

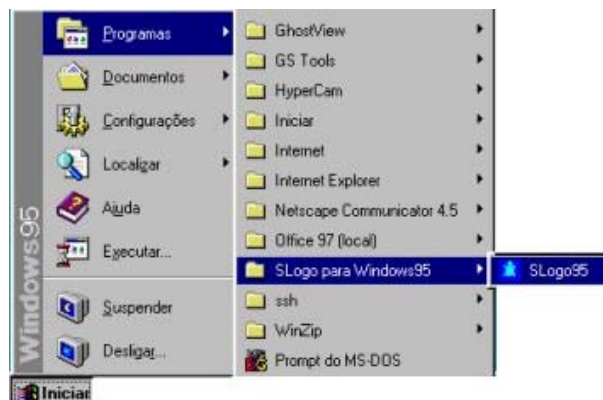
O que é Logo?

*É uma linguagem de programação que foi desenvolvida para ser utilizada com finalidades educacionais. É uma linguagem de propósito geral, isto é, pode ser utilizada em vários domínios de conhecimento. Por ser uma linguagem de programação de propósito geral, permite ao usuário resolver problemas de vários domínios do conhecimento: música, artes, matemática, línguas, etc.. Apresentaremos uma versão da linguagem Logo denominada **SuperLogo** para **Windows 95**.*

Conhecendo o Ambiente Logo*

Neste material será utilizado o ambiente *SuperLogo*¹ que é uma das versões da Linguagem Logo em português.

Abrir o Logo:

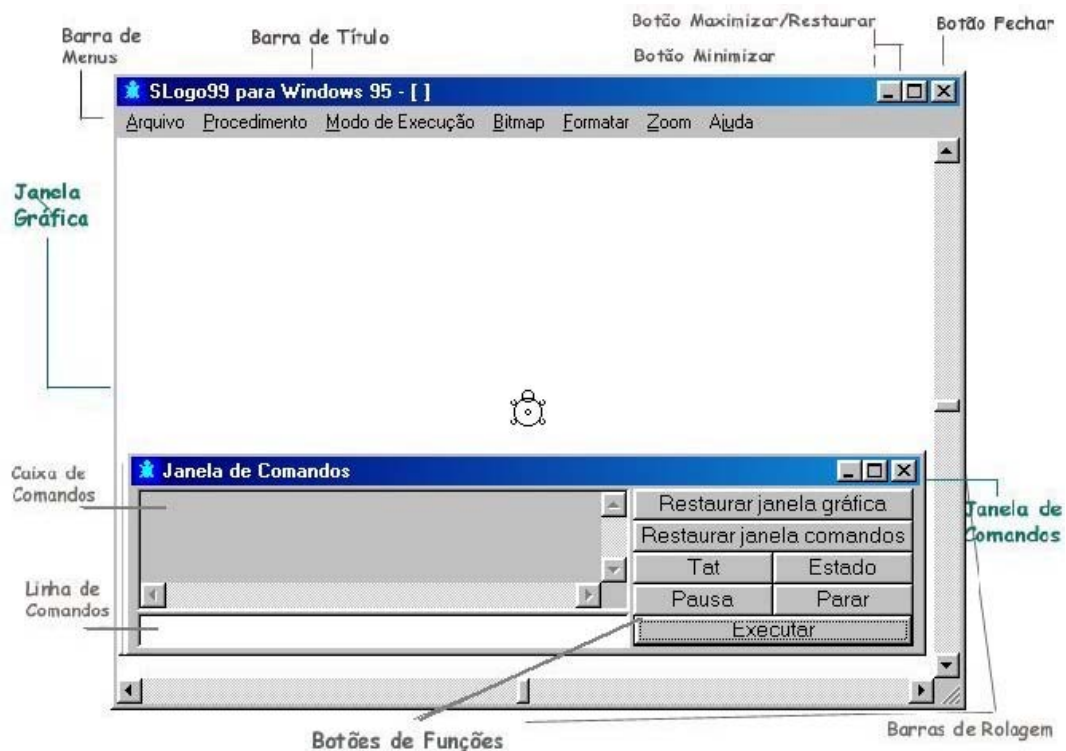


No menu **Iniciar**, selecione **Programas**, opção **SuperLogo**.

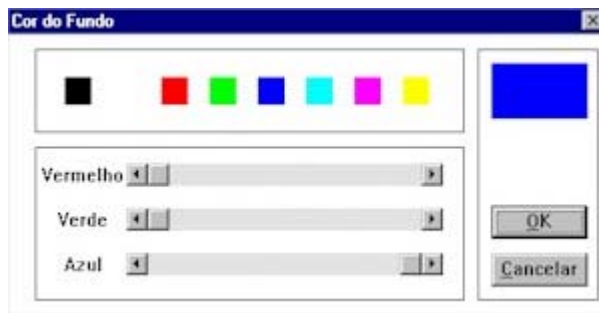
* Este capítulo tem a co-autoria de Maria Cecília Martins.

¹ Esta versão da linguagem Logo vem sendo desenvolvida pela Universidade de Berkley (USA) sob coordenação de George Mills. No Brasil, o Núcleo de Informática Aplicada à Educação da Unicamp é responsável pela sua tradução para o português. A versão traduzida pode ser encontrada em <http://www.nied.unicamp.br>

Serão abertas duas janelas, a **Janela Gráfica** e a **Janela de Comandos**, que constituem o ambiente do *SuperLogo*. No centro da Janela Gráfica aparece a figura de uma tartaruga, um cursor gráfico que, a partir de comandos específicos movimenta-se na tela permitindo a construção de desenhos. Esta janela além de permitir a execução dos desenhos elaborados pelo usuário permite acessar o menu de opções do ambiente. A Janela de Comandos permite ao usuário digitar as instruções a serem executadas pelo *Logo* e acionar os botões do ambiente. As duas janelas podem ser **arrastadas**, **maximizadas** e **minimizadas** mas somente a Janela Gráfica pode ser fechada.

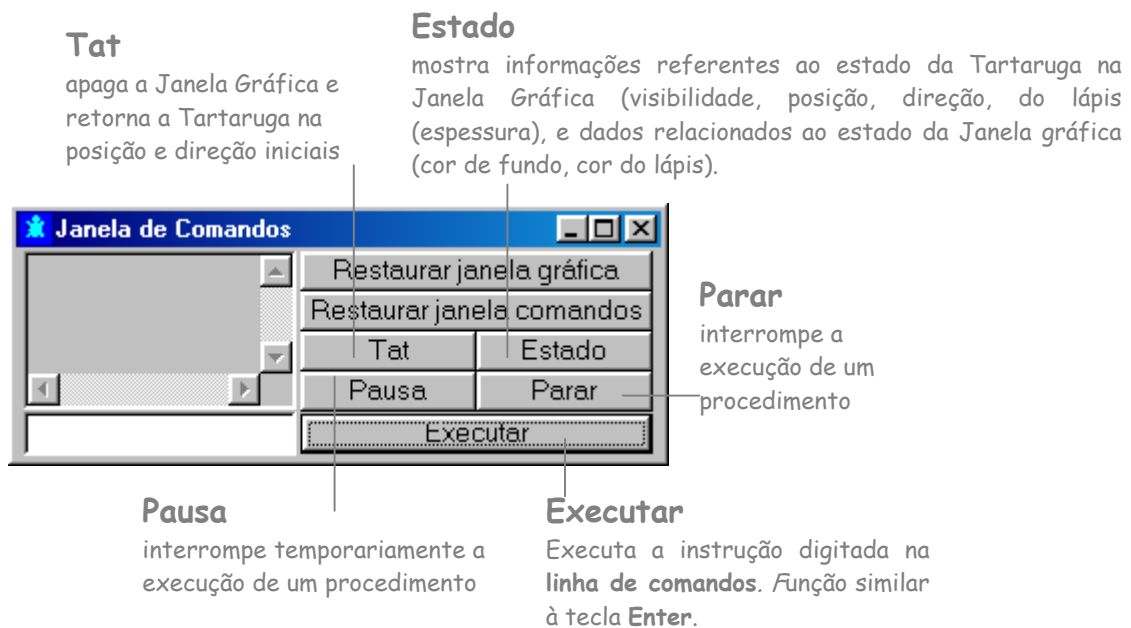


O ambiente dispõe de alguns recursos que podem ser ativados pelo menu de opções e por meio dos botões existentes da Janela de Comandos. Por exemplo, para mudar a cor do fundo da tela aciona-se o menu **Formatar** da Janela Gráfica, opção **Cor do Fundo**.



A seleção da cor é feita com o mouse. A cor ativada fica aparente à direita da caixa de cores. A seleção de cores pode também ser programada, isto é, pode ser efetuada por meio de um comando específico (ex: mudectf "azul")

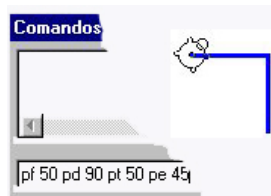
Os botões da Janela de Comandos podem ser acessados clicando o mouse sobre eles. Por exemplo:



*Consultar o item **Índice** do menu **Ajuda** sempre que houver alguma dúvida. Desta forma pode-se obter maiores detalhes sobre os componentes do ambiente: lista dos comandos **Logo**, menu de opções, janelas e botões.*

Conhecendo a Linguagem Logo

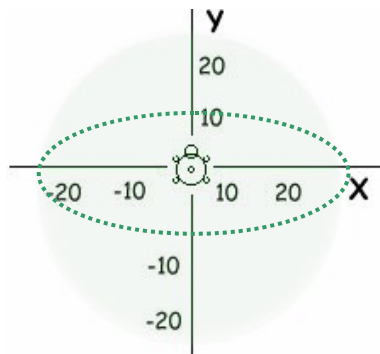
Há quatro comandos básicos que permitem movimentar a tartaruga. Os comandos **para frente (ou pf)** e **para trás (ou pt)** fazem a tartaruga andar e os comandos **para direita (ou pd)** e **para esquerda (ou pe)** giram a tartaruga na tela.



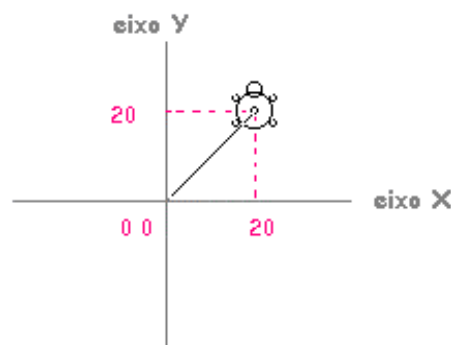
Para usar estes comandos é necessário especificar o número de passos ou o grau do giro. Por exemplo: `pf 50 pd 90 pt 50 pe 45`.

Os comandos **pf** e **pt** alteram a posição da tartaruga e os comandos **pd** e **pe** a sua orientação na tela gráfica.

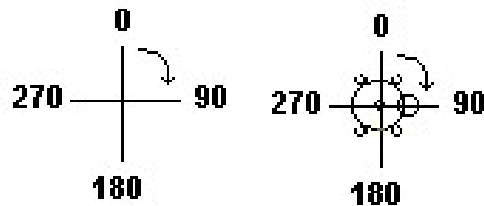
Uma outra maneira de movimentar a tartaruga é usar o sistema de **coordenadas cartesianas** para deslocá-la na tela e usar um “eixo imaginário” (que permite o giro no sentido horário) para alterar sua **orientação**.



Por exemplo, para deslocar a tartaruga para o ponto 20 20 das coordenadas xy pode-se utilizar o comando **mudexy 20 20**.






Supondo-se que a Tartaruga esteja reta, isto é, na sua direção inicial 0, pode-se utilizar, por exemplo, o comando **mudedç 90**, para mudar sua direção:


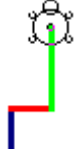
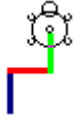





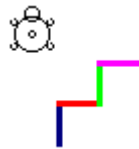

Explorando alguns comandos

A seguir serão apresentados alguns comandos a partir do desenho de uma escada e de um vitral.

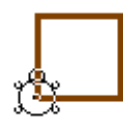
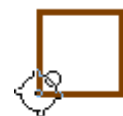

Desenhando uma escada




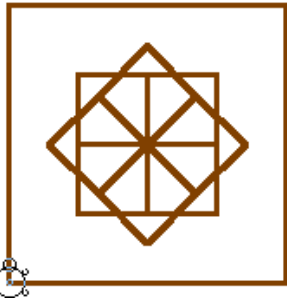
comando	descrição	exemplo
mudeel <lista>	altera a espessura do lápis de acordo com os números contidos na lista. Os dois números devem ser iguais.	mudeel [3 3]
pf <número>	desloca a tartaruga para a frente um determinado número de passos	pf 20 
pd <número>	gira a tartaruga à direita um determinado ângulo	pd 90 
mudecl <objeto>	muda a cor do lápis de acordo com o conteúdo de objeto que pode ser um número ou uma palavra correspondente à cor desejada. É utilizado para desenhar contornos de diferentes cores.	mudecl "vermelho"
		pf 20 



pe <número>	gira a tartaruga à esquerda um determinado ângulo	pe 90 
		mudecl "verdeclaro
		pf 40 
ub	coloca uma borracha sob a tartaruga possibilitando apagar linhas já desenhadas	ub
pt <número>	desloca a tartaruga para trás um determinado número de passos	pt 20 
		pd 90 
ul	coloca um lápis sob a tartaruga possibilitando o desenho de linhas por onde ela passar.	ul
		mudecl " rosachoque
		pf 20 
		pe 90 
un	retira o lápis ou a borracha da tartaruga possibilitando deslocá-la sem deixar rastros visíveis.	un

mudexy <número> <número>	movimenta a tartaruga na tela segundo o sistema de coordenadas cartesianas. O primeiro número corresponde a posição da tartaruga no <i>eixo x</i> e o segundo no <i>eixo y</i> .	mudexy -10 80 
rotule <objeto>	imprime na tela gráfica o objeto especificado a partir da posição da tartaruga	rotule [cores] 

Desenhando vitral

tat	apaga a tela gráfica, colocando a tartaruga na sua posição e direção originais.	tat
		mudeel [3 3]
		mudecl "marrom
repita <número> <lista>	é utilizado quando se deseja repetir uma mesma lista de instruções um determinado número de vezes	repita 4 [pf 40 pd 90] 
mudedç <número>	gira a tartaruga o ângulo especificado (a mudança de direção considera o sentido horário)	mudedç 45 
		repita 4 [pf 40 pd 90] 

		<p>tat</p>
		<p>repita 8 [repita 4 [pf 40 pd 90] pd 45]</p> 
		<p>un mudexy -80 -80</p>  
		<p>ul repita 4 [pf 160 pd 90]</p> 

		<p>un mudexy -60 -60</p> 
mudecp <objeto>	muda a cor de preenchimento de acordo com o conteúdo de objeto. Este comando é utilizado para preencher áreas delimitadas.	mudecp "amarelo
Pinte	pinta uma região com a cor especificada no comando <i>mudecp</i> <objeto>.	<p>Pinte</p> 

O *Logo*, como pode ser observado, permite a programação de diversos tipos de ilustrações com formatos livres ou geométricos, com cores, efeitos de preenchimentos, escrita, *etc.*. Até o momento a exploração dos comandos *Logo* foi realizada no modo direto de uso, ou seja, as instruções para movimentação da tartaruga foram digitadas na linha de comandos. Neste modo de trabalho cada comando dado pelo usuário é imediatamente executado pelo computador. Assim, o usuário pode constatar o efeito das instruções dadas para movimentar a Tartaruga na Janela Gráfica. Porém, este modo de trabalho não permite que as instruções sejam armazenadas na memória do computador para possíveis reutilizações ou reformulações.

Para armazenar as informações dadas pelo usuário é necessário usar o **Editor**. É nesse modo de trabalho que são **definidos os procedimentos** em *Logo*, possibilitando assim que uma sequência de instruções fique armazenada na memória do computador para ser executada no momento que o usuário desejar.

Definindo procedimentos

Um procedimento é um conjunto de comandos organizados numa determinada sequência e referenciados por um determinado nome.

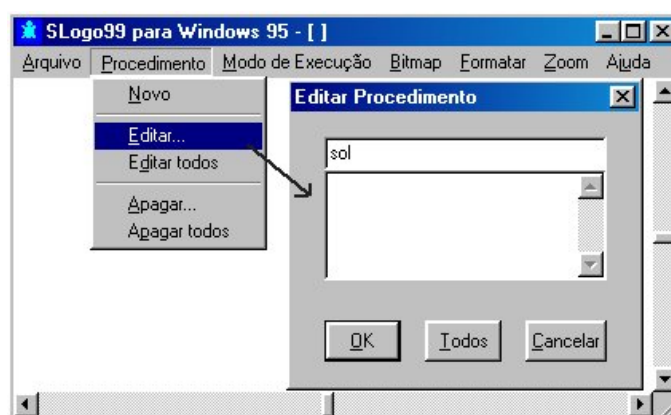
Todo procedimento precisa ter um nome que o identifica. Um procedimento inicia pelo comando **aprenda** e termina com o comando **fim**. Por exemplo:

```
aprenda sol           Nome do procedimento  
  
repita 12 [pf 30 pt 30 pd 30]  instruções  
  
fim
```

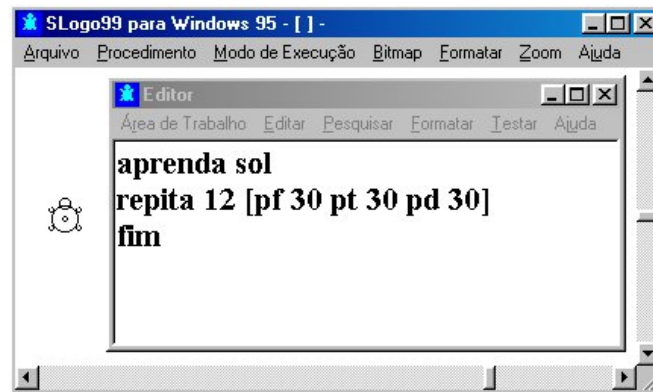
*Uma vez definido um procedimento no **Logo** ele poderá ser utilizado na definição de outros procedimentos. Quando um procedimento é usado como comando de outro procedimento diz-se que o primeiro é sub-procedimento do segundo. Por exemplo, o procedimento **sol** poderia ser um sub-procedimento do procedimento **paisagem**.*

```
aprenda paisagem  
sol  
un pt 90 ul  
repita 4 [pf 40 pd 90]  
fim
```

Para definir procedimento no *Logo* é necessário acessar a **Janela Editor** que é acionada o item **Editar** do menu **Procedimento** da Janela gráfica. Ao selecionar o item **Editar** aparecerá uma caixa de diálogo na qual deverá ser digitado um nome para o procedimento e acionado o botão **OK**.



Em seguida, aparecerá a janela do editor onde os comandos deverão ser digitados entre as linhas dos comandos **aprenda** e **fim**.



*Para abrir uma linha no Editor posicione o cursor (com o **mouse** ou as teclas de navegação ↑ ↓ → ←) no final do nome do procedimento e tecle **Enter** .*

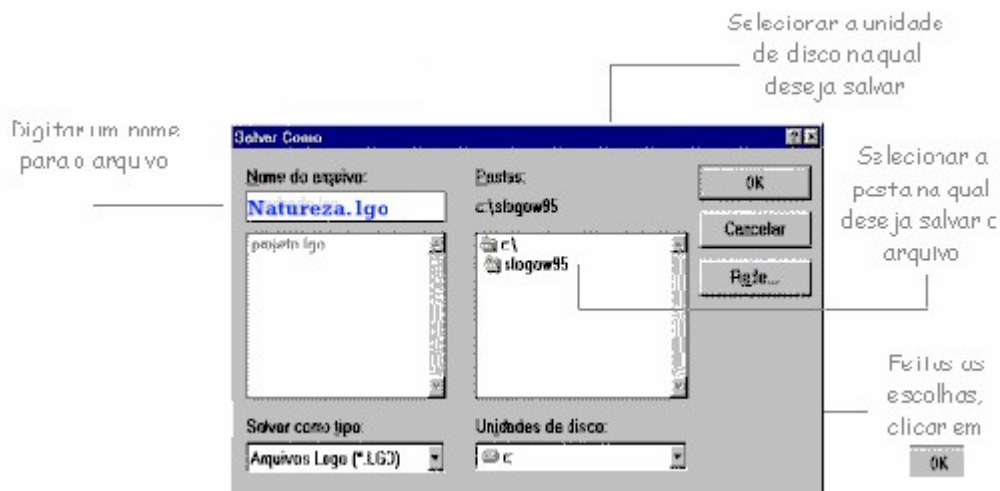
*Para executar o que foi definido no editor é necessário voltar para as Janela Gráfica e de Comandos. Para tal deverá ser acionada a opção **Sair** do menu Área de Trabalho da Janela Editor. Neste momento aparecerá uma caixa de diálogo informando que o conteúdo do editor foi modificado. Acione o botão **SIM** para confirmar a atualização feita no Editor.*

Um procedimento definido funciona de forma análoga aos comandos primitivos do Logo: basta digitar o nome do procedimento na Janela de Comandos e acionar o botão executar (ou teclar <enter>) para que as instruções sejam executadas.

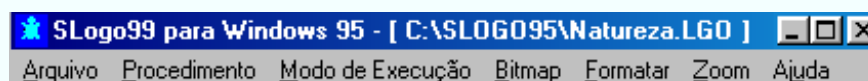


Salvando arquivos

A definição de procedimentos no Editor permite o armazenamento temporário na área de trabalho do *Logo*. Isto significa que se o ambiente *Logo* for fechado as informações contidas no editor serão apagadas. Para o armazenamento permanente dos procedimentos editados é necessário gerar um arquivo com extensão **.lgo**. Para tanto acionar a opção **Salvar Como** do menu **Arquivo**, existente na Janela Gráfica e especificar o nome e o local que será armazenado:



Concluída esta ação aparecerá na barra de título da Janela Gráfica o nome do arquivo e o local em que está armazenado.



Caso seja necessário modificar as informações existentes no editor atualize o arquivo **.lgo** acionando a opção **Salvar** do menu **Arquivo** da Janela gráfica.

O arquivo **.lgo** armazena apenas os procedimentos e variáveis definidas pelo usuário. Para salvar as imagens existentes na Janela Gráfica acione a opção **Salvar como** do menu **Bitmap**. Concluída esta ação será gerado um arquivo **.bmp** que ocupa bastante espaço de disco mesmo quando o desenho parece pequeno na tela do computador. Isto porque toda a área de desenho, mesmo em branco, é considerada parte da imagem.

Carregando arquivos

Para acessar arquivos já armazenados, no disquete ou no *Winchester*, é necessário carregá-lo para a área de trabalho do *Logo*. Para isso, acione o item **Abrir** do menu **Arquivo** da Janela gráfica:



Imprimindo Imagens ou Procedimentos

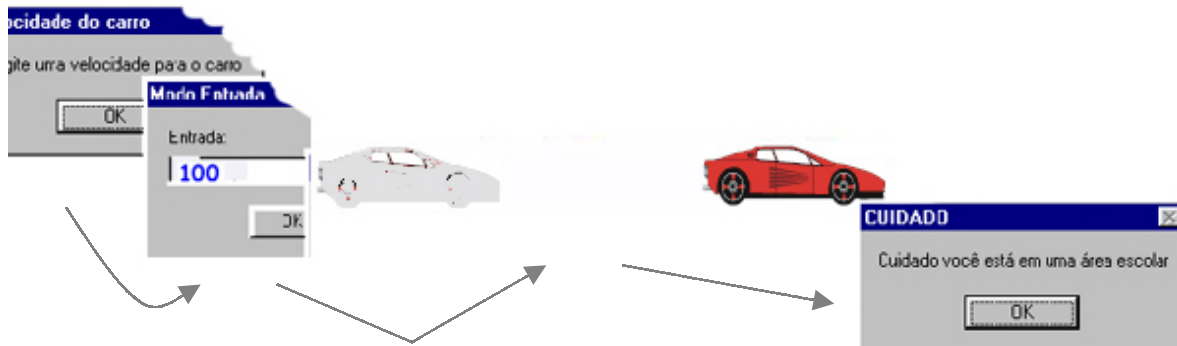
Para imprimir as imagens da Janela Gráfica, selecionar a opção **Imprimir** do menu **Bitmap**. Para imprimir os procedimentos de um determinado arquivo *.lgo* é necessário utilizar um editor de texto para abrir o arquivo *Logo* e, posteriormente, efetuar a impressão das informações contidas no mesmo.

Desenhando e animando a rua da escola

Para mostrar algumas possibilidades da programação *Logo* apresentaremos um exemplo detalhando os conceitos computacionais mais utilizados na elaboração do mesmo. Este programa desenha a rua de uma escola e faz animação de um carro a partir de um diálogo com o usuário.



Para movimentar o carro na Janela Gráfica o usuário tem que dar a velocidade que ele deseja que o carro assuma. Dependendo do valor dado pelo usuário o programa dispara efeitos diferentes na tela do computador. Se a velocidade for maior que 60, por exemplo, o carro se deslocará rapidamente e será emitido um alerta para o usuário. Veja ilustração abaixo:



Caso contrário, o carro se deslocará lentamente e o usuário visualizará uma outra mensagem.

A seguir serão detalhados alguns conceitos desta linguagem de programação que são utilizados para a implementação deste programa.

Desenhando retângulos

Definindo procedimentos com parâmetros

Observe que os desenhos que constituem este programa, tais como: quadra, praça, escola (parede, porta e telhado), utilizam retângulos de vários tamanhos. Pode-se definir um retângulo genérico que possibilita o desenho de todos estes elementos. Para tanto, é necessário usar o conceito de parâmetro. A idéia que vai ser desenvolvida a seguir é a definição de um procedimento genérico para desenhar retângulos de diferentes tamanhos.

Antes, porém, vejamos como definir um procedimento² para desenhar um retângulo de lados 40 e 100.

```
aprenda retângulopequeno
repita 2 [pf 40 pd 90 pf 100 pd 90]
; desenha um retângulo com lados de tamanho 40 e 100
fim
```



A generalização deste procedimento implica a declaração de dois parâmetros, um para o lado menor e outro para o lado maior.

```
aprenda retângulo :lado1 :lado2
repita 2 [pf :lado1 pd 90 pf :lado2 pd 90]
; desenha retângulo com lados de qualquer tamanho
fim
```

Neste caso o tamanho do retângulo não está definido no procedimento. Observe que os valores, 40 e 100, assumidos pelos comandos **pf**, no procedimento **retângulopequeno**, foram substituídos pelos parâmetros **:lado1** e **:lado2**.

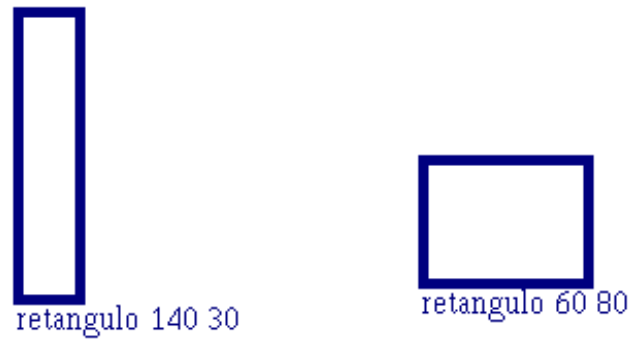
*A utilização de parâmetro no **Logo** envolve um processo de nomeação: o nome do parâmetro deve ser precedido de dois pontos (como por exemplo **:tamanho**). A definição de procedimento com parâmetro requer a colocação do nome do parâmetro na linha título do procedimento e no corpo do procedimento. Por exemplo:*

```
aprenda risco :tamanho
pf :tamanho un pf :tamanho ul
fim
```

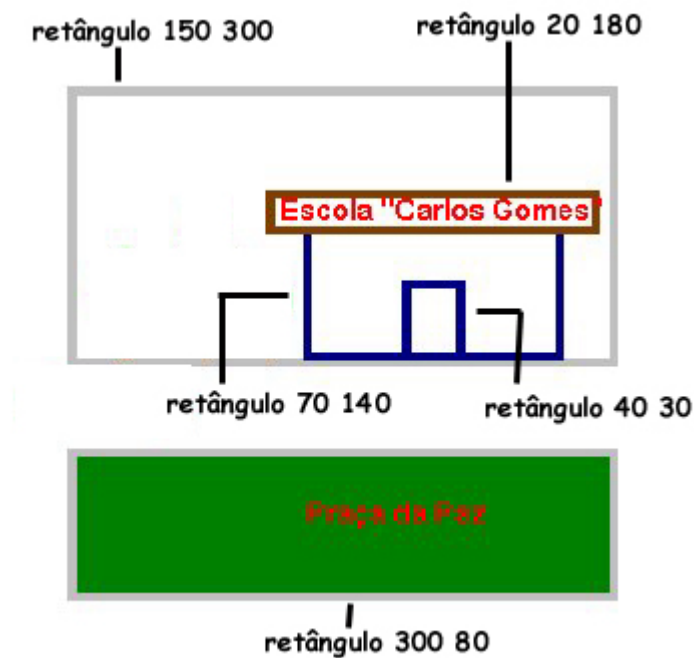
*Na execução do procedimento, na janela de comandos, é necessário que seja especificado o valor que cada parâmetro deverá assumir. (ex: **risco 100** ou **risco 500**) .*

² Para fazer um comentário em um procedimento *Logo* usa-se o sinal de ponto e vírgula (;)

O tamanho do retângulo dependerá do valor dado como entrada do procedimento na janela de comandos, por exemplo:



Este procedimento geral pode ser utilizado para compor as seguintes partes do cenário:



A disposição destes retângulos na Janela Gráfica depende da posição e orientação da tartaruga. Isto pode ser feito de duas maneiras: uma em relação à própria tartaruga e outra em relação ao sistema cartesiano. No primeiro caso o deslocamento é feito pelos comandos **pf** e **pt** e a orientação pelos comandos **pd** e **pe**.

pd 90 pf 55 pe 90

; desloca a tartaruga em relação ao seu último estado, isto é, à posição e direção

retângulo 40 30

;desenha retângulo de lados 40 e 30



O deslocamento da tartaruga em relação ao sistema cartesiano pode ser feito por meio dos comandos: **mudexy**, **mudex**, **mudey** e sua orientação pelo sistema polar: **mudedç**

mudexy -200 -50

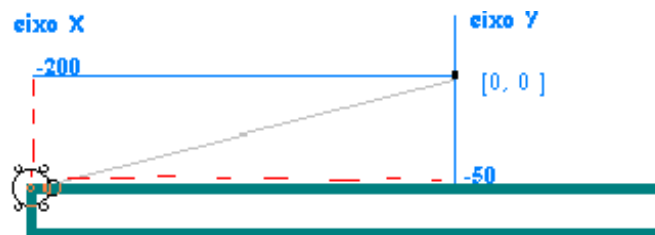
; desloca a tartaruga para um ponto da Janela Gráfica determinado pelos eixos **x** e **y** do sistema cartesiano

mudedç 90

; muda a direção da tartaruga na janela independente do seu último estado

retângulo 300 20

;desenha um retângulo de lados 300 e 20

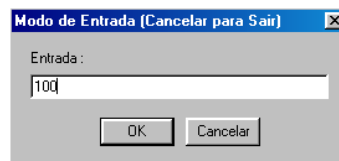


Criando animação

Para criar a animação neste programa a figura da Tartaruga foi substituída pela figura de um carro (imagem do tipo **.bmp**). A obtenção do efeito de movimento do carro é gerado pela repetição de uma sequência de comandos que faz o carro andar um valor determinado em função da escolha feita pelo usuário. Portanto, o movimento do carro dependerá de um parâmetro dado pelo usuário podendo, cada vez que o programa é chamado, surtir um efeito diferente.

Usando operações

A operação **leiap** permite que o usuário dê uma informação qualquer para o *Logo*. Esta operação aciona uma caixa (Modo de Entrada) de diálogo e fica aguardando a digitação de uma palavra/valor pelo usuário.



A operação **leiap** apenas retorna o valor/palavra dado pelo usuário. Para que o valor retornado pela operação seja executado é preciso relacioná-lo a algum comando da linguagem *Logo* ou procedimento definido pelo usuário.

*Veja um exemplo de utilização da operação **leiap** em um procedimento que faz o cálculo da idade a partir do ano de nascimento dado pelo usuário durante a execução do programa:*

```
aprenda idade
rotule [ Em que ano você nasceu?]
; rotula na janela gráfica uma frase
un pt 20
; desloca a tartaruga
rotule ( sentença [Você tem] 1999 - leiap "anos )
; rotula uma frase formada por 3 elementos: uma lista [Você tem], o
resultado de duas operações ( subtração e leiap) e a palavra "anos
fim
```

*Neste exemplo a operação aritmética **subtração** utiliza dois parâmetros: um é o valor do ano atual (1999) e o valor referente ao ano de nascimento que é dado pelo usuário durante a execução do programa, por meio da operação **leiap**.*

*O resultado das operações (**subtração e leiap**) passa a ser um dos parâmetros da operação **sentença**. Por exemplo:*

```
Sentença [Você tem] 37 "anos
Retorna [Você tem 37 anos ]
```

*No exemplo aqui apresentado, os valores retornados pelas operações **leiap**, **subtração** e **sentença** foram executados pelo comando **rotule**.*

*Em **Logo** existem dois tipos de primitivas: comandos e operações. A diferença básica entre elas é que um comando executa uma ação e uma operação retorna um valor. Este valor pode ser utilizado por outra operação e/ou por um comando.*

Usando variáveis

A função da variável é armazenar valores para serem utilizados quantas vezes forem necessárias por diversos comandos e operações. Para definir uma variável utiliza-se o comando **coloque**. Este comando necessita de dois parâmetros: um objeto (que pode ser do tipo palavra, número ou lista) e um nome. A sintaxe deste comando é:

`coloque <objeto> <nome>`

Exemplo:

```
coloque 100 "velocidade
; a variável "velocidade é criada e armazena o valor 100
```

*Para referenciar o conteúdo da variável usa-se o sinal de dois pontos (:) seguido do nome da variável. Exemplo: **rotule** :velocidade, **pf** :velocidade*

Para armazenar um valor dado pelo usuário durante a execução do programa utiliza-se a operação **leiap** na definição da variável. Exemplo

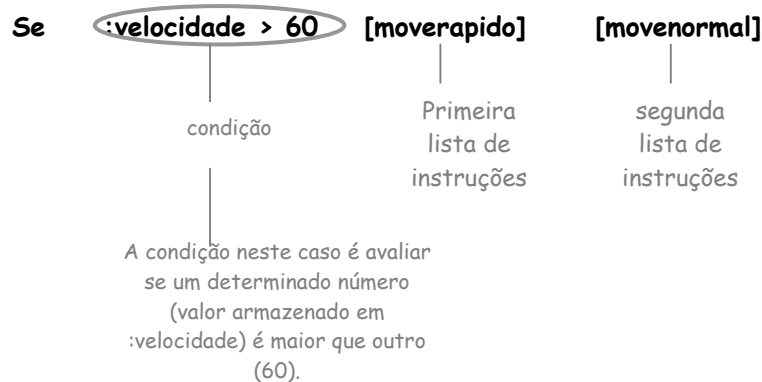
`coloque leiap "dados`

Usando condicionais

Para avaliar uma determinada condição a fim de desencadear diferentes efeitos pode-se utilizar o comando **se**. Este comando executa uma determinada instrução, caso a condição seja verdadeira ou falsa. A condição é sempre constituída por uma operação do tipo predicado. Um predicado em *Logo*, retorna valores booleanos, ou seja, verdadeiro ou falso. A sintaxe do comando **se** é:

`Se <predicado> <lista1> <lista2>`

Quando o resultado do predicado for verdadeiro o comando **Se** executará as instruções da lista1, caso seja falso executará as instruções da lista2. O comando **Se** tem a função de controlar o fluxo de execução do programa. No programa em questão, a condicional é utilizada para produzir diferentes efeitos de animação:

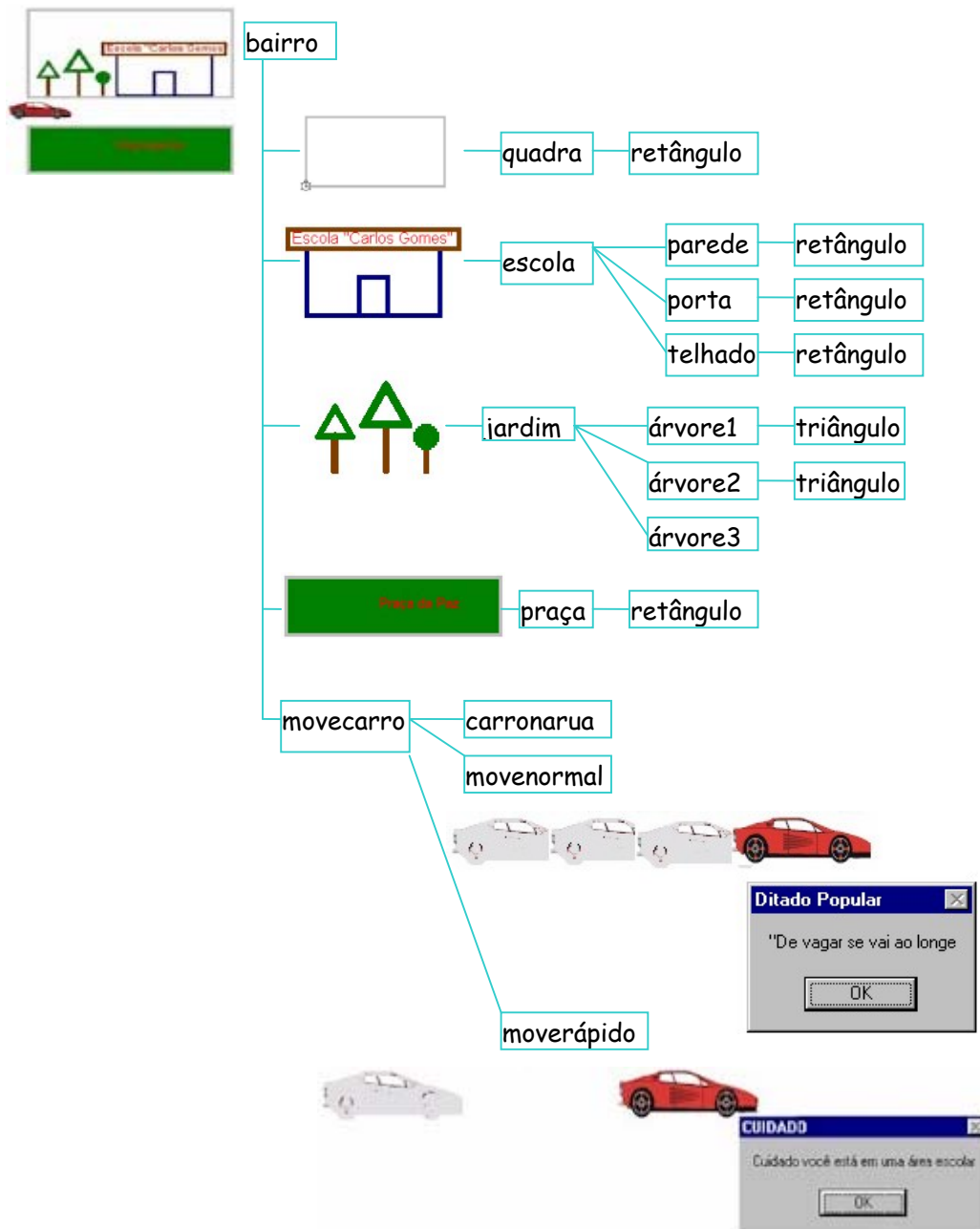


Isto significa que se o valor da velocidade dado pelo usuário (durante a execução do programa) for maior que 60 será executado o procedimento **moverápido**. O efeito desse procedimento é de produzir o deslocamento rápido do carro na Janela Gráfica e emitir um aviso de alerta. Caso o valor dado seja menor que 60 será executado o procedimento **movenormal** que provocará um deslocamento mais lento do carro e emitirá uma outra mensagem.

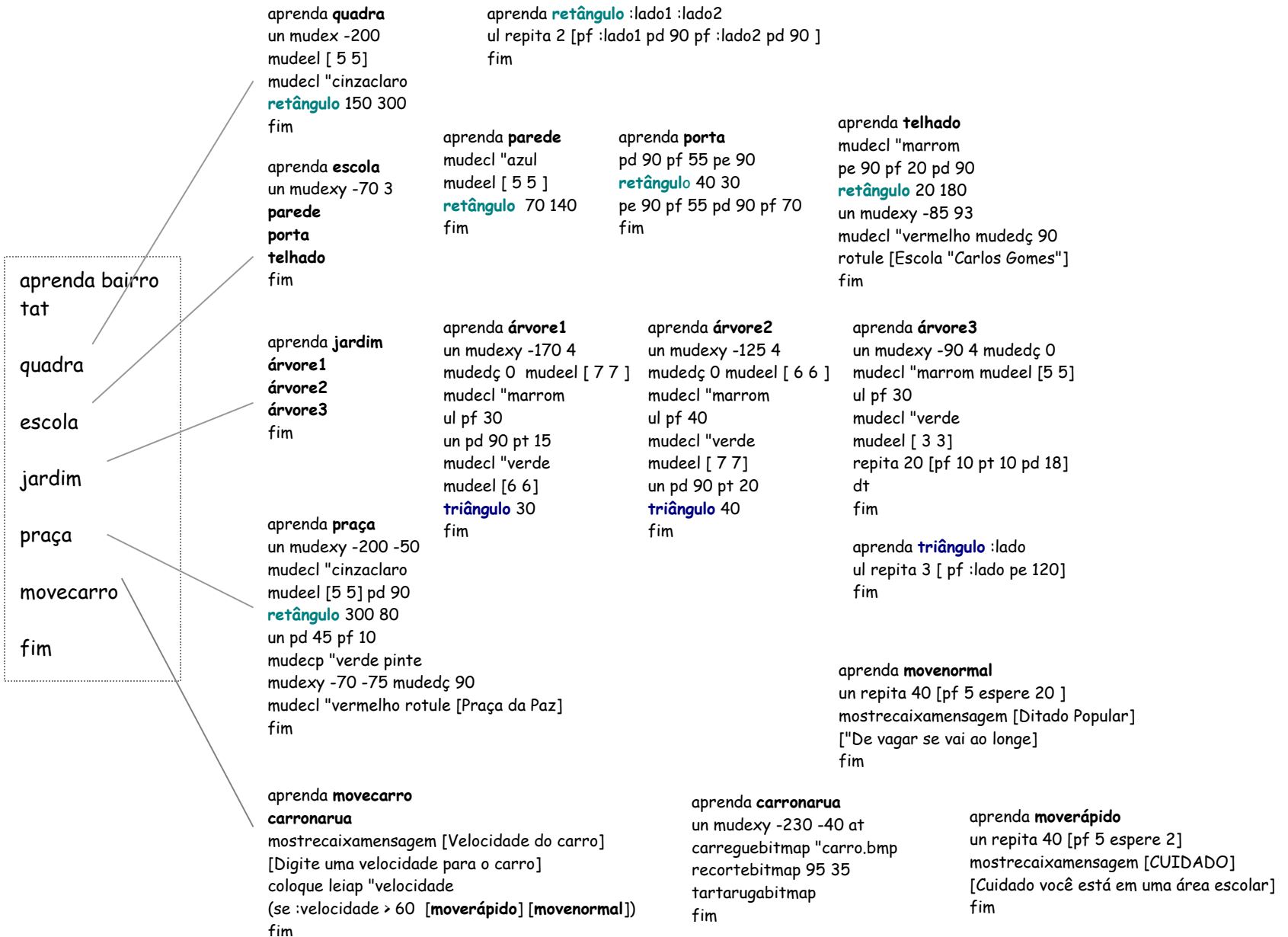
Estruturando procedimentos

Uma vez definidos os procedimentos no *Logo* eles podem ser utilizados na definição de outros procedimentos. No programa exemplo, os procedimentos **porta**, **parede**, **telhado** são utilizados como parte do procedimento **escola**. O procedimento **escola** por sua vez é parte do procedimento **bairro**. Em *Logo*, quando um procedimento é usado como comando de outro procedimento diz-se que o primeiro é sub-procedimento do segundo. Veja a seguir a estruturação dos procedimentos utilizados neste exemplo:

□ Estruturação dos Procedimentos







□ Listagem dos procedimentos

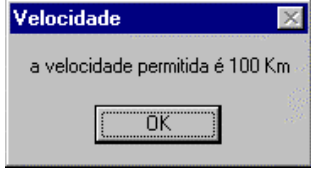


Apresentação novos Comandos e Operações³

A seguir mais algumas operações e comandos do *Logo* que foram utilizados no programa apresentado neste material e poderão ser úteis no item "atividades *Logo*".

comando	descrição	exemplo
carreguebitmap <nome do bitmap>	carrega um arquivo do tipo .bmp para a janela gráfica.	carreguebitmap "carro.bmp  Caso o arquivo a ser carregado não esteja no diretório ativo, a localização do arquivo deverá ser especificada: carreguebitmap "c:/windows/carro.bmp
recortebitmap <largura> <altura> colebitmap	"recorta" uma parte da imagem ativa na tela e a coloca na área de transferência do computador. A área a ser recortada (bitmap) é demarcada a partir da posição da tartaruga e assume a largura do primeiro parâmetro e a altura do segundo argumento do comando .	recortebitmap 50 35  utilizando o comando colebit a imagem alocada na área de transferência pode ser recolocada na tela gráfica: colebitmap 
tartarugabitmap	Substitui a imagem da tartaruga ativa pelo bitmap que estiver na área de transferência do computador.	tartarugabitmap 

³ A linguagem *Logo* possui um grande número de comandos e operações. Neste material apresentamos apenas alguns deles. O usuário pode consultar o menu ajuda na Janela Gráfica para verificar outras primitivas *Logo*.

Mostrecaixamensagem <título> <corpo>	pára o processamento e mostra uma janela de mensagem usando título e corpo especificados. O processamento continua a partir do momento que o usuário clica o botão ok.	mostrecaixamensagem [Velocidade] [a velocidade permitida é 100 Km] 
coloque <objeto><nome>	define uma variável para armazenar um dado valor. Este comando tem dois parâmetros: um objeto (que pode ser um número, palavra, lista) e um nome	coloque100 "velocidade
(se <predicado> <lista1> <lista2>)	Executa a lista 1 quando o retorno do predicado for verdadeiro ou executa a lista2 quando o retorno for falso	se :velocidade > 60 [pf 10] [pd 90]

operação	descrição	exemplo
leiap	Retorna a palavra dada como entrada pelo usuário.	rotule leiap
sentença <objeto1> <objeto2> (Sentença <objeto1> <objeto2> <objeto3>)	É uma operação que concatena objetos e os retorna em uma lista. Esta operação tem 2 parâmetros: <objeto1>, <objeto2> Quando a operação sentença tiver mais de 2 parâmetros deve-se abrir um parênteses antes do nome da operação e fechar após o último parâmetro dado.	rotule sentença [Hoje está fazendo] "frio > Hoje está fazendo frio rotule (sentença "Dia 23 [começa a primavera]) > Dia 23 começa a primavera

diferença <número1> <número2>	retorna o resultado da diferença entre número1 e número2.	rotule diferença 50 10 >40 ou rotule 50 - 10 >40
quociente número1 número2	retorna o quociente da divisão inteira de número1 por número2.	rotule quociente 1000 2 >500
produto número1 número2	retorna o produto dos números especificado	rotule produto 300 2
é maior <número1> <número2>	retorna verdadeiro se número1 for maior que número2. Caso contrário, retorna falso.	rotule é maior 5 6 >falso ou rotule 5 > 6 >falso
cv	esta operação pode ser usada apenas dentro do laço de um comando repita. Retorna o número de repetições que foram efetuadas, incluindo o atual.	Repita 2 [pf 10 un pf 10+cv ul]
palavra palavra1 palavra2	Retorna uma palavra composta pela concatenação de palavra1 e palavra2.	rotule (pal "a "pren "der)

- ❑ definição de procedimento com parâmetros (implica, por exemplo, a descrição de figuras com alguns atributos flexíveis)
- ❑ uso de operação (permite ao usuário entrar com um dado durante a execução do programa)
- ❑ condicional (possibilita a alteração do fluxo de execução, por meio da avaliação de uma determinada condição ser verdadeira ou falsa)
- ❑ estruturação de procedimentos (organização das instruções em procedimentos e subprocedimentos)