

Programação Básica

1. Programas e Algoritmos

Noções de Lógica

A ciência filosofia tem dois ramos: a ética e a lógica. Lógica é parte da filosofia que estuda as formas do pensamento e modo de raciocinar corretamente. O objetivo da lógica é determinar se um raciocínio é verdadeiro ou falso e o raciocínio é a forma mais elaborada do pensamento.

Nós pensamos ordenadamente quando raciocinamos com lógica e pensamos de forma desordenada se raciocinamos sem lógica. Quando pensamos sem lógica, nós misturamos as idéias, agimos confusamente, tiramos conclusões rápidas e erradas. Raciocinar sem lógica é o mesmo que colocar as coisas “de pernas para o ar”. No raciocínio lógico as idéias são desencadeadas dentro de normas e as conclusões são verdadeiras.



O elegante termo **silogismo** diz respeito ao raciocínio lógico. Ele é uma forma de pensamento muito simples, mas que segue regras, isto é, lógica formal.

A lógica nasceu na Grécia Antiga ou Grécia Clássica (mais ou menos 500 a. C.), quando floresceu a filosofia com Sócrates, Platão, Aristóteles e muitos outros. Aos poucos, com o constante crescimento do saber, a filosofia deixou de abranger todo o conhecimento e foi assim que ela se desmembrou em várias ciências: matemática, física, química, biologia, ecologia, geologia, história, economia e muitas, muitas outras.

Foi nessa época que floresceu o silogismo; o mais citado exemplo de silogismo ou de raciocínio lógico é:

- Todo homem é mortal.
- Sócrates é homem.
- Logo, Sócrates é mortal.

Neste exemplo existem duas premissas encadeadas e uma conclusão final.

Um falso silogismo, isto é, raciocínios sem seguirem as normas da lógica formal, é chamado **sofisma**.

Programação Básica

Há lógica em nosso dia-a-dia?

Pensar é próprio do ser humano. O ser humano sempre pensa, mas não necessariamente pensa com lógica ou logicamente. Nós podemos pensar com lógica ou ilógica, ou seja, sem lógica. A palavra é o meio usado para expressar nossos pensamentos; Nós expressamos nossos pensamentos e idéias, ou nos comunicamos com os outros, por meio de palavras, em linguagem escrita ou linguagem oral. Em nosso dia-a-dia quando escrevemos ou quando falamos, nós o fazemos com ou sem lógica, pela lógica ou ilógica.

Imagine por exemplo que você queira comprar um aparelho de televisão de 20 polegadas. Na loja o vendedor lhe diz:

- Este aparelho da conhecida marca X custa tantos reais.
- Este aparelho da conhecida marca Y custa tantos reais a menos.
- Ele lhe apresenta os catálogos ou lhe propõe a lhe falar as vantagens e desvantagens de cada um.

• Quando finalmente você se decide por adquirir um dos aparelhos, você pode escolher apenas por ser mais bonito ou por motivos lógicos.

Imagine outra situação:

Você pretende comprar um brinquedo mecanizado. O vendedor da loja lhe diz:

- Este carro de corrida é muito bom, muitas crianças procuram por ele e custa tantos reais.
- Este kit de robô é um brinquedo pedagógico e custa tantos reais.

Você deve informa-se de todos os possíveis detalhes de ambas as ofertas para poder fazer sua escolha com lógica.

Isto faz você perceber a importância da lógica em nossas vidas na teoria e na prática. Se você quer pensar, falar, escrever e agir corretamente precisa colocar ordem no pensamento, isto é, utilizar lógica.

Exemplos:

a) A gaveta está fechada.

A bala está na gaveta.

Para pegar a bala é preciso abrir a gaveta.

b) José é mais velho que João

João é mais velho que Rodrigo

Logo, José é mais velho que Rodrigo

Programação Básica

Lógica e Algoritmos

O objetivo fundamental da atividade de programação seja ela de computadores ou de robôs, é a elaboração de um ou mais algoritmos que permitam ao robô alcançar o objetivo desejado.

Um algoritmo é uma seqüência de passos, que se forem seguidos, permitem atingir um objetivo bem definido ou cumprir determinada tarefa. Apesar de achar este nome estranho, embora o termo algoritmo não seja muito usado, os algoritmos estão sempre presentes em nosso dia-a-dia. Por exemplo, em receitas de bolo, além dos ingredientes necessários está descrita também uma seqüência de passos para fazer aquele determinado tipo de bolo.

Um programa é a formalização de um algoritmo construído em determinada linguagem de programação para seja entendida por um computador ou um robô.

Por mais sofisticada que seja uma linguagem de programação possa ser ela sempre será muito mais simples que as linguagens que nós usamos para nos comunicar com nossos semelhantes. Isto ocorre porque somos mais complexos e mais inteligentes que qualquer máquina já construída.

Por este motivo, quando escrevemos uma receita, ou melhor, um algoritmo para ensinar alguém a fazer café, podemos fazê-lo sem nos preocuparmos com detalhes como no exemplo a seguir:

Café para quatro pessoas

- Coloque água na cafeteira até a marca de quatro xícaras;
- coloque o filtro de papel;
- coloque quatro medidas de pó de café;
- ligue e espere...
- quando ficar pronto é só adoçar a gosto e servir.

Qualquer um de nós pode seguir este algoritmo e preparar nosso café; mas nenhuma máquina faria o mesmo sem ser programada.

Por exemplo, a receita não menciona o lugar em que é para colocar o pó de café ou em que lugar da cozinha está guardado o medidor. Esta falta de informação, com certeza, não o impediria de tomar o seu café, mas um robô não saberia o que fazer, simplesmente porque ele não tem bom senso, ele não raciocina. É por isso que devemos determinar, com detalhes, todas as ações que um robô deve executar, prevendo todos os possíveis obstáculos e a forma de transpô-los, isto é, descrever uma seqüência finita de passos que garantam a solução do problema. Essa atividade é realizada pelos programadores que podem ser chamados de “construtores de algoritmos”.

Na verdade, nós da PNCA-Robótica e Eletrônica, assim como os demais pesquisadores do planeta, trabalhamos para que um dia nossos robôs possam entender receitas como esta.

Programação Básica

2. A Linguagem de Programação LEGAL

A linguagem de programação LEGAL é apenas notação formal que descreve algoritmos a serem executados pelo MC2, que é o cérebro de seu robô.

Como em qualquer linguagem, a do LEGAL é formada por um conjunto de regras que descrevem como o programa deve ser escrito. Este conjunto de regras define a sintaxe da linguagem.

A língua portuguesa, também possui um conjunto de regras gramaticais ou sintáticas. São essas regras que nos informam quando uma frase está com sintaxe certa ou errada.

“A moça perguntou para o homem: o Sinhô está na fila?

Este livro custou vinte real.

Ontem nós fomos no cinema.”

Nessa três exemplos existem graves erros de sintaxe, ou seja, elas estão escritas de forma errada.

Além das regras sintáticas, o LEGAL tem um conjunto de regras semânticas que especificam o “significado” de qualquer programa. Da mesma maneira, a língua portuguesa, também, tem regras semânticas, ou de significado das palavras. Muitas vezes uma frase pode não conter erros sintáticos e mesmo assim ter significado confuso ou errado: “a loja me deu isto de graça.”

A linguagem LEGAL é uma linguagem de programação, por isso ela possui as principais estruturas comuns a todas as linguagens como: comandos condicionais, comandos de repetição, etc.

Além disso, a linguagem Legal também tem características próprias voltadas para o controle de robôs e outros dispositivos mecatrônicos. Entre estas características podemos destacar:

- Comandos para controlar motores e servos motores.
- Comandos para fazer a leitura de sensores externos.
- Comportamentos inteligentes pré-programados.
- Comandos para coletar e armazenar dados remotamente.

Programação Básica

Palavras Reservadas

As palavras usadas em uma linguagem de programação são chamadas palavras reservadas. A maioria das palavras reservadas do LEGAL são comandos que descrevem as ações que o robô deve executar. A seguir você pode ver uma tabela com todas as palavras reservadas do **LEGAL**:

POR	FAVOR	OBRIGADO	TOQUE	DÓ
RÉ	MI	LÁ	SOL	SI
Hz	LIGUE	DESLIGUE	L1	L2
L3	L4	L5	L6	ESPERE
S	SIM	NÃO	SERVO1	SERVO2
SERVO3	SERVO4	FRENTE	RÉ	DIREITA
ESQUERDA	RÁPIDO	LENTO	NORMAL	SEMPRE
REPITA	VEZES	SIGA	FAIXA	PRETA
BRANCA	SOM	LUZ	CALOR	FUJA
PARE	APRENDA	S1	S2	S3
S4	S5	S6	S7	S8
S9	S10	S11	BT	ENTER
M1	M2	POTÊNCIA	MOTORES	SE
ENTÃO	SENÃO	ENQUANTO	ATIVE	INTERVALO
COLETA	FIMCOLETA			

Você pode ver as palavras reservadas como tijolos que podem ser utilizados na construção de seus programas. São as regras sintáticas que nos dizem como esses tijolos podem ser unidos.

É importante ter em mente que todas as palavras reservadas têm significados fixos dentro da linguagem, significados que não podem ser alterados.

Você aprenderá como ensinar novos comandos ao seu robô; todo novo comando deverá receber um nome específico que não pode ser o mesmo de uma palavra reservada, ou o robô ficará confuso e não saberá o que fazer.

Programação Básica

Estrutura de um Programa em LEGAL

Agora vamos examinar a estrutura de um programa em **LEGAL**, focalizando os elementos individuais dessa estrutura.

Um programa em **LEGAL** consiste de quatro elementos estruturais:

- O módulo Principal;
- Dois módulos de eventos: Evento1 e Evento2 e
- O módulo Aprenda.

A figura a seguir mostra os módulos na tela do ambiente de programação do **LEGAL**. Para mudar de um módulo para outro basta clicar na aba correspondente ao módulo desejado.



Módulos de programação do LEGAL.

Programação Básica

Vamos conhecer agora cada um destes módulos:

Módulo Principal

Como o próprio nome diz, este é o mais importante módulo de um programa. Você pode escrever programas sem usar os módulos de eventos ou o módulo aprenda, mas sempre deverá usar o módulo principal.

Este módulo controla o funcionamento de seu robô, descrevendo suas ações e determinando seu comportamento. O programa escrito neste módulo tem o seguinte formato:

```
Por favor
  Comando 1
  Comando 2
  ...
  Comando n
Obrigado
```

Ou seja, seus programas devem iniciar com o comando Por Favor e **terminar com o comando Obrigado**. **Todas as instruções para seu robô devem ser escritas entre estes dois comandos. Os comandos escritos depois do Obrigado serão ignorados pelo LEGAL.**

Para facilitar seu trabalho, o LEGAL automaticamente coloca os comandos Por Favor e Obrigado todas as vezes que você iniciar um novo programa.

Módulo Evento

Nos módulos Evento1 e Evento2 é possível definir rotinas que serão executadas somente quando determinados eventos ocorrerem. Ao longo deste material você aprenderá como utilizar este módulo em seus programas.

Módulo Aprenda

Neste módulo você pode definir novos comandos para seu robô. No mundo da programação esses novos comandos são chamados sub-rotinas.

Obs: Nos módulos Evento 1, Evento 2 e Aprenda não é necessário iniciar o "código" escrito com os comandos Por Favor e terminar com o comando Obrigado. Isto é obrigatório apenas na módulo Principal.

Programação Básica

3. Iniciando a Programação em *LEGAL*

Agora que você já conhece os módulos de programação, vamos entender a linguagem *LEGAL* e escrever nossos primeiros programas.

Comentários

Antes de escrever seu primeiro programa, é importante aprender a comentá-los. Um comentário é um texto que não interfere na execução do programa, na verdade o *LEGAL* irá ignorar todos os comentários escritos em um programa. O objetivo de tais textos é ajudar na compreensão do Algoritmo.

Para fazer um comentário em **LEGAL** basta colocar o símbolo # que o restante da linha será considerado comentário pelo sistema.

```
# Este é um exemplo de comentário.
Por favor
  Comando1 # este é outro exemplo.
  Comando2
Obrigado
```

Se um comentário for maior que a linha, devemos colocar o símbolo # no início da próxima linha.

```
# Este é um exemplo de um comentário que é
# maior que o comprimento de uma linha.
Por favor
  Comando1 # este é outro exemplo.
  Comando2
Obrigado
```



Lembrete

Um programa sem comentários é como o caderno de um aluno que tem letra feia e não tem capricho: quando ele acaba de escrever somente ele e Deus sabem o que está escrito; depois de quinze dias ... só Deus.

Programação Básica

Emitindo Sons

No interior de seu MC2 há um dispositivo eletrônico semelhante a uma campainha de alarme, ele é responsável pelos sons produzidos pelo MC 2. Este dispositivo, através da variação da frequência e da duração do som pode produzir uma variedade de sons simples, porém divertidos.

Utilizando o comando **Toque** podemos programar o MC 2 para produzir sons de duas maneiras diferentes:

- A primeira, com a sintaxe:

Toque *nota duração*,

faz com que o módulo de controle reproduza a *nota* musical indicada com a *duração* especificada.

Nota pode assumir os seguintes valores: **Dó, Ré, Mi, Fá, Sol, Lá e Si**. Já a *duração* pode ser qualquer valor inteiro entre um e cinco. Se for solicitada uma duração com valor fora desta faixa o sistema emite a seguinte mensagem de erro:

A duração de uma nota deve estar entre 1 e 5.

```
# Exemplo:  
# Uso do comando Toque para reproduzir notas  
# musicais  
Por favor  
    Toque Dó 2  
    Toque Ré D3  
    Toque Mi 2  
Obrigado
```

Programação Básica

• A segunda maneira de se utilizar o comando **Toque** faz com que o MC2 produza sons em diferentes frequências. A sintaxe, neste caso é:

Toque *frequência Hz duração*

O valor da *frequência* deve ser maior que 2.000 Hz e menor 12.500 Hz; é importante você não esquecer de colocar a unidade de frequência Hertz, abreviado Hz. Caso você esqueça, o **LEGAL** mostrará a seguinte mensagem de erro:

Você esqueceu a unidade.

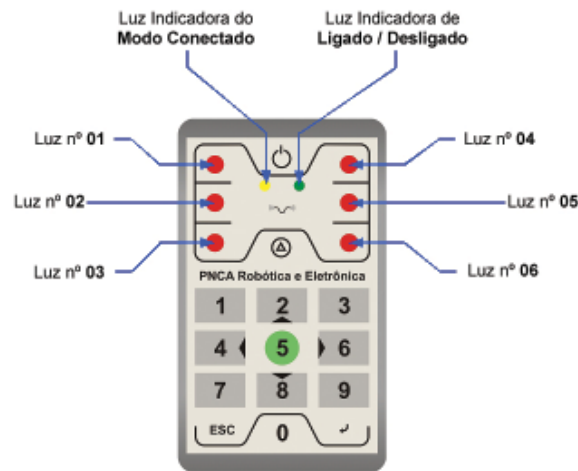
Um exemplo deste uso do comando Toque pode ser visto a seguir:

```
# Exemplo:  
# Uso do comando Toque para reproduzir sons  
# em diferentes frequências.  
Por favor  
    Toque 4000 hz 2  
    Toque 6200 hz 3  
    Toque 8000 hz 2  
Obrigado
```

Ligando as Luzes do MC2

O MC2 possui seis luzes que podem ser usadas para indicar o funcionamento de seus programas. Estas luzes são rotuladas como: L1, L2, L3, L4, L5 e L6, conforme pode ser visto na figura a seguir.

Programação Básica



Luzes do MC2.

O comando usado para ligar estas luzes é **Ligue** e para desligá-las o comando **Desligue**.

A sintaxe destes comandos pode ser vista a seguir:

ligue *atuador*

desligue *atuador*

No qual *atuador* pode ser uma das seis luzes do MC2. No exemplo a seguir o programa liga as luzes L1 e L4 e desliga a L6.

```
# Exemplo:
# Uso do comando Toque para reproduzir sons
# em diferentes frequências.
Por favor
  Ligue L1
  Ligue L4
  Desligue L6
Obrigado
```

Se esquecermos de indicar o atuador ou escrevermos um atuador desconhecido o **LEGAL** mostrará a seguinte mensagem:

O que devo ligar?

Programação Básica

Comando Espere

Em muitos casos pode ser necessário fazer com que o programa faça uma pausa de alguns segundos antes de continuar seu processamento. Para estes casos a linguagem **LEGAL** tem o comando **espere**, cuja sintaxe é:

espere tempo t

A execução deste comando faz com que o MC2 pare seu processamento pelo intervalo indicado em *tempo*. A unidade deste comando é tempo, cujo símbolo é t. Caso a unidade **t** não seja colocada, o programa mostrará a seguinte mensagem de erro:

Você esqueceu a unidade.

Um exemplo do comando *espere* pode ser visto a seguir. Nele a luz 1 é acesa por um intervalo de três segundo e, então, é apagada.

```
# Exemplo:  
# Uso do comando espere  
Por favor  
    Ligue L1  
    Espere 3t  
    Desligue L1  
Obrigado
```

Como o MC2 “Fala” com Você?

Pode-se dizer que as luzes e os sons constituem a linguagem de nossos robôs. Por este motivo devemos explorar este recurso na elaboração dos programas para o MC2. Determinado Toque ou determinada seqüência de luzes podem ser usados, por exemplo, para indicar que o robô iniciou uma tarefa específica ou que ele está em dificuldades, pedindo ajuda.

Embora possamos criar nossa própria biblioteca de efeitos sonoros e visuais para usarmos nos programas, o MC2 vem com dois efeitos pré-programados. Estes efeitos podem ser ativados pelos comandos: **sim** e **não**.

Programação Básica

A idéia é que o comando **sim** venha ser usado para indicar situações nas quais, por exemplo, o robô cumpriu uma tarefa com sucesso, ou que ele está pronto para iniciar uma atividade etc. Esse comando **não** deve ser usado para indicar situações nas quais o robô não conseguiu executar uma tarefa ou quando ele estiver em dificuldades.

A sintaxe destes comandos é bastante simples, como pode ser visto a aqui:

sim

não

Para conhecer os efeitos gerados por estes comandos, digite o programa listado a seguir.

```
# Exemplo:  
# Uso do comando sim e não  
Por favor  
  Sim  
  Espera 3 t  
  Não  
Obrigado
```

Robótica e Eletrônica

Programação Básica

4. Movimentando seu Robô

Os comandos de movimento LEGAL foram desenvolvidos visando o controle de dois tipos de motores: motores de corrente contínua, conhecidos como Motores DC, e os servomotores.

Em aplicações em que a velocidade, o torque e a força do motor são importantes, usamos motores DC. Nas aplicações onde é mais importante o posicionamento, usamos servomotores.

Estas características fazem com que os comandos do LEGAL que são relacionados ao deslocamento do robô sejam escritos para serem usados com motores DC e os comandos de posicionamento angular, por exemplo, os usados na construção de braços mecânicos, foram desenvolvidos visando o controle de servomotores.

Posicionando os Servomotores

O MC2 pode controlar até quatro servomotores com quinze graus de precisão. O comando que torna este controle possível é o comando **servoX**. Onde X corresponde ao número do servomotor que se deseja controlar.

A sintaxe do comando servo é:

servo X ângulo

É importante ter em mente que o valor de ângulo deve ser um múltiplo de 15. Caso você esqueça, o **LEGAL** emitirá a seguinte mensagem de erro:

Este valor de ângulo não é válido.

Programação Básica

Um exemplo de programa usando o comando servo pode ser visto a seguir:

```
# Exemplo:
# Uso do comando servo.
Por favor
    Servo1 45
    Servo2 90
Obrigado
```

Ao executar este programa o MC2 posicionará o servo motor 1 em 45° e o servo motor 2 em 90°.

A PNCA oferece diferentes modelos de servomotores. Para garantir o correto funcionamento dos servomotores, é necessário configurar de maneira adequada o ambiente de programação. Para isto a PNCA fornece, juntamente com os servomotores, uma tabela com os valores de calibração específica para cada modelo de servomotor.

Para entrar com estes valores no **LEGAL** basta pressionar o botão AMBIENTE em qualquer tela do sistema e digitar os valores da tabela na janela relativa ao Ajuste dos Servos Motores, como mostra a figura a seguir:

15	000'	33	045'	49	090'	69	135'	81	180'
21	015'	39	060'	57	105'	75	150'		
27	030'	45	075'	63	120'	78	165'		

Ajuste dos servomotores.

Esta calibração é específica para cada modelo de servomotor, o que torna impossível, ao mesmo tempo, ligar diferentes servomotores em seu módulo de controle.

Programação Básica

Controlando os Motores DC



Montagem

Antes de continuar é importante que você construa o robô Zero, ou crie um robô equivalente, para testar os comandos que são ensinados a seguir.

O **LEGAL** permite que os motores DC sejam controlados de dois modos: o primeiro controla os movimentos do robô não permitindo o controle individual de cada motor; o segundo modo viabiliza este controle, porém, a tarefa de controlar os movimentos de seu robô fica mais complexa, uma vez que a direção e a velocidade de rotação de cada motor tenham que ser diretamente controladas.

Os principais comandos para o controle do movimento de seu robô são:

- **Frente:** este comando aciona os dois motores com a mesma velocidade e direção, fazendo o robô avançar,
- **Ré:** neste comando os motores também são acionados com a mesma velocidade e direção, porém o robô se move no sentido oposto ao do comando Frente;
- **Direita:** ao executar este comando o motor esquerdo gira para frente e o direito para traz fazendo com que o robô gire sobre seu próprio eixo para direita; e
- **Esquerda:** ao executar este comando os motores do robô giram no sentido oposto ao do comando Direita, fazendo com que ele gire para a esquerda.

Estes comandos têm algumas variações sintáticas; a primeira é aquela usada pelos botões de programação, definida como:

frente tempo **t**

ré tempo **t**


direita tempo **t**

esquerda tempo **t**

Programação Básica

Onde tempo indica a duração do comando em segundos. Se você esquecer de informar a duração do movimento, o LEGAL vai mostrar a seguinte mensagem de erro:

Qual é a duração deste comando?

 **Atenção** Por que o LEGAL controla os comandos de movimento usando o tempo e não a distância percorrida pelo robô?
Devido à flexibilidade do **Kit ALFA**, na qual os robôs podem ser construídos de diferentes formas e tamanhos, fica muito difícil controlar-se o deslocamento do robô medindo-se a distância percorrida em cada movimento. Tal dificuldade pode ser superada com o uso de sensores específicos para este fim. Entre em contato com os projetistas da PNCA para obter mais informações sobre este assunto.

Para não confundir o LEGAL, é necessário indicar a unidade de tempo utilizada. Neste caso, o tempo é medido em segundos. Caso você esqueça de indicar a unidade, o LEGAL emitirá a seguinte mensagem de erro:

Você esqueceu a unidade.

Programação Básica

No programa a seguir, o robô irá mover-se, pelo chão, em zigue-zaque.

```
# Exemplo:
# Movimento em zigue-zaque
Por favor
  Frente 8t
  Direita 2t
  Frente 3t
  Esquerda 2t
  Frente 8t
  Direita 2t
  Frente 3t
  Esquerda 2t
Obrigado
```

Programando



Você é capaz de fazer um robô que se movimenta formando um quadrado?

Uma maneira simples de controlar a velocidade com que seu robô se movimenta para frente e para trás é através das instruções: **rápido**, **normal** e **lento**.

A sintaxe para uso destas instruções é mostrada a seguir:

frente *tempo t* velocidade

ré *tempo t* velocidade

* Atenção



As instruções para o controle de velocidade não funcionam com os comandos: **direita** e **esquerda**.

Programação Básica

Observe o exemplo a seguir:

```
# Exemplo:  
# Controlando a velocidade do movimento.  
Por favor  
    Ligue L1  
    Frente 4t rápido  
  
    Ligue L2  
    Frente 4t lento  
  
    Ligue L3  
    Frente 4t normal  
Obrigado
```

Se for necessário que os motores funcionem sem parar, você poderá usar a instrução **sempre**, ou pode substituir a instrução do tempo em segundos. No exemplo a seguir o robô se mover sempre para frente. Observe que ele nunca irá virar para a direita.

```
# Exemplo:  
# Movimento para frente sem parar  
Por favor  
    Frente sempre  
    Direita 2t # este comando nunca será executado  
Obrigado
```

A sintaxe desta instrução é:

```
frente sempre  
ré sempre  
direita sempre  
esquerda sempre
```

Programação Básica

5. Comandos de Repetição

Comando Repita

Os comandos de repetição permitem que um comando (ou bloco de comandos) seja executado repetidamente. O **LEGAL** tem dois comandos de repetição:

- Repita e
- Enquanto.

O comando **Repita** permite que um comando ou bloco de comandos seja repetido um número específico de vezes. A sintaxe do comando Repita é:

```
Repita número vezes  
[  
  Comandos  
]
```

Onde *número* corresponde ao número de repetições desejado e comandos são os comandos que devem ser executados diversas vezes.

No exemplo a seguir as notas Dó, Ré, Mi, Fá, Sol, Lá e Si são geradas 20 vezes.

```
# Exemplo:  
# Toque 20 vezes  
Por favor  
Repita 20 vezes  
[  
  Toque Dó 3  
  Toque Ré 3  
  Toque Mi 3  
  Toque Fá 3  
  Toque Sol 3  
  Toque Lá 3  
]  
Obrigado
```

Programação Básica

Se não indicarmos o número de vezes que o bloco de comandos deve repetir, o **LEGAL** mostrará a seguinte mensagem:

Você esqueceu o número de vezes.

Outro erro comum é esquecermos dos símbolos [e], neste caso o **LEGAL** nos dará as seguintes mensagens:

Você esqueceu do [.

ou

Você esqueceu do].

O comando **Repita**, também aceita a instrução **sempre**, neste caso o bloco de comandos repetirá sem parar.

No exemplo a seguir a luz 1 piscará a cada um segundo, sem parar.

```
# Exemplo:  
# Pisca Luz 1  
Por favor  
Repita sempre  
[  
  Ligue L1  
  Espere 1t  
  Desligue L1  
  Espere 1t  
]  
Obrigado
```

Programação Básica

6. Definido o Comportamento de Seu Robô

Para que um robô faça alguma coisa útil, você deve programar o MC2 para gerar o comando correto para dada situação. O objetivo do programador é escrever programas que monitorem os sensores e enviem comandos para os atuadores produzirem movimentos adequados ao ambiente e ao estado do robô.

Muitas vezes para escrever tais programas devemos usar técnicas de inteligência artificial. As principais abordagens de Inteligência Artificial utilizadas em robótica móvel são as baseadas em planejamento e as reativas.

Os robôs que utilizam a abordagem baseada em planejamento, normalmente fazem a modelagem do mundo à sua volta e utilizam o modelo criado para planejar suas ações antes de executá-las. Esta solução é computacionalmente cara, uma vez que a modelagem do mundo e a atividade de planejamento consomem grande quantidade de recursos computacionais, além de ser impossível garantir que o modelo de mundo criado pelo robô corresponda perfeitamente ao mundo real. No outro extremo temos a abordagem reativa, na qual as ações do robô basicamente são determinadas pelos sensores; praticamente sem consumir qualquer recurso computacional.

Como você já deve ter imaginado isoladamente estas soluções são limitadas e de pouca utilidade, principalmente quando pensamos em construir um robô autônomo capaz de executar tarefas com alguma complexidade sobrevivendo no mundo real. Imagine um motorista que antes de executar uma ação faça uma análise cuidadosa do trânsito à sua volta, construindo um modelo mental do trânsito para, então, planejar detalhadamente suas próximas ações e somente então colocar seu plano em prática. Este nosso cuidadoso motorista não sobreviveria uma hora no trânsito de sua cidade!

Programação Básica

Devido ao tempo gasto entre uma observação e a ação ele poderia, por exemplo, concluir que se ele não parasse a 50 metros atrás ele atropelaria a velhinha. Se o nosso motorista pensasse de forma reativa, a velhinha ainda estaria viva. Porém um motorista puramente reativo, dificilmente seria capaz de sair de sua casa e chegar ao seu emprego, uma vez que ele não planejará seu caminho.

Enquanto não podemos construir robôs que de maneira natural combinem estas duas abordagens, vamos desenvolver robôs que funcionem de maneira reativa, seguindo os estímulos de seus sensores.

Na verdade, na natureza encontramos diversos exemplos de organismos que seguem esta abordagem. Entre estes podemos citar os insetos, que seguindo apenas seus instintos conseguem sobreviver e construir sociedades sofisticadas como as formigas e as abelhas.

Este estilo de programação é conhecida como programação baseada em Comportamento, em inglês *Behavior*.



Good Morning David ...

Com sua voz fria e monótona o super computador HAL-9000 cumprimenta o comandante David Bowman no início de mais um dia na nave espacial Discovery em sua longa missão até Júpiter. Além de educado e subserviente, HAL jogava xadrez e controlava toda a espaçonave, executando uma série de tarefas aborrecidas e monótonas. Seria uma excelente companhia para viagem não fosse o fato de ter enlouquecido e assassinado quase toda a tripulação. Criado por Arthur C. Clarke em seu livro 2001 Uma Odisséia no Espaço, HAL representa o sonho e o pesadelo do homem. Desde o início da Ciência da Computação temos sonhado em criar uma máquina inteligente para nos auxiliar e/ou substituir nas tarefas de nosso dia a dia ou simplesmente para nos fazer companhia em nossa jornada como única espécie inteligente neste pequeno planeta.

Programação Básica

Paradoxalmente, junto com este sonho vem o pesadelo, sempre explorado no mundo da ficção, de sermos subjugados e substituídos por nossa criatura, o complexo de Frankenstein onde esta se volta contra seu criador.

No mundo real, contudo, a Inteligência Artificial está muito longe de criar um HAL-9000 ou um David personagem central do filme A.I., embora muitos profetas da tecnologia antecipem que dentro de poucos anos teremos em um único chip a capacidade de processamento do cérebro humano e antes do final deste século este mesmo chip terá a capacidade de processamento toda a humanidade. O problema aqui não é de Hardware, mas sim de Software. Desenvolver programas de computador é uma tarefa bem mais complexa que projetar os próprios computadores (espero que meus amigos da engenharia me perdoem!).

O desenvolvimento do software sempre esteve atrás do hardware, e, em se tratando de Inteligência Artificial, existem inúmeros fatores que tornam seu desenvolvimento extremamente complexo. O principal deles é o simples fato de ninguém saber exatamente o que é inteligência. Você pode achar que ao jogar xadrez está usando sua inteligência muito mais do que quando ao cruzar uma rua se depara com Pitbull e resolve mudar o caminho ou mesmo fugir.

No entanto, a primeira tarefa é executada a algumas décadas pelos computadores, já a segunda envolve: identificar um animal, reconhecer uma situação de perigo e improvisar uma solução para esta situação, estas atividades são de uma complexidade até então intransponível para nossos amigos de silício.

Deixando de lado este enfoque da máquina inteligente e analisando a Inteligência Artificial como um ramo da Ciência da Computação que procura resolver problemas que não podem ser tratados, pelo menos diretamente, com o uso de modelos convencionais de programação, pode-se dizer que a Inteligência Artificial está repleta de sucessos e faz parte de nosso dia a dia.

Programas oriundos dos laboratórios de pesquisa em Inteligência Artificial auxiliam a dona de casa a lavar roupa (talvez você também tenha em casa uma máquina de lavar que utilize Lógica Fuzzy), controlam alguns sistemas do metrô, ajudam pilotos a traçar rotas e a manter as aeronaves dentro destas, nos ajudam em buscas na Internet, determinam rotas de entrega dentro das grandes cidades, etc.

Quanto ao nosso pesadelo de sermos superados por nossa criatura, acredito que nosso espírito criativo e nossa capacidade de adaptação garantem nossa superioridade, isso para não falar em nossa capacidade de amar, que jamais será reproduzida em uma máquina, mesmo que Stanley Kubrick e Steven Spielberg tente nos convencer do contrário.

Programação Básica

O **LEGAL** já vem com três tipos de comportamentos implementados, os quais você pode utilizar em seus programas:

- Comportamento para seguir uma faixa;
- Comportamento para seguir em direção a um estímulo e
- Comportamento para fugir de um estímulo.

Vamos aprender como usar cada um deles.

Comportamento para seguir uma faixa



Montagem

Para testar o comportamento de seguir faixa é necessário instalar os sensores de faixa em seu robô Zero. Observe a instalação dos sensores na base de seu robô na figura a seguir.



Uma vez instalado os sensores na base de seu robô é necessário conectá-los ao MC2. Para isto, você deve utilizar os conectores Zero + Faixa, para conectar o sensor da esquerda e o da direita respectivamente.

A distância entre os sensores deve ser adequada à largura da faixa que o robô deverá seguir. Esta distância deve ser o dobro da largura da faixa. Embora você possa criar diferentes circuitos para seu robô percorrer, a seguir pode ser visto um exemplo de circuito:

Dica

Para fazer um circuito no chão ou em uma mesa, você pode utilizar fita isolante ou qualquer outra fita adesiva colorida. É importante que a cor da fita se destaque da cor do piso, por isso em pisos escuros utilize fitas claras e quando o piso for claro use fitas escuras.

Programação Básica


O comando do LEGAL que ativa o comportamento de seguir faixa é **Siga Faixa**, sua sintaxe é:

Siga Faixa *cor tempo t*

Onde o parâmetro *cor* se refere à cor da faixa, podendo assumir os valores: **preta** ou **branca**.

No exemplo a seguir o robô irá seguir uma faixa preta por 20 tempos.

```
# Exemplo:  
# Seguir faixa  
Por favor  
  Siga Faixa preta 20t  
Obrigado
```

 **Atenção** Antes de testar o programa você deve calibrar os sensores de faixa. Ajuste o dispositivo para que a luz da caixinha acenda quando o sensor estiver sobre uma superfície clara e apague quando o sensor estiver sobre superfície escura. Para isso, gire o botão (potenciômetro) na parte superior da caixinha do sensor.

O comportamento **SigaFaixa** aceita a instrução **sempre** na definição do tempo de duração do comando, no exemplo a seguir o robô irá seguir a faixa para sempre.

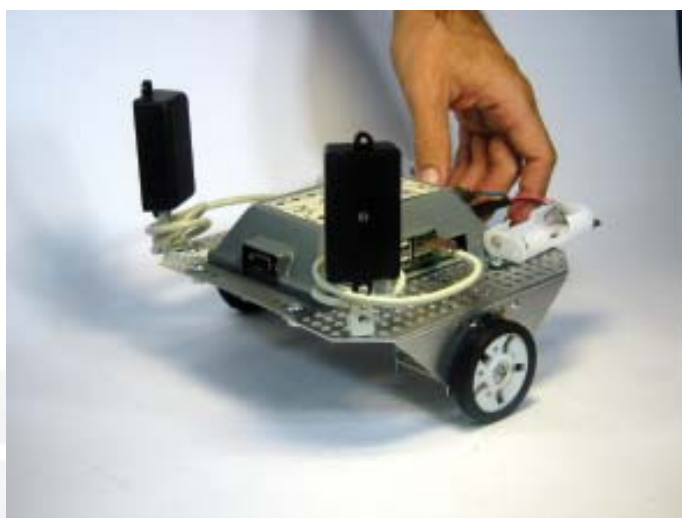
```
# Exemplo:  
# Seguir faixa até as pilhas acabarem.  
Por favor  
  Siga Faixa preta sempre  
Obrigado
```

Programação Básica

Comportamento para seguir um estímulo



Montagem Para testar o comportamento de seguir estímulo é necessário instalar os sensores apropriados em seu robô Zero. Nos próximos exemplos o robô será programado para seguir estímulos luminosos, para isto é necessário instalar os sensores de Luz como mostra a figura:



*** Atenção** Uma vez instalado os sensores na base de seu robô é necessário conectá-los ao MC2. Para isto, você deve utilizar os conectores S3 e S7, para conectar o sensor da esquerda e o da direita, respectivamente.

Uma vez instalados os sensores vamos programar o robô Zero para seguir uma fonte de luz, para isto basta utilizar o comando **Siga Luz**. A sintaxe do comando **Siga** é a seguinte:

siga estímulo tempo t

Onde estímulo deve ser substituído pelo estímulo que seu robô deve seguir, este pode ser: **Luz, Som e Calor**.

Programação Básica

Neste exemplo a luz será utilizada como estímulo. Assim, uma vez corretamente instalados os sensores, basta ensinar o seguinte programa ao seu robô:

```
# Exemplo:  
# Seguir Luz por 10 tempos.  
Por favor  
    Siga Luz 10t  
Obrigado
```

Durante dez segundos seu robô procurará e se moverá em direção à maior fonte luz que ele encontrar no ambiente. Ele poderá mover-se em direção a uma porta, uma janela ou uma lâmpada. Uma lanterna pode ser usada para guiar seu robô pela sala.

Trocando os sensores é possível criar robôs que se movimentem em direção a uma fonte de som ou de calor, conforme os exemplos a seguir:

```
# Exemplo:  
# Seguir Som por 10 tempos.  
Por favor  
    Siga Som 10t  
Obrigado
```

```
# Exemplo:  
# Seguir calor por 10 tempos.  
Por favor  
    Siga Calor 10t  
Obrigado
```

Programação Básica

Comportamento para fugir de um estímulo

Programar seu robô para fugir de um estímulo é tão fácil como foi programá-lo para seguir este estímulo. Para isto basta utilizar o comando **Fuja** estímulo. A sintaxe do comando **Fuja** pode ser conferida a seguir:

```
fuja estímulo tempo s
```

Onde *estímulo* pode ser substituído por: **Luz, Toque e Calor.**

No próximo exemplo o robô irá fugir da luz por 10 segundos procurando o lugar mais escuro da sala.

```
# Exemplo:  
# Fugir da Luz por 10 tempos.  
Por favor  
    Fuja Luz 10t  
Obrigado
```

Os comandos **Fuja** e **Siga**, também aceitam a instrução **sempre**, como pode se visto no exemplo a seguir.

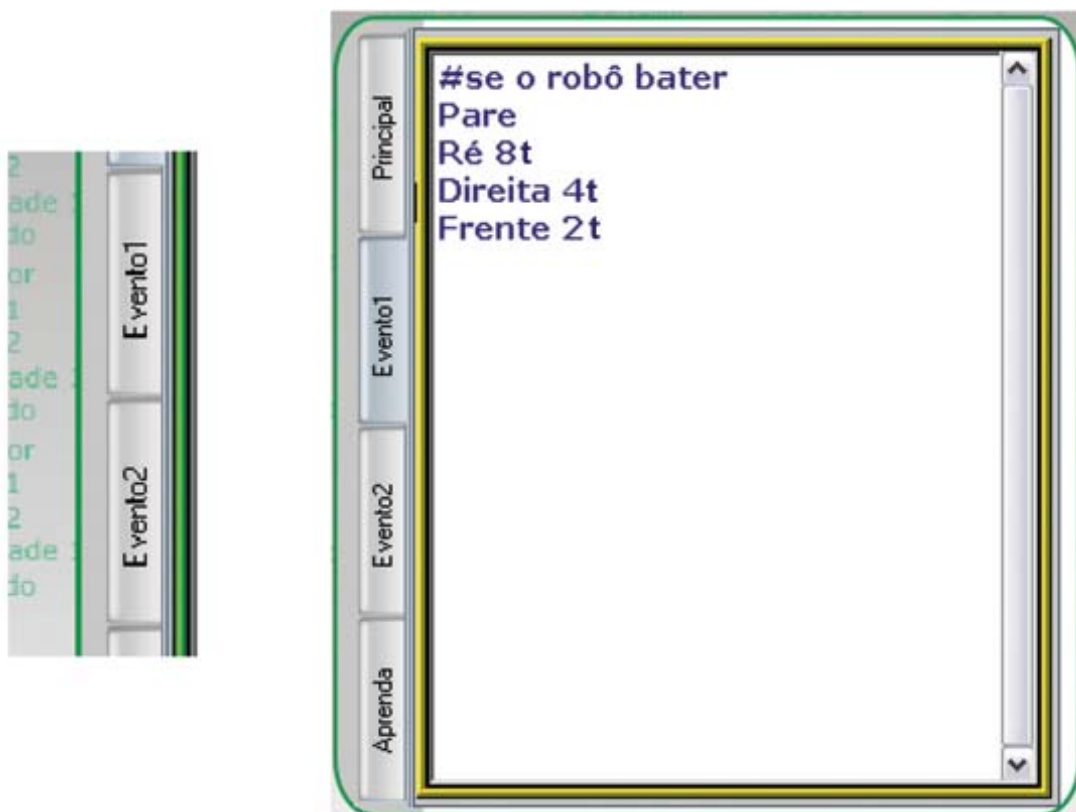
```
# Exemplo:  
# Fugir da Luz até acabar a energia das pilhas  
Por favor  
    Fuja Luz sempre  
Obrigado
```


Programação Básica

7. Programando Eventos

No ambiente de programação do **LEGAL** existem dois módulos para a programação de eventos, mas antes de ver como programá-lo, você precisa entender o que é evento para o **LEGAL**.

Na programação tradicional, a execução do programa tem início na primeira linha do código e segue um fluxo pré-determinado pelo programador. Na Programação orientada a eventos o funcionamento de seu programa é determinado por eventos externos. Estes eventos agem como gatilhos que disparam a execução de funções específicas definidas pelo programador, no caso do **LEGAL** estas funções são os códigos que estão nos módulos Evento1 e Evento2.



Módulos para programação de eventos.

Isso significa que em um programa desenvolvido com base nessa técnica, o funcionamento do robô depende dos eventos que ocorrerem, em outras palavras; dependem do que acontece no mundo em torno do robô.

Programação Básica

Embora qualquer leitura feita em um dos sensores de seu robô possa constituir um evento, nós da PNCA separamos dois conectores de sensores e desenvolvemos um tratamento especial para eles. Os conectores separados foram o S1 e o S5, ambos conectores para sensores digitais. O sensor S1 está associado ao Evento 1 e o sensor S5 está associado ao Evento 2.

Vamos supor, que você conecte um sensor de contato em S1: toda vez que este sensor for pressionado, um evento será gerado; neste caso, o MC2 interrompe o que estiver fazendo e irá executar os comandos escritos no módulo Evento 1. Quando acabar de executar estes comando, o MC2 voltará a fazer a tarefa interrompida.

Para entender melhor o conceito de eventos conecte um sensor de contato em S1 e outro em S5.



Sensores de contato instalados no MC2.

Programação Básica

Agora programe o MC2 com o seguinte programa no módulo principal:

```
# Exemplo:  
# Programa principal  
Por favor  
Repita sempre  
[  
Ligue L1  
Ligue L4  
Espere 1t  
Desligue L1  
Desligue L4  
Espere 1t  
]  
Obrigado
```

Como você já deve ter percebido, este programa fará com que as luzes 1 e 4 pisquem uma vez por segundo, sem parar. Agora digite os seguintes trechos de programas nos módulos Evento1 e Evento2:

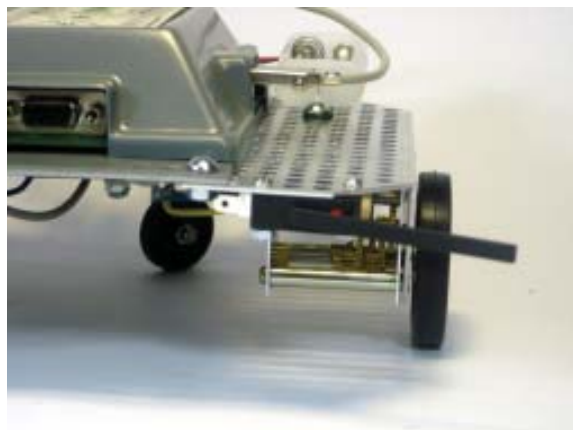
```
# Evento 1  
Ligue L2  
Ligue L3  
Desligue L5  
Desligue L6
```

```
# Evento 2  
Ligue L5  
Ligue L6  
Desligue L2  
Desligue L3
```

Programa o MC2 e veja o que acontece quando você pressiona os sensores de contato. Quando o sensor da esquerda é acionado o programa executa o código no módulo Evento1 acendendo as luzes dois e três e apagando as luzes cinco e seis. Quando o outro sensor é pressionado as luzes dois e três são apagadas e as luzes cinco e seis são acesas.

Programação Básica

Teste o funcionamento dos sensores de contato, instalando-os no robô ZERO.



Sensores de contato instalados no robô ZERO.

Programa o MC2 com o programa a seguir:

```
# Exemplo:
# Programa principal
Por favor
Frente sempre
Obrigado
```

```
# Evento 1
Pare
Ré 3t
Direita 2t
```

```
# Evento 2
Pare
Ré 3t
Esquerda 2t
```

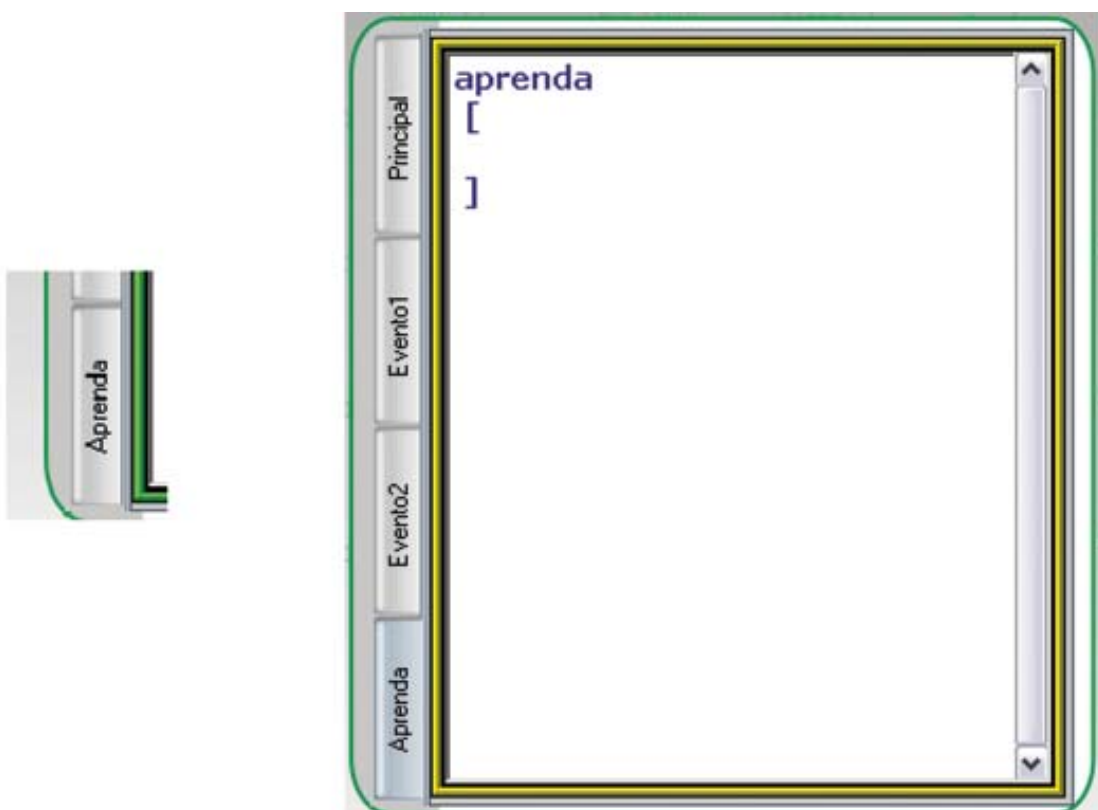
Uma vez programado, seu robô seguirá em frente até encontrar um obstáculo; então ele desviará e continuará a seguir em frente.

Programação Básica

8. Ensinando novos comandos para o LEGAL

A partir dos comandos primitivos, podemos criar outros comandos genericamente denominados sub-programas, sub-rotinas ou simplesmente rotinas. Uma vez programada, uma rotina pode ser executada da mesma maneira que um comando primitivo. De fato, é como se estivéssemos ensinando para o LEGAL um novo comando.

Como já vimos o LEGAL tem um módulo específico para a definição de rotinas: o módulo Aprenda.



Módulo aprenda.

Programação Básica

Módulo Aprenda.

Todas as rotinas criadas devem ser digitadas neste módulo. Quando você estiver no módulo Aprenda os botões de Novo, Abrir, Salvar e Salvar Como têm suas ações voltadas exclusivamente para este módulo.

Ou seja, se você estiver no módulo aprenda e clicar o botão Salvar, Somente os comandos que estiverem no módulo Aprenda é que serão gravados em um arquivo. Desta forma é possível desenvolver uma biblioteca com as rotinas criadas, as quais podem ser utilizadas em diferentes programas.



Os arquivos contendo rotinas são gravados com a extensão *.Bib* e não *.Leg*, como acontece com os demais arquivos do Legal.

Aqui está o formato geral de uma rotina em LEGAL:

```
aprenda nome
[
    comandos
]
```

Toda rotina precisa ter um *nome* que a diferencie dos demais comandos e rotinas. Uma rotina tem início no comando aprenda com um colchete aberto, [, e termina com um colchete fechado,]. Por exemplo:

```
# Rotina
aprenda pisca1
[
  Ligue L1
  Espere 1t
  Desligue L1
  Espere 1t
]
```

Programação Básica

* **Atenção** Existe quatro regras que você deve seguir na criação dos nomes de suas rotinas:

1. O nome de uma rotina deve iniciar com uma letra.
2. Para formar o restante do nome você pode utilizar letras e números.
3. Não pode haver espaços entre as letras que formam o nome de uma rotina.
4. Cada rotina deve ter um único nome. Não pode haver duas rotinas com o mesmo nome. Nem rotinas com nomes de comandos primitivos ou palavras reservadas.

Podemos definir diferentes rotinas no módulo Aprenda, no exemplo a seguir são definidas duas rotinas: pisca1 e pisca4.

```
# Rotina para piscar a luz 1.
aprenda pisca1
[
  Ligue L1
  Espere 1s
  Desligue L1
  Espere 1t
]

# Rotina para piscar a luz 4.
aprenda pisca 4
[
  Ligue L4
  Espere 1t
  Desligue L4
  Espere 1t
]
```

Para executar uma rotina basta usar seu nome em um programa no módulo principal ou em um evento, da mesma maneira que você os usa nos comandos primitivos. **No exemplo a seguir as rotinas pisca1 e pisca4** são utilizadas no programa principal juntamente como os demais comandos do **LEGAL**.

```
# Exemplo:
# Uso de rotinas
Por favor
repita sempre
[
  Pisque1
  Pisque4
]
Obrigado
```

Programação Básica

Com o passar do tempo nossas bibliotecas de rotinas irão crescer, quando isto acontecer é natural de passemos a utilizar as rotinas antigas para definirmos a novas.

O **LEGAL** permite tais construções, a única restrição é que uma rotina Somente poderá utilizar rotinas que estejam declaradas acima de sua definição. No exemplo a seguir a rotina **concorda** usa as rotinas **pisca1**, **pisca2** e **sons**.

```
# Rotina para piscar a luz 1.
aprenda pisca1
[
  Ligue L1
  Espere 1t
  Desligue L1
  Espere 1t
]

# Rotina para piscar a luz 4.
aprenda pisca 4
[
  Ligue L4
  Espere 1t
  Desligue L4
  Espere 1t
]
# Rotina Toque notas musicais
aprenda sons
[
  Toque dó 1
  Toque ré 1
  Toque mi 1
]
# Rotina concord
# esta rotina usa as anteriores
aprenda pisca 4
[
  Pisca 1
  Pisca 4
  sons
]
```


Programação Básica

Se modificarmos a ordem de declaração destas rotinas o programa não irá funcionar, como acontece com a versão da listagem a seguir.

```
# Este programa não funciona!!!

# Rotina concorda
aprenda pisca4
[
Pisca1 # o LEGAL não encontra esta rotina.
Pisca4
sons
]

# Rotina para piscar a luz 1.
aprenda pisca1
[
Ligue L1
Espere 1s
Desligue L1
Espere 1s
]

# Rotina para piscar a luz 4.
aprenda pisca4
[
Ligue L4
Espere 1s
Desligue L4
Espere 1s
]

# Rotina para Toque notas musicais
aprenda sons
[
Toque dó 1
Toque ré 1
Toque mi 1
]
```