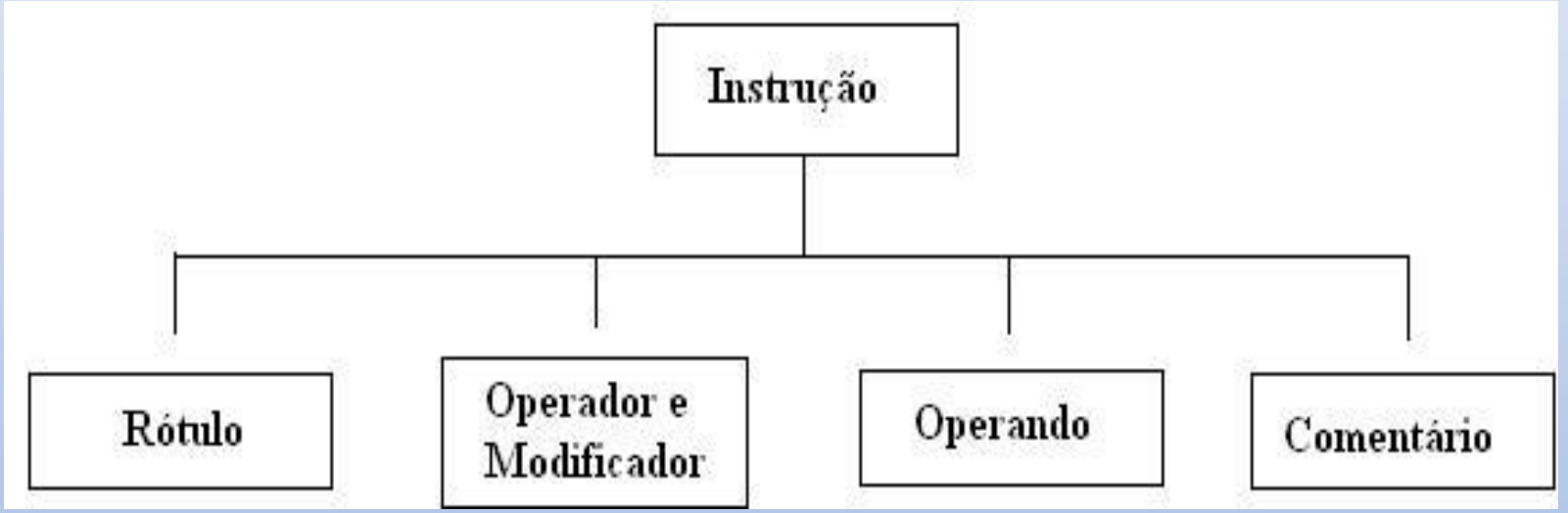


Figura 1 – Estrutura de uma linha de instrução da linguagem IL.

❑ Sintaxe

➢ Exemplo 1

Rótulo	Operador	Operando	Comentário
Início	LD	% IX1	(* botão pressionado? *)
	AND	% MX3	(* comando válido*)
	ST	% QX1	(* liga o motor*)

❖ O valor da entrada % IX1 é carregado para o acumulador;

❖ Em seguida é feita uma operação lógica AND entre o conteúdo do acumulador e da memória % MX3;

❖ O resultado é transferido para a saída %QX1.

❑ Rótulo (etiqueta)

❖ Cada instrução pode ser precedida por um rótulo, que é um nome seguido do caractere “:”. Ele também pode ser colocado em uma linha que não contenha nenhuma instrução. Os rótulos são utilizados como operandos em certas instruções tais como saltos. A sua nomenclatura deve obedecer a seguintes regras:

- O comprimento não deve exceder 16 caracteres.
- O primeiro caractere deve ser uma letra.
- Os caracteres restantes podem ser letras, números ou símbolo “_” (sublinhado).
- Não pode haver no mesmo programa dois rótulos iguais.

❑ Modificadores de instruções

- A lista a seguir representa os modificadores permitidos para as instruções da linguagem. Devem ser anexados após o nome da instrução, sem caractere separador.

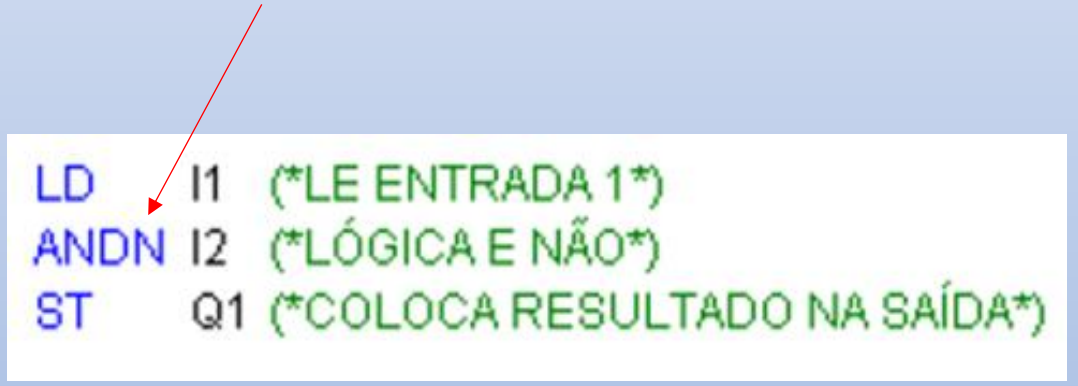
N ⇨ inversão lógica do operando;

(⇨ operação adiada;

C ⇨ operação condicional.

❑ Modificadores de instruções

❖ O Modificador “N” indica que o operando deve ser invertido antes de ser utilizado pela instrução. Por exemplo, a instrução ANDN % I2 é interpretada como “o conteúdo de %I2 é invertido e com o valor de resultado é feita na operação lógica AND com o acumulador”.



```
LD    I1  (*LE ENTRADA 1*)
ANDN  I2  (*LÓGICA E NÃO*)
ST    Q1  (*COLOCA RESULTADO NA SAÍDA*)
```

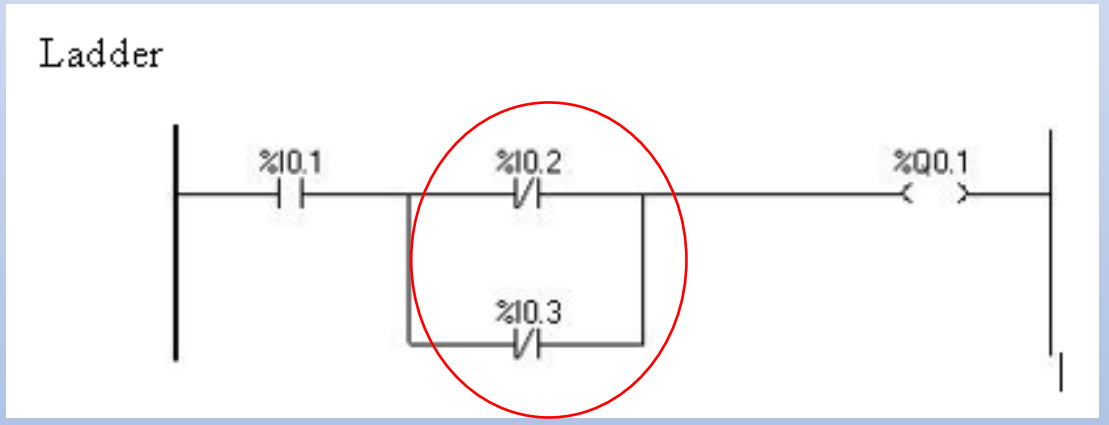
❑ Modificadores de instruções

- O Modificador abrir parênteses "**(**" indica que a avaliação da instrução deve ser adiada até que seja encontrado o próximo fechar parênteses "**)**".

Lista de Instruções

```
LD I.1
AND (N I.2
ORN I.3
)
ST Q.1
```

Ladder



❑ Modificadores de instruções

- O modificador “C” indica que a instrução deve ser executada somente se o conteúdo atual do acumulador for lógico verdadeiro (ou diferente de zero para tipos não booleanos). O modificador “**C**” pode ser combinado com o modificador “N” para indicar que a instrução não deve ser executada, a menos que o resultado seja falso (ou 0 para tipos não booleanos).

Example of an IL program while using some modifiers:

```
LD      TRUE            (*load TRUE in the accumulator*)
ANDN    BOOL1           (*execute AND with the negated value of the BOOL1 variable*)
JMPC    label           (*if the result was TRUE, then jump to the label "label"*)
LDN     BOOL2           (*save the negated value of *)
ST      ERG             (*BOOL2 in ERG*)
label:

LD      BOOL2           (*save the value of *)
ST      ERG             (*BOOL2 in ERG*)
```

A tabela 1 apresenta comandos da linguagem de Lista de Instruções.

Tabela 01 - Lista de instruções:

	Operador	Modificador	Operando	Comentário
Operações Básicas	ADD	(	Qualquer	Adição
	DIV	(	Qualquer	Divisão
	MUL	(	Qualquer	Multiplicação
	SUB	(	Qualquer	Subtração
Funções	LD	N	Qualquer	Carrega operando no acumulador
	ST	N	Qualquer	Armazena acumulador no operando
Funções Lógicas	&	N, (	BOOL	E lógico
	AND	N, (	BOOL	E lógico
	OR	N, (	BOOL	OU lógico
	XOR	N, (	BOOL	OU – Exclusivo
	R		BOOL	Reseta operando para False
	S		BOOL	Seta operando para True

Modificador

N – Nega um valor Booleano.

(- adia uma operação.

C – denota jump condicional (só pode ser usado com JUMP).

A tabela 2 apresenta comandos da linguagem de Lista de Instruções.

Tabela 02 - Desvios e comparações:

Operado	Modificador	Operando	Comentário
EQ	(	Qualquer	Comparação de igual
GE	(	Qualquer	Comparação maior ou igual
GT	(	Qualquer	Comparação maior que
LE	(	Qualquer	Comparação menor ou igual
LT	(	Qualquer	Comparação menor que
NE	(	Qualquer	Comparação se diferente
JMP	C, N	Label	Salto
CALL	C, N	Nome	Chamada de função
RET	C, N		Retorno da função
( )			Prioridade

Modificador

N – Nega um valor Booleano.

(- adia uma operação.

C – denota jump condicional (só pode ser usado com JUMP).

A tabela 3 apresenta comandos da linguagem de Lista de Instruções.

Tabela 03 – Operadores de Blocos de Função

Operador	**Bloco de Função**	**Comentário**
S1, R	Biestável SR	Seta e Reseta o Biestável SR
S, R1	Biestável RS	Seta e Reseta o Biestável RS
CLK	R_Trig, detector de borda de subida	Entrada de clock de borda de subida do bloco lógico
CLK	F_Trig, detector de borda de descida	Entrada de clock de borda de descida do bloco lógico
CU, R, PV	CTU, contador incremental	Parâmetros de controle para o contador incremental CTU; CU incrementa, R reset e PV carrega contador.
CD, LD, PV	CTD, contador decremental	Parâmetros de controle para o contador decremental CTD; CD decrementa, LD carrega; e PV carrega contagem mínima.
CU, CD, R, LD, PV	CTUD, contador universal	Parâmetros de controle para o contador universal CTUD.
IN, PT	TP, temporizador de pulso	Paramentos de controle para o timer de pulso. IN inicia temporização; PT seta o tempo de pulso.
IN, PT	TON, temporizador de atraso de subida.	Paramentos de controle para o timer de atraso de subida. IN inicia temporização; PT seta o tempo de pulso.
IN, PT	TOF, temporizador de atraso de descida	Paramentos de controle para o timer de atraso de descida. IN inicia temporização; PT seta o tempo de pulso.

❑ Operador LD, LDN

- Mnemônico da palavra inglesa LOAD
- **Operação**: carrega um valor para o acumulador.
- **Modificador**: N
- **Operando**: expressão constante.

❑ Operador ST, STN

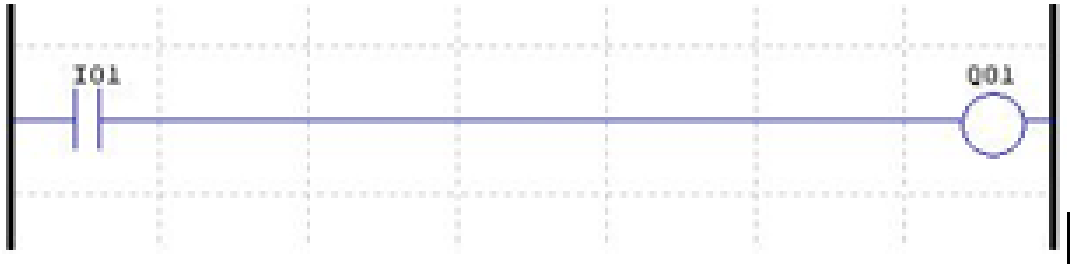
- Mnemônico da palavra inglesa **STORE**
- **Operação**: transfere o conteúdo do acumulador para uma variável.
- **Modificador**: N
- **Operando**: variável interna ou de usuário.

- **Exemplo 1:** Implemente um programa em Ladder e em Lista de instruções (I.L) que tenha a tarefa de acender uma lâmpada L sempre que a chave CH for fechada.

Solução:

Um programa simples no qual a atuação de **uma entrada** causa a atuação de **uma saída**, isto é, utiliza as duas instruções principais que são leitura de variável (**LD**) e atribuição de valor (**ST**), e terá o seguinte aspecto , em Ladder e IL.

Ladder



Lista de Instruções

```
LD I1

ST Q1
```

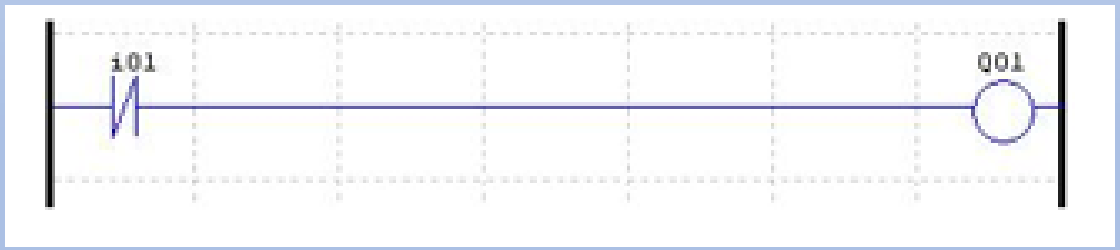
- **Exemplo 2:** Para um contato A do tipo NF, é preciso fazer a leitura de uma variável negada **LDN**, conforme visto a seguir:

Solução:

Lista de Instruções

```
LDN   I1

ST    Q1
```

Ladder



- Neste caso, a partir da instrução **LDN**, o processador efetua a leitura de complemento lógico de **I1** e atribui o valor lido à saída **Q1.**

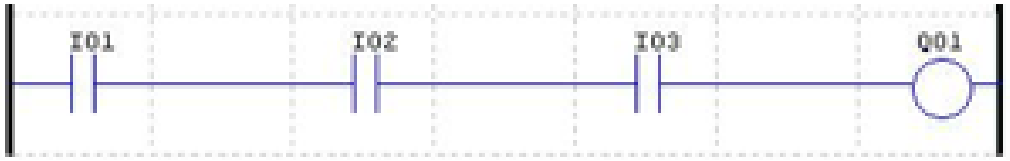
Exemplo 3: Operação E (AND) – Dada a equação lógica Q1= I1**.** I2**.** I3, implemente a função lógica no diagrama Ladder e em Lista de Instruções.

Solução:

Lista de Instruções

```
LD   I1

AND I2

AND I3

ST Q1
```

Ladder



Exemplo 4: Operação OU (OR) – Dada a equação lógica, implemente a função lógica no diagrama Ladder e em Lista de Instruções.

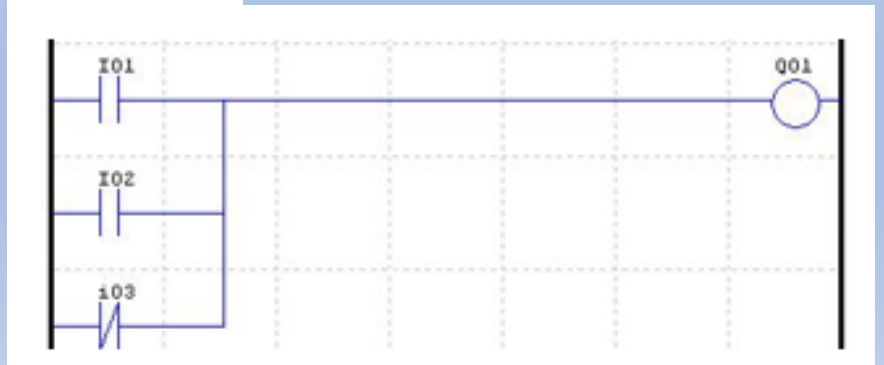
$$Q_1 = I_1 + I_2 + \bar{I}_3$$

Solução:

Lista de Instruções

```
LD I1
OR I2
ORN I3
ST Q1
```

Ladder



❑ Operador S

- É uma instrução de memorização. A letra S é um mnemônico da palavra inglesa **set.**

- **Operação**: Força uma variável booleana a ir para o estado lógico 1 se o acumulador estiver com o valor VERDADEIRO (nível Lógico 1). Nenhuma operação é realizada se o operador estiver com o valor lógico FALSO (nível lógico 0).

- **Modificador**: Não tem

- **Operando**: variável booleana interna ou de saída

Exemplo 5: Faça o diagrama Ladder e a lista de Instruções correspondentes a dois contatos I1 e I2, NA e NF respectivamente, em paralelo, e um contato I3, NF, em série com ambos. O outro lado do contato I3 está conectado à bobina do tipo set de um relé Q1 de auto retenção.

Solução:

Lista de Instruções

```
LD I1

ORN I2

ANDN I3

S Q1
```

Ladder

❑ Operador R

- É um operador que serve para “limpar” o conteúdo da memória. Faz com que o conteúdo da memória vá para o valor zero. A letra R é um mnemônico da palavra inglesa reset.

- **Operação**: força uma variável booleana a ir para o valor lógico 0 se o valor do acumulador for VERDADEIRO (nível lógico 1). Nenhuma operação é realizada se o valor do acumulador for FALSO (nível lógico 0).

- **Modificador**: Não tem.

- **Operando**: Variável lógica binária interna ou de saída.

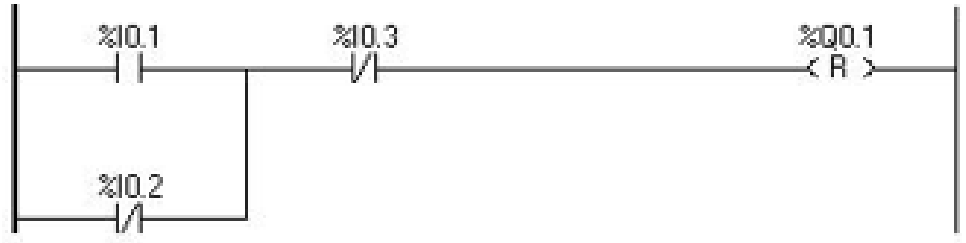
Exemplo 6: Faça o diagrama Ladder e a lista de Instruções correspondentes a dois contatos I.1 e I.2, NA e NF respectivamente, em paralelo, e um contato I.3, NF, em série com ambos. O outro lado do contato I.3 está conectado à bobina do tipo reset de um relé Q.1 de auto retenção.

Solução:

Lista de Instruções

```
LD I.1

ORNI.2

ANDNI.3

R Q.1
```

Ladder



Obs:

1 – A instrução R (reset) sempre opera com a instrução S (set) e vice-versa.

❑ Operações adiadas.

- Como a linguagem I.L só possui um registrador, certas operações podem ser adiadas para alterar a ordem natural da execução das instruções. Os parênteses são utilizados para representar as operações adiadas.

"("  indica que a instrução anterior deve ser adiada;

")"  indica que a operação anteriormente adiada deve agora ser executada.

Exemplo 7: Dada a equação lógica (1), implemente, em Ladder e em Lista de Instruções.

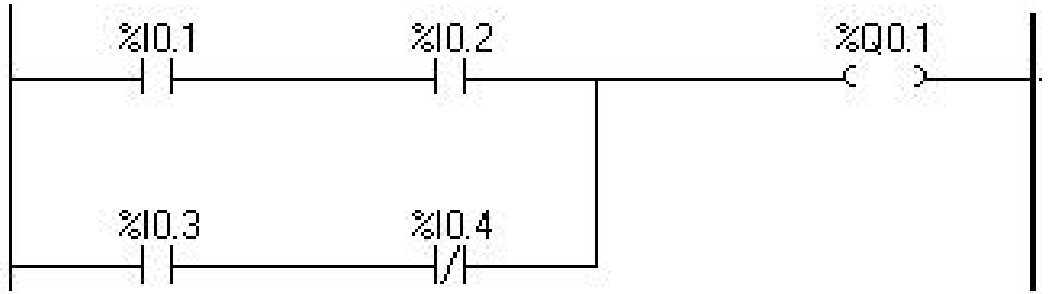
$$Q_1 = (I_1.I_2) + (I_3.\bar{I}_4) \qquad (1)$$

Solução:

Lista de Instruções

```
LD I.1

AND I.2

OR ( I.3

ANDN I.4

)

ST Q.1
```

Ladder



Exercicios Propostos:

- Implemente as seguintes equações lógicas em lista de instruções:

a) $Q_1 = (I_1 + \bar{I_2}) + (\bar{I_3} + I_4)$

b) $Q_1 = I_1 . I_2 + \bar{I_3} . (\bar{I_4} + I_5)$

❑ Mnemônicos de alguns fabricantes

❖ Antes do surgimento IEC 61131-3, cada fabricante utilizava seu próprio conjunto de mnemônicos. Embora muito parecidos entre si, eram diferentes de um fabricante para outro. Assim, antes de implementar um programa em linguagem de I. L em uma aplicação real, deve-se realizar um estudo detalhado do manual do fabricante para determinar os mnemônicos equivalentes à norma IEC 61131-3. A título de exemplo, na tabela 2 são fornecidos os mnemônicos de alguns fabricantes.

IEC 61131-3	Mitsubishi	OMRON	SIEMENS S7-200
LD	LD	LD	LD
LDN	LDI	LD NOT	LDN
AND	AND	AND	A
ANDN	ANI	AND NOT	AN
OR	OR	OR	O
ORN	ORI	OR NOT	ON
ST	OUT	OUT	=

❑ Temporizadores

❖ Implementação de um temporizador TON no CLP que segue a norma IEC 61131-3.

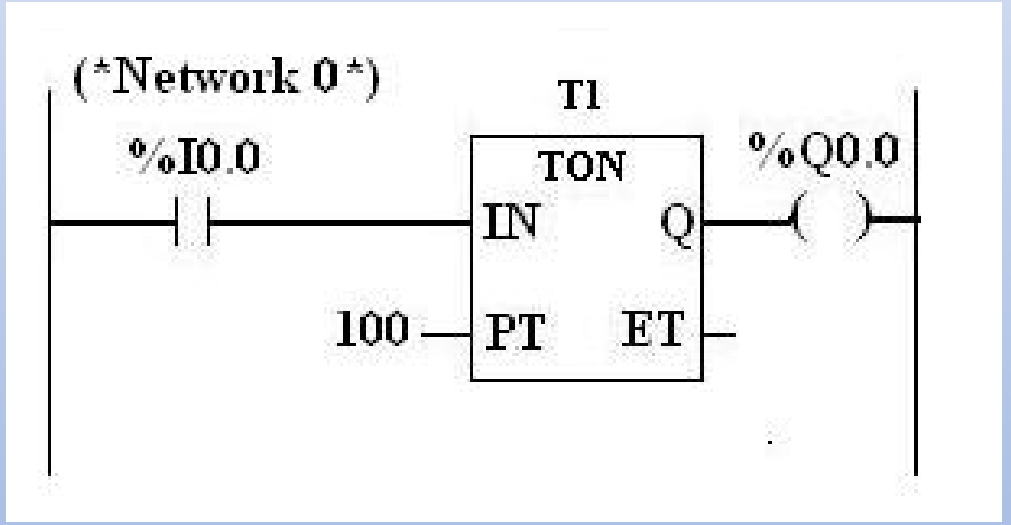
Lista de Instruções

(*Network 0*)

LD %I0.0

TON T1, 100

ST %Q0.0



❑ Contadores

- ❖ Os blocos podem ser chamados de várias maneiras e os fabricantes têm alguma liberdade de implementação.

- ❖ Pela norma IEC 61131-3 as funções podem ser chamadas diretamente, sem a necessidade de um operador que as preceda.

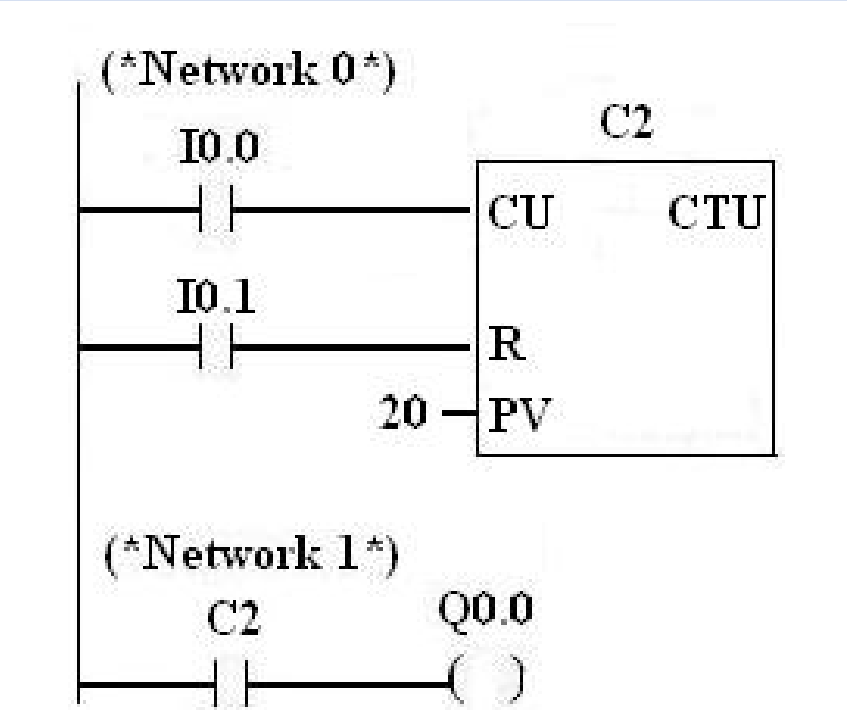
- ❖ De fato, o nome da função pode ser considerado um operador. Os parâmetros passados são: o endereço do contador, o contato que está ligado à entrada reset e o valor de PV.

9 – Contadores

- Implementação de um contador crescente em um CLP que segue a norma IEC 61131-3.

Lista de Instruções

```
(*Network 0*)

LD I0.0

LD I0.1

CTU C2, 20

(*Network 1*)

LD C2

ST % Q0.0
```



Conclusões

Referência

http://professorcesarcosta.com.br/upload/imagens_upload/Apostila%20-%20CLP%20-%20Lista%20de%20instru%C3%A7%C3%B5es.pdf

http://professorcesarcosta.com.br/upload/imagens_upload/Apostila%20Codesys%20Avancada.pdf

http://professorcesarcosta.com.br/disciplinas/n7clpteclp

http://professorcesarcosta.com.br/upload/imagens_upload/Apostila_do_Curso_Clp-1.pdf

INSTITUTO FEDERAL
SÃO PAULO
Campus São Paulo