Resumo

Como o próprio título sugere, este texto apresenta um panorama geral da linguagem Logo. Neste, são mostradas de forma conjunta várias primitivas relacionadas à programação gráfica e simbólica, com o intuito de minimizar a segmentação existente entre o "universo da Tartaruga" e o "universo das listas". Embora o material possua um caráter amplo, a abordagem de cada primitiva procura caracterizar suas peculiaridades, sua funcionalidade, incorporando um linguajar mais técnico que auxilia o entendimento de alguns aspectos da atividade de programação. O material possui o conteúdo acima descrito em forma de texto e, também algumas sugestões de projetos sem, no entanto, esgotá-los. Estes servem para desencadear, no leitor, idéias de novos projetos. Embora os exemplos do material utilizem a implementação do Slogo, todos eles podem ser definidos usando-se outras versões do Logo. Este material constitui a primeira parte de um livro em preparação a ser lançado oportunamente.

Apresentação

Este texto é resultado de nossa experiência na realização de diversos cursos de programação Logo para os mais distintos usuários.

É fato conhecido que o aprendizado de programação Logo é visto como necessariamente dividido em duas etapas: Logo gráfico e Logo simbólico. Isto tem conduzido a uma estrutura de cursos bastante interessante. Existe um primeiro curso de Logo cujo conteúdo restringe-se a apresentação dos comandos gráficos da linguagem e, quando muito, introduz-se objetos de animação. Em seguida, quando solicitado, é apresentado um curso denominado Logo Avançado, onde são introduzidos as operações com palavras e listas. Isto tem gerado um senso comum de que a linguagem Logo é dividida em partes bastante diferentes e, raramente, pode-se ver projetos englobando os assim divididos universos da linguagem, que na realidade são um só. E com isso, uma programação mais avançada em Logo raramente é possível.

É também fato conhecido a extrema dificuldade de se transpor a barreira criada entre as duas partes da linguagem. A maioria das pessoas que programa Logo dificilmente programa problemas não gráficos. Assim, obtém-se uma visão distorcida do potencial de Logo enquanto uma linguagem de programação de propósito geral.

Por esta razão, iniciamos em nossos cursos, com bastante sucesso, uma abordagem integrada da linguagem. O curso inicial de Logo não se restringe mais a comando: gráficos e, sim, oferece um "passeio" significativo por todos os comandos e operações da linguagem. Neste "passeio procuramos ressaltar as diferenças conceituais entre as primitivas sem, contudo, separá-las em dois grandes conjuntos.

Outro aspecto que também acrescentamos aos nossos cursos foi um tratamento mais técnico da linguagem. Por ser Logo uma linguagem essencialmente dirigida a aprendizes, é comum verificar que ela, mesmo em cursos de formação de profissionais, é tratada com muita informalidade e, muitas vezes, com uma certa infantilidade. Acreditamos que não se pode ignorar que Logo é computacional e, portanto, formal e técnica. Ignorar esses aspectos dificulta o entendimento de muitos conceitos inerentes a qualquer linguagem de programação. Por exemplo, é muito comum programadores de Logo terem como conceito de variável somente o de parâmetro. Comprovamos que a introdução precoce de conceitos como a natureza da primitiva, natureza da primitiva, natureza dos parâmetros, ordem de execução de operações, etc., não dificulta em nada o aprendizado do iniciante em programação, pelo contrário, facilita o entendimento do funcionamento da linguagem em que estão aprendendo a se expressar.

Este texto é a primeira parte do material de apoio que estamos utilizando para atender a estes objetivos. É importante frisar que simplesmente o uso do texto não irá viabilizar o que pretendemos. Existe associado a ele toda uma postura pedagógica pois, de forma alguma, pretendemos menosprezar a dificuldade que é programar com autonomia em uma linguagem que possui, pelo menos, dois diferentes paradigmas.

Vejamos então, duas maneiras diferentes de estruturar o projeto casa:

```
AP CASA I
PAREDE
TELHADO
FIM

AP PAREDE
REPITA 5 [PF 100 PD 90]
FIM
```

```
AP TELHADO
REPITA 3 [PF 100 PE 120]
FIM
```

Esquematicamente, o interpretador Logo, funcionaria da seguinte maneira, durante o processo de execução do procedimento CASA I:

O subprocedimento PAREDE é uma variação de um quadrado. O número de repetições usado pelo comando REPITA deixa a Tartaruga em uma posição ideal, já adequada para a execução do subprocedimento seguinte que é TELHADO. Este, por sua vez, usa um giro para a esquerda que possibilita o acerto do TELHADO sobre a PAREDE. Note que essa solução implica a antecipação da posição e da direção da Tartaruga ao término de um subprocedimento e início do seguinte.

Outra forma de estruturar o projeto casa seria:

```
AP CASA 2
QUADRADO
ACERTATAT
TRIÂNGULO
FIM
```

```
AP QUADRADO
REPITA 4 [PF 100 PD 90
FIM
```

```
AP ACERTAT
PF 100
PD 30
FIM
```

```
AP TRIÂNGULO
REPITA 3 [PF 100 PD 120]
FIM
```

Veja a representação esquemática da execução de CASA2 pelo interpretador Logo:

Neste caso, optou-se por um procedimento intermediário - ACERTAT - que faz o acerto da posição e direção da Tartaruga possibilitando o arranjo entre as duas figuras básicas que compõem o desenho original: QUADRADO e TRIÂNGULO.

Os procedimentos QUADRADO e TRIÂNGULO possuem duas características bastante importantes do Logo: a modularidade e a transparência. Diz-se que um procedimento é modular quando ele é independente do seu contexto de uso e, portanto, pode ser reutilizado em muitas outras
Situações. Por exemplo, o QUADRADO poderia ser usado para fazer uma composição de quadrados como essa:

AP COMPOSIÇÃO
REPITA 3 [QUADRADO UN PF 25 PD 90 PF 25 UL PE 90]
FIM

Procedimentos como esse facilitam a atividade de programação. O próprio usuário vai criando um arsenal de ferramentas que ele já sabe como funciona.

A transparência de procedimento é parte da modularidade no sentido em que ela torna os procedimentos mais fáceis de serem manipulados. Um procedimento gráfico transparente faz com que a Tartaruga comece e termine o desenho na mesma posição e direção. Dessa maneira, o usuário não precisa se preocupar com a última posição ou direção da Tartaruga para idealizar o procedimento seguinte. Se ele dispõe de peças modulares e transparentes ele pode passar a se ocupar dos procedimentos de ligação, análogos ao procedimento ACERTATAT do exemplo anterior.

Entretanto, modularidade e transparência não se restringem à direção e posição da Tartaruga "default". É preciso compreender essas características de forma mais abrangente. Ao realizar um projeto computacional o usuário precisa pensar sobre o estado inicial dos vários objetos que ele estiver manipulando: cor do fundo de tela, número de tartarugas, cores dos lápis de cada uma delas, arrajo inicial de posição e orientação, tartarugas com ou sem lápis, aparentes ou não, com sprites ou não, ativadas ou não, etc.

A Diversidade de Descrições e a Diversidade da Estruturação de Procedimentos

Diferentes descrições dos problemas permitem diversas formas de estruturar os procedimentos que solucionam os mesmos. Observando a figura a seguir e algumas soluções possíveis:

Descrição 1

A tartaruga desenha dois quadrados sobrepostos. Há um giro entre eles para dispor os quadrados. Em seguida, a Tartaruga desenha um feixe de semi-retas

```
AP QUADRADO
REPITA 4 [PF 100 PD 90]
FIM

AP FEIXE
REPITA 8 [PF 50 PT 50 PD 45]
FIM

AP QUADRADOS1
QUADRADO
PF 50 PD 90
UNPT 20 PE 45 UL
QUADRADO
UN PD 45 PF 70 UL FEIXE
FIM
```

Descrição 2

A tartaruga desenha um feixe de semi-retas e, depois dois quadrados sobrepostos, mantendo um giro entre eles.

```
AP FEIXE
REPITA 8 [PF 50 PT 50 PD 45]
FIM

AP QUADRADO
REPITA 4 [PF 100 PD 90]
FIM

AP QUADRADOS 2
FEIXE
PT 50 PE 90 PT 50
QUADRADO
PF 50 PD 90
UN PT 20 PE 45 UL
QUADRADO
FIM
```

Decrição3

A Tartaruga desenha um quadrado pequeno e vira um pouco. No total ela desenha 8 quadrados e, portanto o giro entre casa quadrado é de 45 graus.

```
AP QUAPEQUENO
REPITA 4 [PF 50 PD 90]
FIM

AP QUADRADOS 3
REPITA 8 [QUAPEQUENO PD 45]
FIM
```

A leitura dos procedimentos permite-nos observar a diversidade das descrições na implementação de um mesmo projeto. Selecionamos apenas algumas das descrições possíveis neste contexto. Nosso objetivo não é apontar "a melhor descrição". Ao contrário, queremos salientar a maleabilidade da

atividade de programação que permite a cada pessoa utilizar, criar e descobrir suas próprias estratégias. A atividade de programação, assim compreendida, colabora no sentido de evitar o estabelecimento de soluções padronizadas. Dessa forma, aquele que está aprendendo a programar pode adquirir flexibilidade tanto na tarefa de resolução de problemas quanto no conhecimento dos recursos oferecidos pela linguagem computacional.

ATIVIDADES

A seguir são apresentados vários projetos computacionais que utilizam os conceitos explorados durante o "passeio" pelo Logo. Cada seqüência de mini- projetos destaca os principais conceitos e/ou recursos da linguagem envolvidos.

Experimente o seguinte comando:

REPITA 10 [MD SORT 360]

Complemente-o para:

...que a cada mudança de direção da tartaruga apareça escrito na tela o novo valor da direção;
...obter um maior tempo de leitura usando o comando ESPERE;
...a tartaruga mudar de cor aleatoriamente;
...a tartaruga andar um número fixo de passos;
...deixar o carimbo da tartaruga a cada repetição.

Escreva comandos para repetir a seguinte sequência:

Andar com a tartaruga um número aleatório de passos,
Girar a tartaruga para a direita um número aleatório de graus,
Escrever na tela a posição da tartaruga,
Mudar a cor da tartaruga aleatoriamente e carimbá-la.

Finalmente, reescreva os comandos de modo que o número de repetições também seja aleatório.

Observe o resultado do comando abaixo:

ESC SP "aparecida

- Modifique o comando para obter:

Pare
Cida
Ida

Par

-Selecione o a de **a**parecida
-Selecione o a de ap**a**recida
-Selecione o a de aparecid**a**

A partir da frase:
ESC [Voleibol é a maior diversão do mundo!]
- Escreva comandos para obter as seguintes frases:

Voleibol é a maior diversão
A maior diversão do mundo!
Diversão do mundo
Volei
Diversão

Considere as palavras abaixo:

ateu mola bica
-Escreva comandos para escrever as seguintes palavras:
time
taco
latão
motel
cama

A partir da frase:

ESC [Nada como um bom livro]
-Escreva comandos para escrever as seguintes sentenças (acrescente e/ou retire as palavras que forem necessárias)

Nada como um peixe
Um bom livro é companhia
Bom dia
Nada bom

A partir da palavra CAROLINA

-escreva comandos para obter:
a letra R
a letra I
a palavra CAROL
a palavra LINA

A partir da palavra AMORA

-escreva comandos para obter:
a palavra AMOR
a palavra RAMO
a palavra ROMA

A partir da lista [COCA GUARANÁ FANTA]

-escreva comandos para obter:
a palavra GUARANÁ
a palavra COCA COLA
a palavra FANTA LARANJA

A partir da lista [1 2[3 4 5]6]

-selecione o número 6
-selecione a lista [3 4 5]
-selecione o número 4

A partir da lista [O rato roeu[a roupa]]
-selecione a letra p da palavra roupa
-selecione a letra u da palavra roeu
-selecione a letra t da palavra rato

Usando as operações JI e JF construa as estruturas que se seguem a partir do seguinte comando:

ESC JI "nome [ ]

Meu nome
Meu nome é
Meu nome é Xuxa
O meu nome é Xuxa
O meu primeiro nome é Xuxa

A partir de cada sequência de operações, antecipe se o resultado será FALSO ou VERD:

ESC ÉNÚMERO PRI [42 53]
ESC ÉNÚMERO ULT [AB CD [30]]
ESC ÉVAZIA SP SP SP "Ana
ESC ÉPALAVRA SP SP SP "Ana
ESC ÉLISTA SP SU [São Paulo]
ESC ÉPALAVRA PRI [casa [40 50]]
ESC ÉNÚMERO PRI SP [AB 85 37]
ESC ÉLISTA ULT [A B [C]]
ESC ÉVAZIA SP SP [x] [y]]
ESC ÉPALAVRA ULT [[33] 45]
ESC ÉLISTA SP [alfa 13 77]

A partir de cada sequência de operações, antecipe se o resultado será FALSO ou VERD:

ESC E (ÉNÚMERO 423) (ÉPALAVRA PRI [44])
ESC E (ÉPALAVRA "a) (ÉLISTA [a])
ESC E (ÉPALAVRA PRI [a 3 4]) (ÉNÚMERO PRI SP [a 3 4])
ESC ALGUM (ÉNÚMERO [66]) (ÉLISTA [a b c])
ESC ALGUM (ÉVAZIA SP SU [3 4]) (ÉPALAVRA ULT[a b c])
ESC ALGUM (ÉLISTA "abacaxi) (ÉPALAVRA ULT [a b c])
ESC E (ÉPALAVRA ULT [amarelo 22]) (ÉNÚMERO SU 238X)
ESC E (ÉNÚMERO [5]) (ÉLISTA [a b c])
ESC (PRI [45 63]) > (SP ULT [36 543])
ESC (ULT 239) + 31 =40
ESC "Paulo=[Paulo]
ESC SÃOIGUAIS "Ana PRI [Ana]
ESC SÃOIGUAIS SP [lua] "lua

CONSIDERAÇÕES FINAIS

Nosso intuito ao desenvolver este material foi o de organizar informações relevantes para o leitor que deseja aprofundar e/ou formalizar seus conhecimentos sobre a linguagem Logo do ponto de vista computacional. Não se trata, portanto, de um manual organizador de "aulas" de um curso de Logo. Seu uso independe de um curso e vice-versa. Muitas vezes temos usado este texto como um material de apoio durante os cursos; outras, somente o recomendamos após a experiência do curso. Em outras palavras, sua leitura depende do contexto e do objetivo de uso da linguagem Logo. Note-se, por exemplo, que a definição de procedimentos é iniciada quase no final deste material. Essa escolha restringe-se a um critério de organização de um material escrito e não à abordagem utilizada em laboratório durante a realização de cursos.

BIBLIOGRAFIA

Abelson, N. e Abelson, A.. (1992) ***Logo for the Macintosh: Na Introduction Through Object Logo. Cambridge, MA: Paradigm Software Inc.***

Manual do Super Logo. (1994) Campinas, SP: NIED/UNICAMP.

PC Logo for Windows: Tutorial, Reference and Glossary. (1994) Cambridge, MA: Harvard Associates Inc.

Manipulando a Tartaruga

A tartaruga é um cursor gráfico que aparece no centro da tela gráfica. Para fazer desenhos basta movimentá-la na tela de modo que ela deixe traços pelo seu caminho. Há quatro comandos básicos que movimentam a tartaruga. Os comandos PARAFRENTE nº (PF nº) e PARATRÀS nº (PT nº) fazem a tartaruga andar e os comandos PARADIREITA nº (PD nº) e PARAESQUERDA nº (PE nº) giram a tartaruga[1]. Ao usar esses comandos é necessário especificar o número de passos ou a medida do grau do giro.

Observe a sequência dos comandos e acompanhe o efeito da mesma[2].

A tartaruga é definida por uma posição em relação a um sistema de coordenadas cartesianas (x, y) cujo ponto [0 0]representa o centro da tela gráfica e por uma orientação em relação a um eixo imaginário cujo ponto inicial é 0º. Os comandos PF e PT alteram a posição da tartaruga e os comandos PD e PE a sua orientação.

Os números que seguem os comandos PF, PT, PD, PE são chamados de parâmetros ou entradas. Existem comandos em Logo que não precisam de parâmetro como o comando TARTARUGA (TAT). Da mesma forma, há comandos que precisam de mais de um parâmetro. No exemplo mostrado acima os parâmetros usados são números mas, um parâmetro pode ser também uma palavra ou uma lista como veremos adiante. A omissão de um parâmetro quando ele é necessário produz uma mensagem de erro.

Para movimentar a tartaruga sem deixar traços usa- se o comando USENADA (UN) seguido de um comando que desloca a tartaruga. Analogamente, para apagar um traço na tela existe o comando USEBORRACHA (UB). Para retornar ao traço digita-se o comando USELÁPIS (UL). O comando DESAPAREÇATAT (DT) torna a tartaruga invisível e o comando APAREÇATAT (AT) faz retornar a sua figura. Ambos são úteis durante a realização de um desenho. Se for necessário recomeçar um desenho ou iniciar um novo pode-se usar o comando TAT que limpa a tela e recoloca a tartaruga na sua posição e orientação originais.

As figuras a seguir mostram o uso desses comandos. Note que os comandos não precisam de parâmetros.

A cor do traço da tartaruga pode ser mudada alterando - se a cor do lápis da tartaruga através do comando MC nº que significa mude a cor. Para se obter um melhor contraste pode - se modificar a cor da tela por meio do comando MCF nº que significa mude a cor do fundo. Cada número usado como

parâmetro desses comandos corresponde a uma cor pré - estabelecida pelas diferentes implementações Logo.

Utilizando várias Tartarugas

Na maior parte das implementações de Logo pode-se manipular várias tartarugas. A tartaruga que aparece no centro da tela gráfica ao iniciar o Logo é uma tartaruga "default" identificada pelo número 0. As demais tartarugas, uma acionadas através do comando ATENÇÃOTAT (ATAT), passam a obedecer os mesmos comandos usados pela tartaruga "default". À cada tartaruga corresponde um número sendo que o número máximo de tartarugas que podem ser acionadas depende de cada implementação. O ATAT precisa de uma única entrada que pode ser um número, caso se queira acionar apenas mais uma tartaruga, ou uma lista, caso se deseje várias tartarugas. Uma lista em Logo é sempre escrita usando-se colchetes ([]).

Uma vez estando ativada uma determinada tartaruga ela está apta a obedecer todos os comandos primitivos gráficos que forem dados a partir daquele instante. Em suma, o comando ATAT tem duplo papel: ao mesmo tempo que ativa uma ou várias tartarugas, implicitamente, desativa a(s) tartaruga(s) ativada(s) anteriormente. Durante a utilização de várias tartarugas o comando RG é bastante útil por que ele devolve ao sistema todas as características gráficas originais do Logo, ativando somente tartaruga de nº 0 - "default". Acompanhe através das figuras a seguir um exemplo de uso de várias tartarugas:

As tartarugas podem assumir diferentes figuras denominadas de "sprites" que são identificadas por números. O número total de sprites também depende do sistema Logo que se está utilizando. Há um certo número de sprites pré-definidos[3] e, geralmente, o Logo oferece meios para que novas figuras sejam definidas pelo usuário. O comando que viabiliza a mudança da figura de tartaruga para outra qualquer é MUDEFIG nº.

Estes comandos com outros recursos que veremos mais adiante permitem a implementação de programas que simulam animação.

Repetindo uma seqüência de ações

Um comando bastante útil do repertório do Logo é o comando REPITA. Ele é usado quando se quer efetuar uma mesma ação, ou sequência de ações, um determinado número de vezes. O REPITA precisa de dois parâmetros: um número e uma lista. O número refere-se ao número de vezes que a lista deve ser repetida e a lista refere-se as ações que devem ser realizadas. Portanto, a forma genérica desse comando é:

REPITA *<número>* *<lista>*

O comando REPITA pode ser usado em vários contextos, entre eles, para desenhar figuras geométricas. Observe diferentes contextos de uso do comando REPITA:

REPITA 4 [MF 5 MC 0 ESPERE 250 MFC 0 MC 15 ESPERE 250] [4]
A execução deste comando provoca os efeitos ilustrados nas duas figuras anteriores de forma alternada.

Outras Possibilidades do Logo

Como vimos o lápis da tartaruga pode assumir diferentes cores. Com este recurso pode-se pintar áreas perfeitamente delimitadas da tela, através do comando PINTE. Para pintar é necessário seguir alguns passos. Veja um exemplo de como proceder para pintar o interior de um quadrado.

Observe que para pintar a tartaruga deve ser levada, sem lápis, para dentro da região a ser pintada. Em seguida para pintar ela deve estar usando lápis.

De forma análoga funciona o comando CARIMBE. Com ele pode-se carimbar na tela as diferentes figuras usadas pela tartaruga. Por exemplo:

As tartarugas podem assumir diferentes velocidades através do comando MUDEVEL. O parâmetro desse comando é um número que corresponde à velocidade pretendida.

```
TL ATAT I AT UN
DESCONGELE
MUDEX - 40 PE 90
MUDEVEL 20
```

Através desta sequência de comando a tartaruga de número 1 fica em movimento contínuo como se estivesse "embrulhando a tela". Observe o comando TL, TireLimite. Caso ele não tivesse sido utilizado a tartaruga 1 pararia seu movimento no limite da tela. Se desejar parar as tartarugas em movimento, use o comando CONGELE.

Escrevendo na Tela

Pode-se escrever na tela através do comando ESCREVA (ESC). Este comando precisa de um parâmetro, que pode ser um número, ou uma palavra ou uma lista. Veja os exemplos:

Observe que quando a entrada do comando ESC é uma lista os colchetes não aparecem impressos na tela. O comando ESC só imprime o conteúdo da lista usada como parâmetro.

Pode-se escrever em diferentes lugares da tela usando-se o comando MUDECURSOR. O parâmetro desse comando é uma lista de dois números. O primeiro corresponde à coluna e o segundo à linha[6]. A figura a seguir ilustra alguns exemplos:

Compreendendo a natureza dos parâmetros

Como já foi dito, existem comandos que precisam de parâmetro. Esses parâmetros possuem diferentes naturezas: podem ser números, palavras ou listas.

Palavra é uma sequência de caracteres precedida por aspas ("). Por exemplo:
"casa "123 "dx40 "345F

Número é um tipo especial de palavra, que dispensa o uso de aspas para facilitar as operações aritméticas. Um número é constituído somente por dígitos, podendo ser um valor inteiro ou real (com ponto decimal). Exemplo:
123 354.5 1.5 48

Lista é um conjunto de palavras, números ou listas escrito entre colchetes ([]). Como por exemplo:

[abacaxi casa 32]
Esta lista possui 3 elementos: os dois primeiros são palavras e o último é número; ou, todos são palavras. Observe que quando uma palavra é elemento de uma lista não precisa vir precedida por aspas.

[[pastel] hamburger pizza]
O elemento de uma lista também pode ser uma lista como é o caso do primeiro elemento desse exemplo.

[[coca cola] guaraná suco]
Esta lista possui 3 elementos: o primeiro é uma lista e os dois últimos são palavras. Neste caso o primeiro elemento da lista é também uma lista de dois elementos que são palavras.

Existe a definição de lista vazia que é denotada por [] e palavra vazia que é denotada por aspas seguida por um espaço em branco: " .

Conhecendo as Operações

Operações são primitivas que possibilitam a passagem de informações. As operações retornam valores que são utilizados por comandos ou por outras operações. Há operações que lidam com o universo da Tartaruga, outras que manipulam exclusivamente números e que são denominadas de operações aritméticas, e outras ainda, que lidam com palavras e listas. Vejamos alguns exemplos de cada uma delas:

ESC CF
O comando ESC imprime na tela o resultado da operação CORDOFUNDO
(CF) que é um número que corresponde à cor do fundo da tela.

ESC CL1
O comando ESC imprime na tela o resultado da operação CORDOLÁPIS
(CL) que é um número corresponde à última cor sumida pelo lápis da tartaruga 1.[7]

ESC DÇ 0
O comando ESC imprime na tela o resultado da operação DIREÇÃO (DÇ) que é um número que corresponde à última orientação da tartaruga 0.

ESC FIG 5
O comando ESC imprime na tela o resultado da operação FIGURA (FIG) que é um número que corresponde à figura que está sendo usada pela tartaruga 5.

ESC POS2
O comando ESC imprime na tela o resultado da operação POSIÇÃO (POS) que é uma lista que corresponde à posição adotada pela tartaruga 2 de acordo com o sistema de coordenadas cartesianas (eixo x e eixo y respectivamente).

As operações aritméticas como: + ou - * / precisam de parâmetros sempre numéricos e produzem como resultado, também, um número. Por exemplo:

PF 50 * 2
O comando PF desloca a tartaruga 100 passos que corresponde ao resultado da operação de multiplicação.

REPITA 6 [PT 50 PE 360/6]
O comando PE gira a tartaruga 60 graus que corresponde ao resultado da operação de divisão.

PT 5 + 6 +15 + 5
O comando PT desloca a tartaruga 31 passos que é o resultado das três operações de adição.

PD 3* (DÇ 1)
O comando PD gira a tartaruga 1 um determinado número. Este número é o resultado da operação de multiplicação que, por sua vez, usa como um dos parâmetros o resultado da operação DÇ.

Observe como o interpretador Logo faz a avaliação dessa instrução, supondo-se que a tartaruga de nº 1 estivesse orientada em 30º.

Há um extenso conjunto de operações que lidam com palavras e listas. Ele pode ser subdividido de acordo com diferentes funções: operações que separam elementos, que buscam e contam elementos, que constroem estruturas e que concatenam elementos. A seguir apresentaremos as diferentes operações agrupados de acordo com as suas funções:

Separando Elementos

Existe, basicamente, dois tipos de operações que facilitam a separação de elementos de palavras e listas. Operações que retornam o primeiro ou o último elemento de uma palavra ou de uma lista: PRIMEIRO (PRI) e ÚLTIMO (ULT). E, operações que retornam o restante de uma palavra ou de uma lista sem o primeiro ou sem o último elemento da palavra ou da lista: SEMPRIMEIRO (SP) e SEMÚLTIMO (SU). Essas operações não aceitam palavras ou listas vazias como parâmetro. Veja alguns exemplo:

ESC PRI "abacaxi
O comando ESC imprime na tela o resultado da operação PRI que é a palavra a

ESC PRI [a b c]
O comando ESC imprime na tela o resultado da operação PRI que é a palavra a

ESC SP "abacaxi
O comando ESC imprime na tela o resultado da operação SP que é a lista [b c][9]

ESC SU "amora
O comando ESC imprime na tela o resultado da operação SU que é a palavra amor

ESC PRI 5476
O comando ESC imprime na tela o resultado da operação PRI que é o número 5

O comando ESC imprime na tela o resultado da operação SU que é o número
679

ESC SU [246]
O comando ESC imprime na tela o resultado da operação SU que é a lista
[2 4]

Observe que as operações SP e SU retornam valores que são sempre da mesma natureza do seus parâmetros. Isto é, se o parâmetro da operação SP for uma palavra o resultado será uma palavra e se o parâmetro for uma lista, o resultado será uma lista.

Pode-se também, combinar essas operações para se obter operações mais sofisticadas. Suponha que se queira imprimir a letra m da palavra ameixa. Pode-se combinar as operações da seguinte forma:

ESC PRI SP "ameixa

A forma de execução deste comando pode ser descrita pela seguinte sequência de passos:

1.O interpretador reconhece o comando ESC e sabendo que ele precisa de um parâmetro continua lendo a instrução...

2.Em seguida o interpretador encontra a operação PRI e reconhece que o seu resultado será o parâmetro do comando ESC.

3.O interpretador sabe que a operação PRI precisa de um parâmetro e continua então lendo a instrução sob o controle da operação PRI.

4. A próxima palavra encontra é SP que é reconhecida como o nome de uma operação que também precisa de um parâmetro. O interpretador continua lendo sob o controle da operação SP. "Observe que ESC e PRI estão suspenso aguardando os seus parâmetros."

5. É encontrada a palavra ameixa que é reconhecida como parâmetro da operação Sp.

6. A operação SP é resolvida retornando a palavra meixa.

7. Este valor é passado para a operação PRI como seu parâmetro. A operação PRI é efetivada retornando o valor m.

8. Este valor é o parâmetro de ESC que é então executado imprimindo na tela a palavra m.

Veja uma representação dessa avaliação:

Pode-se obter o valor da coordenada y[10] da tartaruga de nº 0 utilizando o vallor que retorna da operação POS. Para tanto é necessária uma combinação de operações do tipo:

ESC ULT POS 0

Esquematicamente e supondo-se que a tartaruga de n] 0 esteja na posição [30 50], a avaliação do comando seria feita pelo interpretador Logo na seguinte maneira:

Ou ainda, para se obter a letra c da lista [a b[c d e]f] pode-se fazer:
ESC PRI PRI SP SP [a b[c d e]f]
Ou
ESC PRI SU ULT SU [a b[c d e]f].

Um aspecto importante dessa forma de notação é o mecanismo que precisa ser bem compreendido para se operar com palavras e listas de modo geral. Observe que todas as operações em Logo, com exceção da aritméticas, são pré - fixas, ou seja, o operador é escrito antes dos respectivos operandos.

Buscando e Contando Elementos

Para selecionar um determinado elemento de uma palavra ou de uma lista usa- se a operação ELEMENTO (ELEM). Essa operação precisa de dois parâmetros: um número e uma palavra ou lista.

ELEM número <*palavra ou lista*>

O número indica a posição do elemento desejado da palavra ou da lista. Veja alguns exemplos:

ESC ELEM 2 [casa [pneu carro]]
O comando ESC imprime na tela o resultado da operação ELEM que é a lista [pneu carro]

ESC ELEM 5 "boneca
O comando ESC imprime na tela o resultado da operação ELEM que é a palavra "c

Quando se deseja saber o número total de elementos de uma palavra ou lista pode-se usar a operação NUMELEM (NEL). Essa operação só tem um parâmetro que é o próprio objeto.

ESC NEL [[abc]]
O comando ESC imprime na tela o resultado da operação NEL que é o número 1

ESC NEL "abc
O comando ESC imprime na tela o resultado da operação NEL que é o número 3

Construindo Estruturas

Pode-se construir estruturas do tipo palavras através da operação PALAVRA (PAL) e do tipo listas através das operações LISTA e SENTENÇA (SN). Todas essas operações precisam de no mínimo dois parâmetros[11]. A seguir apresentamos exemplos de cada uma dessas operações:

ESC PAL "cachorro" quente

O comando ESC imprime na tela o resultado da operação PAL que é a palavra "cachorro quente

ESC (PAL 34 57 28)
O comando ESC imprime na tela o resultado da operação PAL que é a palavra 345728

A operação PAL precisa de parâmetros que sejam palavras.

A diferença entre as operações LISTA e SN é bastante sutil. Por essa razão apresentaremos o mesmo exemplo para as duas operações para facilitar a comparação entre elas. Ambas as operações aceitam tanto listas quanto palavras como entrada.

ESC LISTA 45 80
O comando ESC imprime na tela o resultado da operação LISTA que é a lista [45 80]

ESC SN 45 80
O comando ESC imprime na tela o resultado da operação SN que é a lista [45 80]

A operação LISTA coloca em uma lista os parâmetros usados mantendo a natureza dos mesmos. Nesse exemplo, a operação SN é idêntica. O seu resultado é uma lista cujos elementos são as palavras usadas como entradas. Veja outros exemplos:

ESC (LISTA [tudo] [bem] [José])
O comando ESC imprime na tela o resultado da operação LISTA que é a lista [[tudo] [bem] [José]

ESC (SN [tudo] [bem] [José]
O comando ESC imprime na tela o resultado da operação LISTA que é a lista [[tudo] [bem] [José]]

ESC (SN [tudo] [bem] [José]
O comando ESC imprime na tela o resultado da operação SN que é lista [tudo bem José]

ESC LISTA [Bom] "Dia
O comando ESC imprime na tela o resultado da operação LISTA que é a lista [[Bom] Dia]

ESC SN [Bom dia] "Dia
O comando ESC imprime na tela o resultado da operação SN que é a lista
[Bom dia]

Observe que nos quatro exemplos anteriores a operação LISTA não altera a natureza das entradas; nesse exemplo, elas continuam sendo lista e palavra enquanto elementos do resultado final. O mesmo não acontece com a operação SN; o resultado final apresenta elementos que são palavras, independentemente da natureza original dos parâmetros. Mais exemplos:

ESC LISTA [8] [3[6]]
O comando ESC imprime na tela o resultado da operação LISTA que é a lista [[8] [3[6]]

Note como a operação SN trata os parâmetros quando esses são listas constituídas por sub- listas. A operação retorna uma lista cujos elementos são palavras e listas. O parâmetro que originariamente era lista orna-se uma palavra da lista final. A sub- lista do parâmetro original torna-se uma lista na lista final.

Resumindo, a operação LISTA retorna uma lista de suas entradas e a operação SN retorna uma lista dos elementos de suas entradas.

Concatenando Elementos

Qualquer palavra ou lista pode ser adicionada a uma outra lista. A operação JUNTENOINICIO (JI) adiciona uma palavra ou lista ou lista no inicio de uma determinada lista e a operação JUNTENOFIM (JF) adiciona uma palavra ou lista no fim de uma dada lista. Portanto, o primeiro parâmetro dessas operações é pode ser uma palavra ou lista e o segundo é sempre uma lista. O resultado final dessas operações é sempre uma lista. Vejamos alguns exemplos:

ESC JF [coca] [[cachorro quente] hamburger pastel]
O comando ESC imprime na tela o resultado da operação JF que é a lista
[[cachorro quente] hamburger pastel [coca]]

ESC JI "fim [[começo]]
O comando ESC imprime na tela o resultado da operação JI que é a lista [fim [começo]]

ESC JF "início []
O comando ESC imprime na tela o resultado da operação JF que é a lista [início]

ESC JI 30 [de março de 1961]
O comando ESC imprime na tela o resultado da operação JI que é a lista [30 de março de 1961]

ESC JF PRI POS 0 SP POS 1
O comando ESC imprime na tela o resultado da operação JF que é uma lista.
Veja o esquema da avaliação desta instrução supondo que a tartaruga de nº 0 está na posição [30 40] e a de nº 1 na posição [10 20]:

Conhecendo os Predicados

São denominados predicados as operações que retornam valores booleanos identificados pelas palavras VERD e FALSO que representam verdadeiro e falso respectivamente. Por exemplo ÉVISÍVEL é uma operação que retorna a palavra VERD se a tartaruga está visível na tela e FALSO caso contrário e, portanto, é um predicado.

Dentre as operações de manipulação de palavras e listas existem predicados definidos para saber se um determinado objeto é um palavra, uma lista, um elemento de uma estrutura, etc.

Por exemplo, ÉLISTA é um predicado que tem um parâmetro que pode ser uma palavra ou uma lista e retorna VERD se for lista e FALSO caso contrário.

ÉPALAVRA é outro predicado que tem um parâmetro, que pode ser uma palavra ou lista. Retorna VERD se o parâmetro for uma palavra e FALSO se for uma lista. Com funcionamento análogo existe o predicado ÉNÚMERO que verifica se o parâmetro é um número ou não.

ÉVAZIA é também um predicado que tem como parâmetro uma palavra ou lista. Retorna VERD se o parâmetro for uma palavra vazia ou uma lista vazia e FALSO caso contrário.

Conhecendo os Operadores Lógicos e Relacionais

Os operadores lógicos: ALGUM, E e NÃO operam sobre os predicados e produzem VERD ou FALSO de acordo com a seguinte tabela, denominada Tabela Verdade:

PRED 1	PRED 2	ALGUM	E	NÃO PRED 1
F	F	F	F	V
F	V	V	F	V
F	F	V	F	F
F	V	V	V	F

A leitura desta tabela se faz da seguinte maneira: considerando a primeira linha, tem-se que se PRED 1 (predicado 1) é FALSO e PRED 2 (predicado 2) também é falso, a operação ALGUM aplicada aos dois predicados retorna o valor FALSO, a operação E também retorna FALSO e a operação NÃO PRED 1 retorna VERD. Observe que as operações ALGUM e E são binárias, ou seja, precisam de dois operandos, e a operação NÃO é unária, precisa de um operando.

ESC E (ÉNÚMERO 123) (ÉPALAVRA PRI [123 456])
O comando ESC imprime na tela o resultado do operador lógico E que é a palavra VERD. Neste caso os parênteses servem para clarificar a operação desejada.

Acompanhe o esquema avaliação:

ESC ALGUM (ÉLISTA "ana) (ÉNÚMERO[680])
O comando ESC imprime na tela o resultado do operador lógico ALGUM que é a palavra FALSO

ESC NÃO ÉLISTA [5 6 7]
O comando ESC imprime na tela o resultado do operador lógico NÃO que é a palavra FALSO

Os operadores relacionais: Maior (>), menor (<) e igual (=) são usados na construção de expressões lógicas que retornam VERD ou FALSO. Por exemplo:

ESC 738 > 725
O comando ESC imprime na tela o resultado do operador relacional > que é a palavra VERD

ESC 7 = 7,0
O comando ESC imprime na tela o resultado do operador relacional = que é a palavra VERD

ESC "Paulo = [Paulo]
O comando ESC imprime na tela o resultado do operador relacional = que é a palavra FALSO

O operador relacional = possui também uma notação préfixa que é SÃOIGUAIS. Veja mais exemplos:

ESC SÃOIGUAIS " []
O comando ESC imprime na tela o resultado do operador relacional SÃOIGUAIS que é a palavra FALSO

ESC SÃOIGUAIS [] SP [elemento]
O comando ESC imprime na tela o resultado do operador relacional SÃOIGUAIS que é a palavra VERD

ESC SÃOIGUAIS (PRI POS 0) COORX 0
O comando ESC imprime na tela o resultado da operador relacional SÃOIGUAIS comparando o resultado da operação PRI POS 0 e da operação COORX 0, que é VERD.

Definindo Procedimentos

Os comandos e operações vistos até aqui fazem parte do conjunto de primitivas da linguagem Logo e podem ser usados na definição de novas palavras, isto é, na criação de procedimentos que expandem o conjunto inicial de palavras conhecidas da linguagem de programação. A criação de novas palavras é possível graças à atividade de programação. Embora o mecanismo de definir programas em Logo seja bastante simples, a atividade de programação é extremamente interessante por que obriga o usuário a pensar no processo de solução de um problema e no domínio de conhecimento que será utilizado neste processo. Além disso é a única maneira de se Ter controle sobre o computador, de modo a fazê-lo produzir qualquer resultado que se deseje.

Na maioria das implementações da linguagem Logo, para se definir um procedimento é necessário estar no modo de edição ou no editor de programas.

Suponha que se queira definir um triângulo equilátero com o lado de tamanho 100. A sequência de comandos a serem dados a Tartaruga poderia ser:

```
PF 100
Pd 120
PF 100
PD 120
PF 100
Pd 120
```

Ou

```
REPITA 3 [PF 100 PD 120]
```

Se desejássemos definir um procedimento Logo que desenhasse este triângulo deveríamos, dentro do modo de edição, teclar:

```
AP TRIÂNGULO
PF 100
PD 120
PF 100
PD 120
PF 100
FIM
```

Ou

```
AP TRÂNGULO
REPITA 3 [PF 100 PD 120]
FIM
```

Ao sairmos do modo de edição e retornarmos ao modo direto de uso podemos dar o comando TRIÂNGULO para a tartaruga, obtendo-se o resultado desejado, como pode ser visto na figura:

Neste caso, dizemos que TRIÂNGULO passa a fazer parte do elenco de palavras que a Tartaruga conhece assim como: UB, TAT, etc..

Todo procedimento em Logo tem:

1. Uma linha título que consiste no uso do comando AP seguido de um nome[12] que referencia o procedimento.
2. Um corpo, representado pelo conjunto de comandos que compõem aquele procedimento
3. Uma linha de finalização que consiste no uso do comando FIM

Portanto, um procedimento em Logo é uma sequência finita de comandos que se caracteriza por duas coisas: ouso do comando Aprenda (AP) seguido de um nome e o uso do comando FIM que assinala o término das instruções pertencentes ao procedimento.

Existe em Logo o conceito de área de trabalho que é uma região de memória do computador onde todos os procedimentos definidos durante uma sessão de trabalho ficam armazenados. No exemplo do TRIÂNGULO, ao sairmos do modo de edição, o procedimento que define o triângulo fica armazenado na área de trabalho enquanto se está usando o Logo. Se desejarmos um armazenamento permanente em disquete ou disco rígido isto deverá ser feito usando-se comandos para manipulação de arquivos específicos de cada implementação.

Estruturando um Projeto Simples

Uma vez definido um procedimento ele pode ser usado na definição de outros procedimentos. Por exemplo, o procedimento TRIÂNGULO visto anteriormente pode ser reutilizado nos seguintes contextos:

Quando um procedimento é usado como comando de um outro procedimento, diz-se que o primeiro é um subprocedimento do segundo. No exemplo, TRIÂNGULO é um subprocedimento de TREVO.

Suponhamos que se queira implementar procedimentos que desenhem uma casa como a da figura a seguir:

À princípio, pode-se criar um único procedimento que descreve passo-a- passo as ações da Tartaruga. Poderíamos, então, definir o seguinte procedimento:

```
AP CASA
PF 100 PD 90
PF 100 PD 90
PF 100 PD 90
PF 100 PD 90
PF 100 PD 30
PF 100 PD 120
PF 100
FIM
```

Apesar de ser uma solução, esta forma de definir nem sempre é a mais adequada. No desenvolvimento de projetos mais complexos acaba-se tendo procedimentos excessivamente longos e de difícil entendimento. Imagine o trabalho do usuário para descobrir um giro da Tartaruga equivocado no meio de um procedimento de 50 linhas! Daí a necessidade de introduzirmos a noção de estruturação de procedimentos.