COMO PROGRAMAR SEU TIME

1) INSTALAÇÃO:

Instale o simulador da categoria SIMUROSOT da FIRA. O simulador é gratuito e está disponível para *download* no site da FIRA (www.fira.net) ou no site da competição (www.campeonato.gprufs.org).

2) ONDE COLOCAR AS ESTRATÉGIAS:

Ao instalar o simulador, será criada automaticamente a pasta "Strategy" no diretório "C:\". Dentro dessa pasta existem duas outras pastas:

- "\blue"
- "\yellow"

Na pasta "C:\Strategy\blue" devem ser colocadas as estratégias para o time azul e na pasta "C:\Strategy\yellow" as estratégias do time amarelo. O simulador só conseguirá ler a estratégias se elas estiverem na pasta correta.

3) ESTRATÉGIAS = ARQUIVOS ".TXT":

A estratégia de jogo deve ser escrita em um arquivo ".txt" usando a linguagem lingo, cuja sintaxe é bastante simples e será apresentada no curso preparatório ministrado para os alunos inscritos na competição. O arquivo da apresentação será disponibilizado para os alunos após o curso para que eles possam rever a sintaxe caso tenham alguma dúvida.

4) O CÓDIGO BASE:

A organização do campeonato irá fornecer um código base com algumas funções de controle primitivas que permitem controlar mais facilmente o movimento dos robôs do seu time. Esse código base será disponibilizado no site da competição (www.campeonato.gprufs.org) após o término do curso preparatório, onde o uso do código base e suas funções serão apresentados.

Para utilizar o código base, faça o *download* do mesmo no site do campeonato e substitua a pasta "C:\Strategy\" do seu computador pela pasta "Strategy" com os códigos base.

Na pasta do código base existe um código base para o time amarelo e um para o time azul. Os dois códigos possuem as mesmas funções de controle.

IMPORTANTE: Não troque os códigos do time azul com o do time amarelo, pois isso pode gerar erros na compilação da estratégia por parte do simulador.

IMPORTANTE 2: Sempre faça cópias de segurança antes de alterar seu código. Caso algo dê errado será mais fácil de voltar a trás.

5) SOBRE O CÓDIGO BASE:

Existem algumas regras importantes sobre a programação das estratégias. Leia-as com atenção nas regras da competição!!!

Eis aqui algumas das regras mais importantes:

- Só é permitido mandar comando de velocidade para os robôs utilizando as funções de controle disponibilizadas no código base;
- Não é permitido modificar ou atribuir valores para as variáveis do jogo (destacadas no começo do código base);
- Não é permitido passar comandos para os robôs do time adversário;
- Não é permitido alterar as funções de controle disponibilizadas no código base;
- Apenas modifique o código base nas partes indicadas, evitando assim que seu time seja punido ou mesmo eliminado da competição!

Essas e outras regras de programação serão verificadas antes e depois de cada jogo. Se alguma infração for cometida, a equipe poderá ser punida de acordo com as regras da competição e a gravidade da falta, a ser julgada pela comissão organizadora. Portanto tenha cuidado e siga as regras!!!!

6) SOBRE O SIMULADOR:

O uso do simulador é bastante simples e suas funções serão demonstradas durante a apresentação do curso preparatório. O arquivo da apresentação será disponibilizado pela organização após o curso. Com ele os alunos podem rever o que foi mostrado.

7) PLANTÃO DE DÚVIDAS:

A organização do campeonato disponibilizará monitores em diversos horários para tirar dúvidas sobre quaisquer questões do campeonato, das estratégias, do código base e/ou do simulador.

O horário do plantão de dúvidas será disponibilizado no site da competição após o curso preparatório.

Nos horários indicados, os monitores ficarão no Laboratório de Pesquisa do Departamento de Engenharia Elétrica da UFS (LABIC-DEL/UFS) à disposição dos alunos de todas as equipes.

Outra forma de tirar as suas dúvidas sem precisar ir à universidade é através do nosso e-mail (campeonato@gprufs.org) ou através do *Facebook* do campeonato.

8) AS VARIÁVEIS DO JOGO:

O jogo possui algumas variáveis que não devem ser alteradas, mas podem, e devem, ser utilizadas na elaboração das estratégias. A principal delas é a bola (variável "**ball**"). Além da bola temos os robôs e algumas coordenadas do campo, mostradas a seguir.

a) Variáveis dos robôs do time amarelo:

Imagem Do robô					
Nome da variável	**Y1**	**Y2**	**Y3**	**Y4**	**Y5**

b) Variáveis dos robôs do time azul:

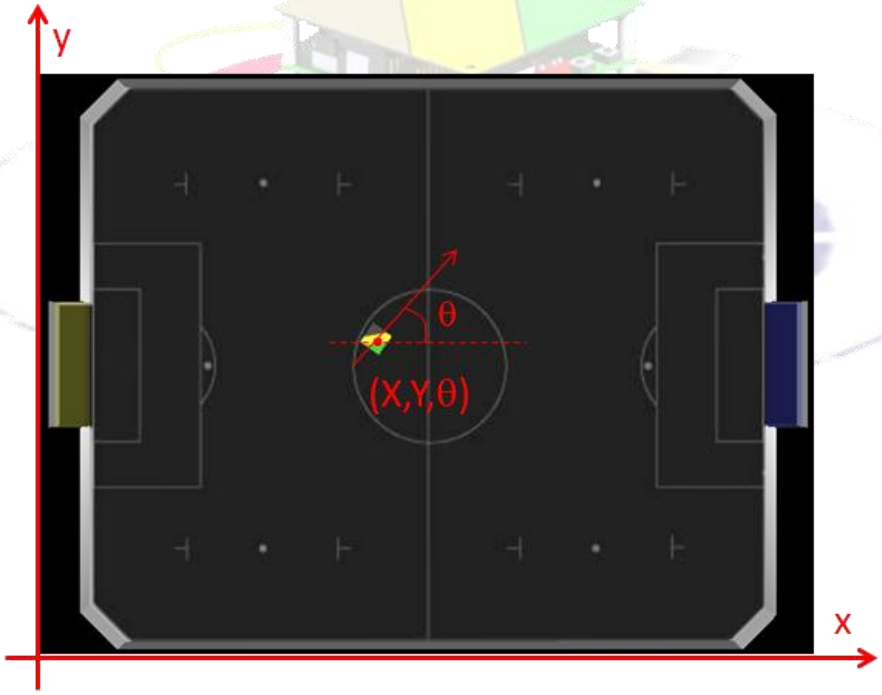
Imagem Do robô					
Nome da variável	**B1**	**B2**	**B3**	**B4**	**B5**

c) Variáveis que definem posições fixas do campo:

- **FTOP**: é a coordenada y da linha que delimita a parte de cima do campo, representada no plano cartesiano indicado na figura;
- **FBOT**: é a coordenada y da linha que delimita a parte de baixo do campo, representada no plano cartesiano indicado na figura;
- **GTOP**: é a coordenada y da trave superior do gol, representada no plano cartesiano indicado na figura;
- **GBOT**: é a coordenada y da trave inferior do gol, representada no plano cartesiano indicado na figura;
- **FLEFTX**: é a coordenada x da linha que delimita o campo pela esquerda, representada no plano cartesiano indicado na figura;
- **FRIGHTX**: é a coordenada x da linha que delimita o campo pela direita, representada no plano cartesiano indicado na figura.

9) COMO USAR AS VARIÁVEIS DO JOGO:

As variáveis do jogo podem ser utilizadas de diferentes formas, como mostrado no curso preparatório. Alguns exemplos são mostrados aqui:



- y4.pos.x ---> é a posição x do robô y4 no plano cartesiano do campo;
- y4.pos.y ---> é a posição y do robô y4 no plano cartesiano do campo;
- y4.rot.z ---> é o ângulo θ do robô y4 em relação ao eixo x do plano cartesiano do campo;
- A coordenada x da linha do meio do campo pode ser calculada por = (frightx + fleftx)/2.

10) O TIME AZUL E O TIME AMARELO:

Em cada partida, a equipe irá disputar dois tempos de jogo. No primeiro tempo jogará com uma cor e no segundo tempo jogará com a outra cor, sendo que o amarelo sempre ataca para a direita e o azul sempre ataca para a esquerda.

Tendo isso em conta, é importante lembrar às equipes que elas devem preparar as estratégias da forma mais geral possível, permitindo assim uma adaptação mais fácil da estratégia para as duas cores de time.

IMPORTANTE: Cuidado ao "traduzir" a estratégia do azul para o amarelo e vice-versa. Se você, por exemplo, esquecer um comando para um jogador do time azul no meio da sua estratégia para o amarelo sua equipe poderá ser desclassificada.

11) SINTAXE BÁSICA:

A sintaxe de uma linguagem de programação é a forma utilizada para definir os comandos dentro dessa linguagem. No caso do LINGO, que é a linguagem utilizada no nosso simulador, alguns comandos são muito importantes e úteis e são listados na tabela a seguir (outros comandos podem ser encontrados na apresentação do curso preparatório e no *help* do simulador).

COMANDO	SIGNIFICADO
+	Operação de adição
-	Operação de subtração
*	Operação de multiplicação
/	Operação de divisão
sqrt(*a*)	$\sqrt{a}$
--	Indica um comentário (utilize isso nas linhas que você quiser que o simulador não leia), por exemplo: --Essa linha não faz parte da estratégia!
>	Comparação: maior que
<	Comparação: menor que
=	Pode ser usado como comparação: igual a
=	Pode ser usado como operação de atribuição, utilizada para guardar um valor em uma variável qualquer, por exemplo: Tamanho = 5

12) AS FUNÇÕES DE CONTROLE:

As funções de controle são aquelas funções definidas no código base e que não devem ser alteradas. Essas funções são a única forma possível de movimentar os seus robôs respeitando as regras.

Na categoria SIMUROSOT da FIRA para mandar um robô se movimentar é utilizado o comando "velocity(robô,vr,vl)". A sua sintaxe não é detalhada pois, na nossa competição, **esse comando é proibido.**

A única forma de mandar velocidade para os robôs é utilizando as funções de controle definidas no código base. Essas funções e suas características são apresentadas a seguir.

a) Função vai_para:

- sintaxe: **vai_para(robô, posicao.x, posicao.y, aceleração)**
- faz com que o "**robô**" vá para a posição do plano cartesiano especificada por "**posicao.x**" e "**posicao.y**" com a "**aceleração**" indicada.
- "**robô**" = deve ser um robô do seu time.
- "**posicao.x**" e "**posicao.y**" = podem representar um ponto fixo (por exemplo, o meio do campo ou o centro do gol) ou até a bola, fazendo com que o robô vá até ela.
- "**aceleração**" = define o quão rápido o robô ganha velocidade. Deve ser um valor entre **0** (aceleração mínima) e **125** (aceleração máxima).
- A função **vai_para** possui uma velocidade mínima definida para o robô. Dessa forma, mesmo enviando um comando com "**aceleração = 0**" o robô, ainda assim irá se movimentar com a velocidade mínima definida na função.
- A velocidade mínima da função **vai_para** não pode ser alterada.

b) Função olhar_para:

- sintaxe: **olhar_para(robô, posicao.x, posicao.y)**
- faz com que o "**robô**" "olhe" para a posição do plano cartesiano especificada por "**posicao.x**" e "**posicao.y**", sem sair do seu lugar.
- "**robô**" = deve ser um robô do seu time.
- "**posicao.x**" e "**posicao.y**" = podem representar um ponto fixo (por exemplo, o meio do campo ou o centro do gol) ou até a bola, fazendo com que o robô vá até ela.

c) Função andar_frente:

- sintaxe: **andar_frente(robô, velocidade)**
- faz com que o "**robô**" ande para frente com a "**velocidade**" especificada.
- "**robô**" = deve ser um robô do seu time.
- "**velocidade**" = define a velocidade do robô. Deve ser um valor entre **0** (velocidade mínima) e **125** (velocidade máxima).

*** ONDE É A FRENTE DO ROBÔ??? é possível identificar a frente pela posição das rodas (pequeno círculo vermelho na lateral do robô) e pela cor do robô, com indicado no exemplo da figura abaixo.

d) Função andar_fundo:

- sintaxe: **andar_fundo(robô, velocidade)**
- faz com que o "**robô**" ande para trás com a "**velocidade**" especificada.
- "**robô**" = deve ser um robô do seu time.
- "**velocidade**" = define a velocidade do robô. Deve ser um valor entre **0** (velocidade mínima) e **125** (velocidade máxima).

e) Função girar_horario:

- sintaxe: **girar_horario(robô, velocidade)**
- faz com que o "**robô**" gire em torno do próprio eixo no sentido horário com a "**velocidade**" especificada.
- "**robô**" = deve ser um robô do seu time.
- "**velocidade**" = define a velocidade de rotação do robô. Deve ser um valor entre **0** (velocidade mínima) e **125** (velocidade máxima).

f) Função girar_antihorario:

- sintaxe: **girar_antihorario(robô, velocidade)**
- faz com que o "**robô**" gire em torno do próprio eixo no sentido anti-horário com a "**velocidade**" especificada.
- "**robô**" = deve ser um robô do seu time.
- "**velocidade**" = define a velocidade de rotação do robô. Deve ser um valor entre **0** (velocidade mínima) e **125** (velocidade máxima).

g) Outras funções:

- Existem outras funções definidas no código base que não as funções de controle. Estas funções são funções auxiliares utilizadas dentro das funções de controle aqui definidas.
- Os parâmetros, e para que servem essas funções, são definidos no próprio código base na forma de comentários.

13) PROGRAMA PARA ESCREVER A ESTRATÉGIA (opcional):

Uma vez que a estratégia e feita utilizando um arquivo ".txt", qualquer editor de bloco de notas pode ser utilizado para escrever as estratégias. No entanto, existem algumas vantagens em utilizar um editor um pouco melhor, mesmo dando um pouco mais de trabalho.

A principal vantagem é que, utilizando um bloco de notas padrão, um fragmento do seu código ficaria assim:

```
...
---------------------------- AQUI COMEÇA A SUA ESTRATÉGIA -----------------------
andar_frente(y2,10)
andar_fundo(y3,10)
...
```

Utilizando um editor de texto que "entenda" que o que você está escrevendo é um programa em LINGO, seu fragmento de programa ficaria assim:

```
...
---------------------------- AQUI COMEÇA A SUA ESTRATÉGIA -----------------------
andar_frente(y2,10)
andar_fundo(y3,10)
...
```

Apesar de a diferença ser apenas visual, as cores inseridas por um editor especializado podem ajudar muito a encontrar erros no seu programa e a escrever um código mais organizado, além de separar bem o que é comentário do que é estratégia de fato.

Infelizmente, um editor para LINGO não é algo muito fácil de encontrar. Por isso, fizemos uma pequena modificação para o programa "notepad++" que permite algumas facilidades visuais para a edição do seu programa.

Caso se interesse, o programa para editar o código com cores pode ser baixado no nosso site (www.campeonato.gprufs.org) juntamente com as instruções de instalação.

14) EXEMPLO DE ESTRATÉGIA:

Nessa seção final faremos um exemplo de uma estratégia básica, apenas para ilustrar a utilização das funções e mostrar que, mesmo não podendo utilizar o comando "velocity" (como explicado anteriormente), é possível criar funções que ajudam na organização do código.

Por questão de simplicidade, faremos um exemplo de estratégia para o time azul. Nesse caso utilizaremos o arquivo "codigobaseazul.txt" que se encontra na pasta "C:\Strategy\blue".

A parte destinada para a estratégia está destacada no código (ver figura abaixo).

```
-- Início da função que controla o time azul
-- (não altere a função strategyb)
On strategyb


--------------------------------------------------------------------------------
------------------------- INÍCIO DA ESTRATÉGIA DE JOGO -------------------------
--------------------------------------------------------------------------------
------- Funções de controle básicas (para uso nas estratégias dos alunos) --------
-- vai_para(robô, posicao.x, posicao.y, aceleração)
-- olhar_para(robô, posicao.x, posicao.y)
-- andar_frente(robô,velocidade)
-- andar_fundo(robô,velocidade)
-- girar_horario(robô,velocidade_ângular)
-- girar_antihorario(robô,velocidade_ângular)

---------------------------- AQUI COMEÇA A SUA ESTRATÉGIA ------------------------


---------------------------- AQUI TERMINA A SUA ESTRATÉGIA ------------------------

------------------------------------------------------------------------
--------------------- FIM DA ESTRATÉGIA DE JOGO ----------------------
------------------------------------------------------------------------


-- Fim da função que controla o time azul
End
```

Vamos fazer inicialmente o robô b2 ir atrás da bola com uma aceleração igual a 80. Para isso usaremos a função "vai_para" da seguinte forma:

- vai_para(b2, ball.x, ball.y, 80)

Faremos também o robô b3 e o robô b4 ficarem apenas olhando para a bola. para isso usaremos a função "olhar_para" da seguinte forma:

- olhar_para(b3, ball.x, ball.y)
- olhar_para(b4, ball.x, ball.y)

Como não foram mandados comandos para os outros robôs, eles irão continuar fazendo a última ação que foi ordenada. Como eles estavam parados continuarão parados quando começar o jogo.

A estratégia então ficaria da seguinte forma:

```
-- Início da função que controla o time azul
-- (não altere a função strategyb)
On strategyb


-------------------------------------------------------------------------------
------------------------- INÍCIO DA ESTRATÉGIA DE JOGO -------------------------
-------------------------------------------------------------------------------
------- Funções de controle básicas (para uso nas estratégias dos alunos) --------
-- vai_para(robô, posicao.x, posicao.y, aceleração)
-- olhar_para(robô, posicao.x, posicao.y)
-- andar_frente(robô,velocidade)
-- andar_fundo(robô,velocidade)
-- girar_horario(robô,velocidade_ângular)
-- girar_antihorario(robô,velocidade_ângular)

---------------------------- AQUI COMEÇA A SUA ESTRATÉGIA -------------------------

vai_para(b2, ball.x, ball.y, 80)
olhar_para(b3, ball.x, ball.y)
olhar_para(b4, ball.x, ball.y)


---------------------------- AQUI TERMINA A SUA ESTRATÉGIA -------------------------

-------------------------------------------------------------------------
---------------------- FIM DA ESTRATÉGIA DE JOGO ----------------------
-------------------------------------------------------------------------


-- Fim da função que controla o time azul
End
```

É possível ainda fazer com que, a depender de uma condição, o robô faça coisas diferentes. Por exemplo, vamos fazer com que o robô b5 corra atrás da bola com aceleração máxima se ela estiver no seu campo de ataque (campo da esquerda) e se ela estiver no seu campo de defesa (campo da direita) ele fique parado.

Para montar este código, primeiro precisamos saber onde é o meio de campo no jogo. Precisamos apenas da coordenada x do meio de campo.

Podemos calcular o valor do meio de campo a partir dos valores da parede da esquerda ("fleftx") e da parede da direita ("frightx"). Como o meio de campo fica exatamente na metade do campo, o valor da sua coordenada x pode ser obtido pela média entre "fleftx" e "fright". Assim, o código para isso seria:

- meio_de_campo = (fleftx + frightx)/2

Agora que temos a posição do meio campo em uma variável podemos utilizar um "if" para comparar com a posição da bola (também no eixo x). Então o código ficaria:

- if ball.x > meio_de_campo then -- se a bola estiver na defesa
- andar_frente(b5,0) -- o robô fica parado (anda com velocidade zero)
- else -- senão
- vai_para(b5, ball.x, ball.y, 125) -- vai para a bola com aceleração máxima
- end if -- final do "se"

Isso no nosso programa ficaria assim:

```
-- Início da função que controla o time azul
-- (não altere a função strategyb)
On strategyb


--------------------------------------------------------------------------------
------------------------- INÍCIO DA ESTRATÉGIA DE JOGO -------------------------
--------------------------------------------------------------------------------
------- Funções de controle básicas (para uso nas estratégias dos alunos) --------
-- vai_para(robô, posicao.x, posicao.y, aceleração)
-- olhar_para(robô, posicao.x, posicao.y)
-- andar_frente(robô,velocidade)
-- andar_fundo(robô,velocidade)
-- girar_horario(robô,velocidade_ângular)
-- girar_antihorario(robô,velocidade_ângular)

---------------------------- AQUI COMEÇA A SUA ESTRATÉGIA ------------------------

vai_para(b2, ball.x, ball.y, 80)
olhar_para(b3, ball.x, ball.y)
olhar_para(b4, ball.x, ball.y)

meio_de_campo = (fleftx + frightx)/2 -- calcula o valor do meio do campo
if ball.x > meio_de_campo then -- se a bola estiver na defesa
andar_frente(b5,0) -- o robô fica parado (anda com velocidade zero)
else -- senão
vai_para(b5, ball.x, ball.y, 125) -- vai para a bola com aceleração máxima
end if  -- final do "se"



---------------------------- AQUI TERMINA A SUA ESTRATÉGIA -----------------------

--------------------------------------------------------------------------
---------------------- FIM DA ESTRATÉGIA DE JOGO -------------------------
--------------------------------------------------------------------------


-- Fim da função que controla o time azul
End
```

Essa é a ideia de estratégia que vocês precisam. A partir daqui fica por conta da criatividade de vocês!!!

Testem as possibilidades! Pensem em todas as perguntas e variáveis que envolvem um jogo de futebol e tentem respondê-las. Como defender? Como atacar? Quantos zagueiros? E atacantes? Que comportamento um goleiro precisa ter?

Enfim... divirtam-se e elaborem um grande time de futebol de robôs para ganhar essa competição!!!!