LINGUAGEM

João Carlos Pandolfi Santana

→ **Exemplo da linguagem:**

```
INICIAR

        ENQUANTO_(NAO_CHEGOU_NO_OBJETIVO)_FOR_VERDADE:
                SE_(NAO_EXISTE_NADA_A_FRENTE)_FOR_VERDADE:
                        ANDAR_PARA_FRENTE
                FIM_DO_SE
                SENAO:
                        PARAR
                        SE_(EXISTE_OBJETO_A_DIREITA)_FOR_VERDADE:
                                VIRAR_PARA_ESQUERDA
                        FIM_DO_SE
                        SENAO:
                                VIRAR_PARA_DIREITA
                        FIM_DO_SENAO
                FIM_DO_SENAO
        FIM_DO_ENQUANTO

FIM_DO_PROGRAMA
```

→ **Explicação:**

“INICIAR” → Define o inicio do programa
“FIM_DO_PROGRAMA” → Define o fim do programa

<condicao> → Se comporta como um IF de linguagem de programação comum.

<verificao> → Retorna um resultado lógico para cada premissa
“NAO_EXISTE_NADA_A_FRENTE” → Se não tem nada a frente do robô
“EXISTE_OBJETO_A_FRENTE” → Se não tem objeto na frente do robô
“EXISTE_OBJETO_A_DIRETA” → Se não tem objeto ao lado direito do robô
“EXISTE_OBJETO_A_ESQUERDA” → Se não tem objeto ao lado esquerdo do robô

“SENSOR_LINHA_DIREITA_RECONHECEU” , “SENSOR_LINHA_MEIO_RECONHECEU” , “SENSOR_LINHA_ESQUERDA_RECONHECEU” → Retorna se determinado sensor de linha detectou alguma linha a baixo

“AGARROU_OBJETO” → Informa se agarrou o objeto ou não
“ESTA_PARADO” → Informa se o robô está parado
“ESTA_ANDANDO” → Informa se o robô está andando
“CHEGOU_NO_OBJETIVO” → Informa se o robô chegou no objetivo
“VERDADEIRO” → Retorna sempre verdadeiro
“FALSO” → Retorna sempre falso

<laco> → Se comporta como estruturas de laço em linguagens de programação comum.
“ENQUANTO_(”<verificacao>”)_FOR_VERDADE: ” <logica> “FIM_DO_ENQUANTO” → Funciona como um WHILE

“REPITA_(”<numero>”)_VEZES: ”<logica> “FIM_DO_REPITA” → Funciona como um FOR

<negacao> → Nega a premissa que esteja concatenada a ela

<comando> → chama uma função para execução de um determinado comando
“VIRAR_PARA_DIREITA” , “VIRAR_PARA_ESQUERDA” → Vira o robô para determinada direção
“ANDAR_PARA_FRENTE” → Faz o robô andar para frente.
“PARAR” → Faz o robô parar
“ANDAR_PARA_FRENTE_POR_(”<numero>”)_SEGUNDOS” → Faz o robô andar para frente por um numero determinado de segundos
“ABRIR_GARRA” → Abre a garra
“ABRIR_GARRA_(”<numero>”)_GRAUS” → Abre a garra determinados graus
“FECHAR_GARRA” → Fecha a garra
“FECHAR_GARRA_(”<numero>”)_GRAUS” → Fecha garra determinados graus
“LIGAR_MOTOR_DIRETA” , “LIGAR_MOTOR_ESQUERDA” → Liga determinado motor.
“DESLIGAR_MOTOR_ESQUERDA” , “DESLIGAR_MOTOR_DIREITA” → Desliga determinado motor.

→ **Gramática:**

<prog> ::= “INICIAR” <logica> “FIM_DO_PROGRAMA”

<logica>::= <logica> <logica>
| <laco>
| <condicao>
| <comando>

<condicao> ::= “SE_(”<verificacao>”)_FOR_VERDADE: ” <logica> ”FIM_DO_SE”
| “SE_(”<verificacao>”)_FOR_FALSO: ” <logica> ”FIM_DO_SE”
| “SENAO:” <logica> ”FIM_DO_SENAO”

<laco> ::= “ENQUANTO_(”<verificacao>”)_FOR_VERDADE: ” <logica>
“FIM_DO_ENQUANTO”
| “REPITA_(”<numero>”)_VEZES: ”<logica> “FIM_DO_REPITA”

<verificacao> ::= “NAO_EXISTE_NADA_A_FRENTE”
| “EXISTE_OBJETO_A_FRENTE”
| “EXISTE_OBJETO_A_DIRETA”
| “EXISTE_OBJETO_A_ESQUERDA”
| “SENSOR_LINHA_DIREITA_RECONHECEU”
| “SENSOR_LINHA_MEIO_RECONHECEU”
| “SENSOR_LINHA_ESQUERDA_RECONHECEU”
| “AGARROU_OBJETO”
| “ESTA_PARADO”
| “ESTA_ANDANDO”
| “CHEGOU_NO_OBJETIVO”
| “VERDADEIRO”
| “FALSO”
| <negacao><verificacao>

<negacao> ::=”NAO_”
| “E_FALSO_QUE_”
| “O_CONTRARIO_DE_”

<comando> ::= “VIRAR_PARA_DIREITA”
| “VIRAR_PARA_ESQUERDA”
| “ANDAR_PARA_FRENTE”
| “PARAR”
| “ANDAR_PARA_FRENTE_POR_(”<numero>”)_SEGUNDOS”
| “ABRIR_GARRA”
| “ABRIR_GARRA_(”<numero>”)_GRAUS”
| “FECHAR_GARRA”
| “FECHAR_GARRA_(”<numero>”)_GRAUS”
| “LIGAR_MOTOR_DIRETA”
| “LIGAR_MOTOR_ESQUERDA”
| “DESLIGAR_MOTOR_DIREITA”
| “DESLIGAR_MOTOR_ESQUERDA”

<numero> ::= [0-9]*