

# MAC2166 – Introdução à Computação para Engenharia

## Escola Politécnica – Primeiro Semestre de 2012

### Angry Bixos

Terceiro Exercício-Programa (EP3)

Data máxima para entrega: Sábado, 19/05/2012 às 23:55

DEVE SER FEITO INDIVIDUALMENTE

## 1 Introdução

Nesse exercício programa vamos implementar um jogo que tem por objetivo “acertar” alvos (veteranos da Poli) atirando “bixos” com um “estilingue” (sim! essa é sua chance de descontar nos seus veteranos por tudo aquilo que eles fizeram com vocês na semana de recepção). Uma versão mais “velha” deste jogo e não menos interessante é o Angry Birds (<http://chrome.angrybirds.com/>) que consiste em acertar porcos atirando pássaros com um estilingue (utilize o navegador Chrome para jogar *online*). Qualquer semelhança entre os dois jogos é mera coincidência.

Recomendamos você a jogar o Angry Birds para um melhor entendimento deste EP :-).

A figura 1 ilustra uma situação do Angry Bixos. Em cada partida, a posição  $E = (x_E, y_E)$  do estilingue (poderoso o bastante para lançar um bixo) e  $A = (x_A, y_A)$  do alvo não variam. Você deve controlar a intensidade e sentido da velocidade inicial  $v$  com que um bixo é lançado ao alvo. O seu programa deve imprimir um gráfico mostrando a trajetória do bixo e indicar se o alvo foi atingido ou não. Em cada partida, você terá um número limitado  $nBix$  de bixos para atirar.

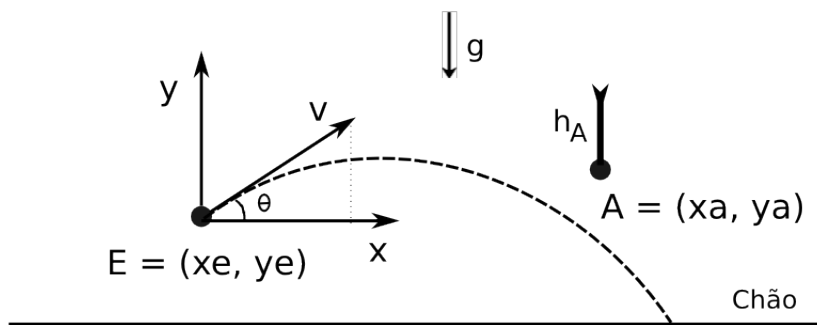


Figura 1: Representação do estilingue na posição  $E = (x_E, y_E)$  e do alvo na posição  $A = (x_A, y_A)$  com altura  $h_A$ . Um bixo é lançado do estilingue com uma velocidade inicial  $v$  e sofre a ação da gravidade  $g$ . A trajetória do bixo ilustra uma situação onde a velocidade  $v$  e/ou o ângulo  $\theta$  precisam ser ajustados para atingir o alvo.

Para lançar um dos  $nBix$  bixos utilizando o estilingue, você precisa fornecer um deslocamento inicial (quanto você vai puxar o elástico), e um ângulo (formado entre o vetor  $v$  e o eixo horizontal  $x$ ). Para simplificar o problema, vamos considerar apenas 2 dimensões como mostra a figura 1, ou seja, altura e distância, não vamos considerar profundidade.

O objetivo é derrubar o alvo usando no máximo os  $nBix$  bixos que você tem a disposição. A cada lançamento, o seu programa deve simular esse processo, calculando e exibindo graficamente (em forma de

texto usando o comando `printf`) a trajetória completa do bixo. A partir da visualização dessa trajetória, você pode modificar o deslocamento e o ângulo para tentar acertar o alvo.

O exemplo abaixo ilustra uma tela do jogo. O caractere 'E' representa a posição do estilingue. Os vários caracteres 'A' representam o alvo de altura  $h_A$ . O caractere 'o' representa a trajetória do bixo. Observe que o bixo atinge o chão nesse lançamento, onde permanece até o fim da simulação. Quando o bixo atinge o alvo, a parte atingida do alvo deve ser substituída por um caractere 'o'.

```

+---+---+---+---+---+---+---+---+
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
+---+ooooo---+---+---+---+---+---+
|   o   |o  |   |   |   |   |   |
|  o|   | o |   |   |   |   |   A
| o |   | o |   |   |   |   |   A
|   |   | o |   |   |   |   |   A
+---o---+---+---+---+---+---+---+
|   |   |   |   |   |   |   |   |
|o  |   |   |o  |   |   |   |   |
E   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
+---+---+---+o---+---+---+---+---+
|   |   |   |   |   |   |   |   |
|   |   |   |   | o |   |   |   |
|   |   |   |   |   |   |   |   |
|   |   |   |   | o |   |   |   |
+---+---+---+---+---+ooooo-----+

```

## 2 O que você deve fazer

Você deve escrever um programa em C que:

1. pergunta ao usuário em qual modo devem ser apresentados os resultados, se no modo **depuração** ou **jogo**. O usuário deve digitar 0 para o modo **jogo** e 1 para **depuração**. Esses modos são explicados mais adiante.
2. chama a função `LeiaParametros`, que lê um arquivo de entrada de nome “entrada.txt” (um exemplo deste arquivo será fornecido juntamente com este enunciado) para carregar os valores iniciais dos seguintes parâmetros que definem uma partida (você deve assumir que todos os valores estão em uma unidade apropriada, por exemplo, no SI):
  - $y_E, v_{max}$ : 2 reais que definem a posição do estilingue e a velocidade máxima com que um bixo pode ser arremessado;
  - $y_A, h_A$ : 2 reais que definem a posição do alvo e sua altura;

- *dist*: real positivo que define a distância  $x_A - x_E$  entre o alvo e o estilingue.
- *nBix*: inteiro que define o número de bixos que podem ser lançados;
- *nLin, nCol*: 2 inteiros que definem as dimensões do gráfico a ser impresso (ambos devem ser múltiplos de 5);
- *nUni*: real utilizado como fator de escala das alturas (número de unidades de altura por linha);
- *g*: real negativo que define a aceleração da gravidade.

A função `LeiaParametros` é dada e se encontra dentro do arquivo `esqueletoEP3.c`, que você **deve** utilizar para iniciar e desenvolver a sua solução.

3. Após o carregamento dos valores iniciais dos parâmetros que definem a partida, seu programa deve imprimir a configuração inicial do jogo, ou seja, as posições relativas do estilingue e do alvo, em um gráfico com  $nLin + 1$  linhas e  $nCol + 1$  colunas, pois o gráfico deve sempre exibir a linha 0 (chão) e a coluna 0, onde se encontra o estilingue  $E$ . O alvo estará sempre na coluna  $nCol$ . Para facilitar a visualização do arremesso, o seu gráfico deve mostrar linhas auxiliares (como nos exemplos), a cada 5 linhas, assim como colunas auxiliares a cada 5 colunas. Você pode assumir que os valores de  $nLin$  e  $nCol$  são múltiplos de 5.
4. Em seguida, enquanto existirem bixos a serem lançados, seu programa deve ler 2 números reais que definem um lançamento (deslocamento e orientação) e imprimir o gráfico mostrando a trajetória do bixo.

Caso o alvo tenha sido atingido, o jogador é informado de que o alvo foi atingido e o jogo termina. Caso contrário, o jogador é informado de que o alvo não foi atingido e o jogo continua até que todos os bixos tenham sido lançados.

Veja ao final desse enunciado alguns exemplos para entender melhor o que acontece durante uma partida de **Angry Bixos**.

## 2.1 Como calcular a trajetória de um bixo

Para o lançamento de um bixo, seu programa deve ler um real *des* (deslocamento) e outro real *ori* (orientação) e calcular a trajetória do bixo da seguinte forma:

- o deslocamento  $des \in [0; 1]$  indica quanto você puxa o elástico. Assim, quanto maior o deslocamento, maior a velocidade inicial do bixo. O arquivo de entrada define a velocidade máxima ( $v_{max}$ ) com que o bixo pode ser lançado do estilingue. Assim, a velocidade de lançamento pode ser calculada como:

$$v = des \times v_{max}$$

- a orientação  $ori \in [-1; 1]$  deve ser transformada em um ângulo  $\theta$  em radianos usando a função `Converte` descrita mais adiante. A velocidade inicial  $v$  do bixo pode ser assim decomposta em  $(velX, velY) = (v \times \cos(\theta), v \times \sin(\theta))$
- o ângulo  $\theta$  varia de  $[-\frac{\pi}{2}; \frac{\pi}{2}]$ , i.e., no intervalo normalizado  $[-1; 1]$ .
- assumindo que a velocidade  $velX$  é constante (bixos de cabeça raspada apresentam um baixo coeficiente de resistência ao ar), calcule o tempo total  $T$  que o bixo levaria para chegar ao alvo  $A$ , ou seja, o tempo para percorrer a distância  $dist = x_A - x_E$ , fornecida pelo arquivo de entrada. Você deve ter cuidado quando  $velX = 0$  ou muito próximo de zero. Você deve assumir que a velocidade horizontal ( $velX$ ) mínima do bixo é  $10^{-4}$ .

- Para calcular a trajetória do bixo,  $T$  deve ser dividido pelo número de intervalos ( $nCol$ ) do gráfico. Esse valor  $dt = T/nCol$  define o passo da simulação, ou seja, quanto tempo o bixo leva para chegar na próxima coluna do gráfico. Assim, cada coluna do gráfico corresponde a um passo da simulação, ou um instante no tempo onde a altura do bixo precisa ser calculada.
- Dado que a posição inicial do bixo é  $E$  e sua velocidade inicial é  $(velX, velY)$  você deve calcular a altura do bixo para todos os passos (ou instantes)  $t$  da simulação (ou seja, para  $nCol$  passos).
- A altura do bixo no início da simulação ( $t = 0$ ) é  $y_E$ . A altura para um outro instante  $t > 0$  deve ser calculada da seguinte forma:

$$y^t = y^{t-1} + velY^{t-1} \times dt + 0.5 \times g \times dt \times dt$$

onde  $y^t$  corresponde a altura  $y$  no instante  $t$ , e  $g$  é a aceleração da gravidade, que é lida do arquivo de entrada.

- A velocidade  $velY^t$  deve ser calculada pela equação:

$$velY^t = velY^{t-1} + g \times dt$$

Observe que após o bixo se espatifar no chão, ele permanece lá. Esse caso deve ser tratado pela função

`void AtualizaPosicaoEVelocidade( float *y, float *velY, float dt, float g )`  
definida mais adiante e que você precisa implementar e utilizar em seu programa.

## 2.2 Como imprimir a trajetória

O gráfico da trajetória do bixo deve ser impresso como mostrado nos exemplos fornecidos neste enunciado. O gráfico deve ter exatamente  $nLin + 1$  linhas (de 0 a  $nLin$ ) por  $nCol + 1$  colunas (de 0 a  $nCol$ ), onde os valores de  $nLin$  e  $nCol$  são lidos do arquivo “entrada.txt”.

O seu gráfico deve apresentar ”uma grade” com moldura para facilitar a visualização da trajetória. A cada 5 linhas, deve ser exibida uma linha da grade (o mesmo intervalo deve ser utilizado para as colunas), como ilustrado nos exemplos desse enunciado.

A primeira coluna do gráfico deve indicar a posição do estilingue ( $y^0 = y_E$ ) usando o caractere 'E'. As demais colunas devem indicar a altura do bixo em cada instante  $t$  (ou passo da simulação), usando o caractere 'o'.

O fator de escala das colunas é fornecido pelo arquivo de entrada (parâmetro  $nUni$ , descrito anteriormente). Para um instante  $t$ , a altura  $y^t$  do bixo deve ser convertida pela  $nUni$  usada no gráfico da seguinte forma:

$$altura\_inteira = Arredonda(y^t/nUni)$$

A última coluna do gráfico deve mostrar o alvo usando o caractere 'A', de  $y_A$  até  $y_A + h$ , e também a altura por onde o bixo passou no instante  $T$  (final da simulação).

## 2.3 Quando o jogo termina?

Seja  $y^T$  a altura (posição vertical) do bixo no final da simulação ( $t = T$ ). O alvo é atingido quando a seguinte condição é verdadeira:

$$y_A < y^T < y_A + h$$

A partida deve terminar quando o alvo for atingido, ou após o lançamento do  $nBix$  bixos.

## 2.4 Modos de operação

Para que você possa testar o seu programa, você deve implementar dois modos de operação, um de *depuração* e outro de *jogo*.

No modo *depuração*, seu programa deve imprimir os valores lidos do arquivo “entrada.txt”, os valores iniciais do ângulo em radianos, o cosseno do ângulo, o seno do ângulo, a velocidade inicial e suas componentes horizontal ( $velX$ ) e vertical ( $velY$ ), o tempo total para que o bixo atinja o alvo ( $T$ ) e o incremento temporal ( $dt$ ). Além disso, para cada instante de tempo  $t$ , o programa deve imprimir à direita do gráfico, o número da linha correspondente, o número de vezes que o bixo “corta” o eixo  $x$  e as DUAS (e somente duas) alturas mais à esquerda.

Veja o exemplo a seguir (na próxima página) para uma melhor compreensão. A coluna *lin* indica a linha correspondente. A coluna *nInters* indica o número de vezes que o bixo “cortou” aquela linha. As colunas  $y1$  e  $y2$  representam respectivamente, as alturas do bixo na posição mais à esquerda. Note que, como o gráfico é discretizado, ou seja, uma linha do gráfico corresponde a um  $\Delta altura$ , é normal que a trajetória do bixo cruze e/ou se mantenha em uma mesma linha do gráfico, porém, com valores reais  $y1$  e  $y2$  diferentes.

Por exemplo, veja a linha 12 do gráfico que tem fator de escala  $nUni = 2$ . Na linha 12,  $y1 = 24.1551$  (altura do bixo no instante de tempo igual a 7 do gráfico, ou seja, oitava coluna, já que a primeira coluna é o instante de lançamento). Pelo nosso processo de discretização,  $Arredonda(y1/nUni) = Arredonda(12.0775) = 12$ , ou seja, a altura do bixo de 24.1551 deve ser impresso na linha 12. Outro exemplo na mesma linha 12 é a altura do bixo no instante de tempo 27. Neste caso,  $y2 = 24.8298$ . Discretizando este valor usando nosso critério,  $Arredonda(y2/nUni) = Arredonda(12.4149) = 12$ , ou seja, a altura do bixo de 24.8298 deve ser impresso na linha 12 também, como mostra a figura. Note também que o bixo “corta” esta linha (linha 12) 3 vezes mas somente as DUAS primeiras alturas foram impressas. Caso o bixo não “corte” a linha, imprima a altura do bixo como igual a -1.0000.

Observe que, para imprimir o gráfico corretamente, a primeira linha a ser impressa deve ser a mais alta, e a última a mais baixa (linha zero).

Grafico do lancamento com deslocamento/orientacao = [ 0.6000, 0.4000]

Condicoes iniciais desse lancamento:

```

ang (rad)   =   0.6283   cos =   0.8090   sin =   0.5878
velocidade  =   30.0000   velX =   24.2705   velY =   17.6336
tempo total =    4.5323   incremento temporal =    0.1133 entre colunas

```

	lin	nInters	y1	y2
+---+---+---+---+---+---+---+---+	25	0	-1.0000	-1.0000
	24	0	-1.0000	-1.0000
	23	0	-1.0000	-1.0000
	22	0	-1.0000	-1.0000
	21	0	-1.0000	-1.0000
+---+---+---+---+---+---+---+---+	20	0	-1.0000	-1.0000
	19	0	-1.0000	-1.0000
	18	0	-1.0000	-1.0000
	17	0	-1.0000	-1.0000
	16	0	-1.0000	-1.0000
+---+---+---oooooooo---+---+---+---+	15	9	29.2104	29.6485
ooo     ooo	14	6	27.2027	27.9875
oo       oo       A	13	4	25.2865	26.3024
o         oo     A	12	3	24.1551	24.8298
oo           o     A	11	3	21.5457	22.9082
+---o+---+---+---+---+---oo---+---+	10	3	20.0676	20.9446
o             o	9	2	18.4740	17.7768
o             o	8	2	16.7649	16.0196
Eo               o	7	3	13.0000	14.9402
o	6	1	12.1586	-1.0000
+---+---+---+---+---+---+---+o---+	5	1	10.0547	-1.0000
o	4	1	7.8353	-1.0000
o	3	1	5.5004	-1.0000
o	2	1	3.0499	-1.0000
	1	0	-1.0000	-1.0000
+---+---+---+---+---+---+---+---o	0	1	0.4839	-1.0000

Voce nao acertou o alvo nessa jogada.

No modo *jogo*, seu programa deve imprimir apenas o gráfico e as mensagens que indicam o estado da partida (se o jogador acertou ou não). Dê uma olhada nos exemplos no final deste enunciado para entender melhor os valores e as mensagens que devem ser impressas em cada modo.

## 2.5 Funções auxiliares

Para resolver o EP3, você deve implementar e utilizar **obrigatoriamente** em seu programa as seguintes funções auxiliares:

1. `float Seno( float x, float eps )`

A função `Seno` recebe um real  $x$  e um real  $eps$  e devolve a aproximação do **seno** de  $x$  em radianos, calculando todos os termos da série:

$$Seno(x) = \frac{x}{1!} - \frac{x^3}{3!} + \frac{x^5}{5!} - \dots + (-1)^k \frac{x^{2k+1}}{(2k+1)!} + \dots$$

até que  $|(-1)^k \frac{x^{2k+1}}{(2k+1)!}| < eps$ . Inclua esse último termo no cálculo de `Seno(x)`.

2. `float Cosseno( float x, float eps )`

A função `Cosseno` recebe um real  $x$  e um real  $eps$  e devolve a aproximação do **cosseno** de  $x$  em radianos, calculando todos os termos da série:

$$Cosseno(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots + (-1)^k \frac{x^{2k}}{(2k)!} + \dots$$

até que  $|(-1)^k \frac{x^{2k}}{(2k)!}| < eps$ . Inclua esse último termo no cálculo de `Cosseno(x)`.

3. `float Converte( float x )`

A função `Converte` recebe um número real  $x$  no intervalo  $[-1; 1]$  e devolve o número  $x$  re-escalado no intervalo  $[-\frac{\pi}{2}; \frac{\pi}{2}]$ .

4. `void AtualizaPosicaoEVelocidade( float *y, float *velY, float dt, float g )`

A função `AtualizaPosicaoEVelocidade` recebe 4 números reais  $*y$ ,  $*velY$ ,  $dt$  e  $g$  que representam a posição vertical atual do bixo, a atual componente vertical da velocidade do bixo, o  $dt$  (passo da simulação) e a aceleração da gravidade, respectivamente, e devolve em  $*y$  e  $*velY$  a posição atualizada do bixo e a componente vertical da velocidade após  $dt$ , respectivamente.

Observe que o bixo não deve passar do chão. Assim, se a altura ( $*y$  = posição vertical) do bixo se tornar negativa, a função deve devolver zero em  $*y$  e em  $*velY$ .

5. `int Arredonda( float x )`

A função `Arredonda` recebe um real  $x$  e devolve o número inteiro arredondado. A regra é, se o primeiro algarismo decimal for maior ou igual a cinco, arredonda para cima. Caso contrário, arredonda para baixo.

Exs.:

$2.53 \rightarrow 3$

$0.9 \rightarrow 1$

$5.42341 \rightarrow 5$

6. `void ImprimaCondicaoInicial( int nLin, int nCol, float nUni, float yE, float yA, float hA )`

A função `void ImprimaCondicaoInicial` recebe dois inteiros  $nLin$  e  $nCol$  que são respectivamente os números de linhas e colunas do gráfico a ser impresso, e quatro reais  $nUni$ ,  $yE$ ,  $yA$  e  $hA$  que são o fator de escala das alturas, a posição do estilingue, a posição do alvo e sua altura, respectivamente. Esta função deve imprimir a condição inicial do jogo (veja os exemplos no final do enunciado).

7. Você pode implementar outras funções se desejar, mas deve utilizar **obrigatoriamente** essas 6 (seis) funções anteriores.

8. Utilize também a função `LeiaParametros`, já implementada e disponível no arquivo `esqueletoEP3.c`.

### 3 O que você deve entregar

Você deve entregar, na página da disciplina, um programa em C (código fonte) que implemente o jogo Angry Bixos, como descrito nesse enunciado. Você deve seguir as instruções sobre entrega de EPs disponível em <http://www.ime.usp.br/~mac2166/infoeps/>.

### 4 Dicas e Observações

- Não deixe para fazer o EP na última hora.
- Você deve implementar e utilizar em seu programa, OBRIGATORIAMENTE, as funções definidas neste enunciado.
- Você NÃO pode alterar os protótipos das funções obrigatórias.
- Você NÃO DEVE utilizar funções da biblioteca `math.h`, e também não deve utilizar nenhuma função ou recurso do C não visto em aula (Exs.: vetores e matrizes).
- A inconsistência dos dados não será avaliada e nem testada. Em outras palavras, nenhum monitor ou professor testará seu EP com ângulos fora do intervalo  $[-1, 1]$ , ou deslocamento fora do intervalo  $[0, 1]$ , por exemplo.
- Você pode utilizar um *eps* constante e igual a  $10^{-4}$  para todas as funções. Para isso, crie uma constante `#define EPS 0.0001` e utilize esse *eps* em todas as chamadas.
- Você pode definir o valor de  $\pi$  como igual a 3.1415954. Para isso, crie uma constante `#define PI 3.1415954` e utilize essa constante em seu programa.
- Para visualizar o gráfico corretamente, você precisa escolher uma fonte de caracteres que apresente uma largura uniforme para todos os caracteres, por exemplo, a fonte “Andale Mono”.
- Para padronizar os gráficos, você deve utilizar o caractere ‘-’ nas linhas auxiliares, ‘|’ nas colunas auxiliares, e ‘+’ nos cruzamentos. Os valores de *nLin* e *nCol* serão sempre múltiplos de 5.
- Teste e brinque com o seu programa “inventando” outros arquivos “entrada.txt”.
- **Esse exercício deve ser feito individualmente e você é responsável por qualquer cópia realizada de seu programa (e partes dele). Casos de plágio serão tratados de acordo com as regras descritas na folha de informações gerais.**
- Você não será pego por plágio pelo fato de usar o programa `esqueletoEP3.c`.



## 5 Exemplo de execução

### 5.1 Exemplo de jogo em que o jogador NÃO acerta o alvo

Voce deseja ligar o modo de depuracao?

Digite 1 (sim) ou 0 (nao): 0

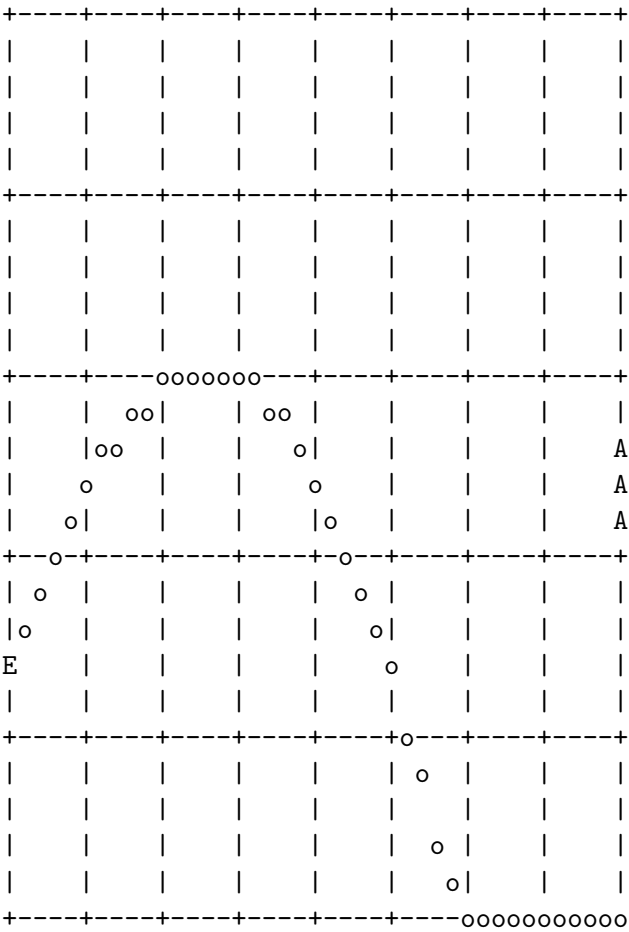
Para fazer um lançamento, digite um deslocamento (no intervalo  $[0,1]$ ) e uma orientacao (no intervalo  $[-1, 1]$ )

## Condicao Inicial

[illegible]

```
====> Faca o lancamento 1 / 5: .5 .5
```

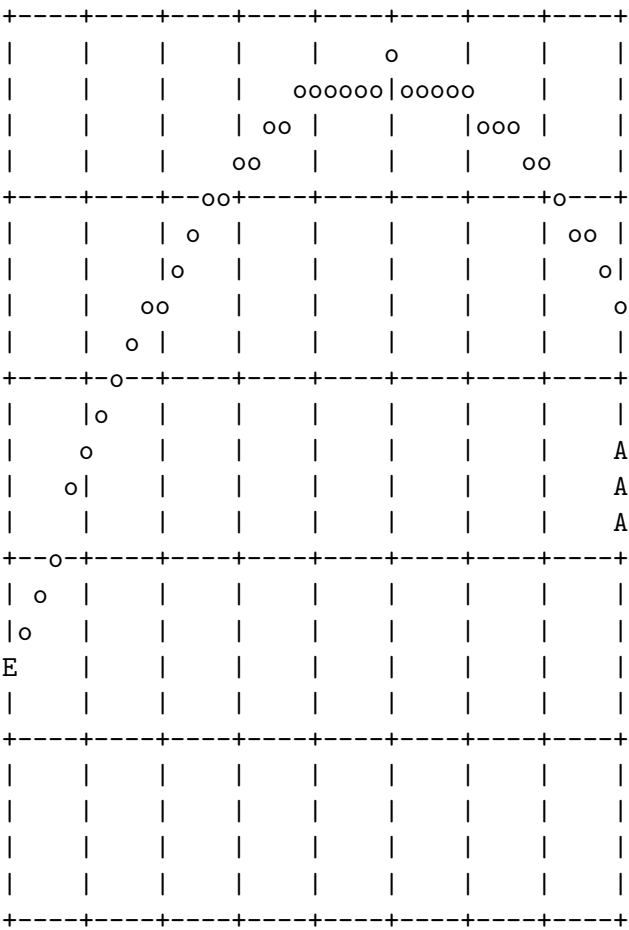
Grafico do lancamento com deslocamento/orientacao = [ 0.5000, 0.5000]



Voce nao acertou o alvo nessa jogada.

====> Faca o lancamento 2 / 5: 0.7 0.5

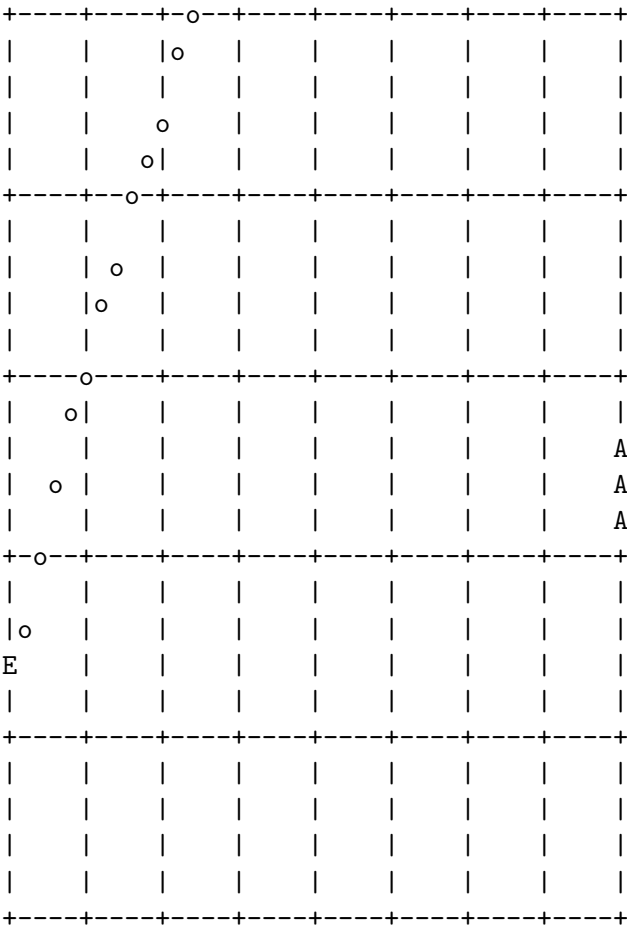
Grafico do lancamento com deslocamento/orientacao = [ 0.7000, 0.5000]



Voce nao acertou o alvo nessa jogada.

====> Faca o lancamento 3 / 5: 0.8 0.6

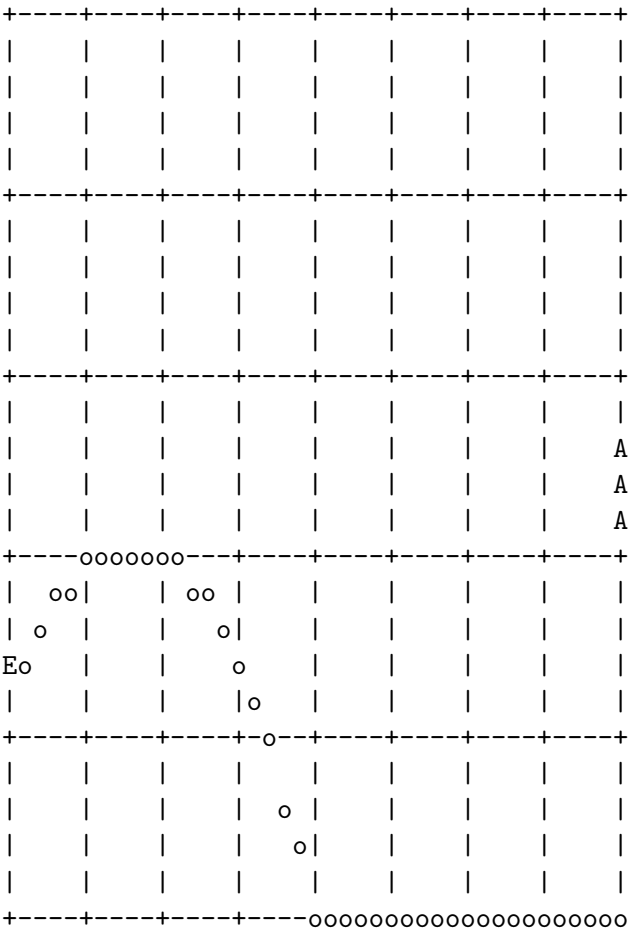
Grafico do lancamento com deslocamento/orientacao = [ 0.8000, 0.6000]



Voce nao acertou o alvo nessa jogada.

====> Faca o lancamento 4 / 5: 0.4 0.4

Grafico do lancamento com deslocamento/orientacao = [ 0.4000, 0.4000]



Voce nao acertou o alvo nessa jogada.

====> Faca o lancamento 5 / 5: 0.6 0.4

[illegible]

14

## 5.2 Exemplo de jogo em que o jogador acerta o alvo e mostra informações no modo depuração

Voce deseja ligar o modo de depuracao?

Digite 1 (sim) ou 0 (nao): 1

Modo de depuracao ligado!

Dados de entrada:

yE = 13.0000 vMax = 50.0000

yA = 22.0000 hA = 6.0000

dist = 110.0000

nBix = 5

nLin = 25 nCol = 40

nUni = 2.0000

g = -9.0000

Para fazer um lancamento, digite um deslocamento (no intervalo [0,1])  
e uma orientacao (no intervalo [-1, 1])

Condicao Inicial

```

+---+---+---+---+---+---+---+---+
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+

```

====> Faca o lancamento 1 / 5: 0.5 0.5

Grafico do lancamento com deslocamento/orientacao = [ 0.5000, 0.5000]

Condicoes iniciais desse lancamento:

ang (rad) = 0.7854 cos = 0.7071 sin = 0.7071  
 velocidade = 25.0000 velX = 17.6777 velY = 17.6777  
 tempo total = 6.2225 incremento temporal = 0.1556 entre colunas

	lin	nInters	y1	y2
+---+---+---+---+---+---+---+---+	25	0	-1.0000	-1.0000
	24	0	-1.0000	-1.0000
	23	0	-1.0000	-1.0000
	22	0	-1.0000	-1.0000
	21	0	-1.0000	-1.0000
+---+---+---+---+---+---+---+---+	20	0	-1.0000	-1.0000
	19	0	-1.0000	-1.0000
	18	0	-1.0000	-1.0000
	17	0	-1.0000	-1.0000
	16	0	-1.0000	-1.0000
+---+---ooooo---+---+---+---+---+	15	7	29.6100	30.0731
oo     oo	14	4	28.0304	28.9291
oo     o         A	13	3	25.5796	26.9139
o       o         A	12	2	24.0275	24.4400
o         o       A	11	2	22.2576	22.7251
+--o---+---+---+---+---+---+---+	10	2	20.2699	20.7924
o           o	9	2	18.0644	18.6419
o           o	8	2	15.6411	16.2736
E             o	7	2	13.0000	13.6875
	6	0	-1.0000	-1.0000
+---+---+---+---+---+---+o---+---+	5	1	10.8836	-1.0000
o	4	1	7.8619	-1.0000
	3	0	-1.0000	-1.0000
o	2	1	4.6224	-1.0000
o	1	1	1.1651	-1.0000
+---+---+---+---+---+---ooooo-----	0	11	0.0000	0.0000

Voce nao acertou o alvo nessa jogada.

====> Faca o lancamento 2 / 5: 0.7 0.7



Grafico do lancamento com deslocamento/orientacao = [ 0.7000, 0.7000]

Condicoes iniciais desse lancamento:

ang (rad) = 1.0996 cos = 0.4540 sin = 0.8910  
 velocidade = 35.0000 velX = 15.8897 velY = 31.1852  
 tempo total = 6.9227 incremento temporal = 0.1731 entre colunas

	lin	nInters	y1	y2
+---+---o+---+---+---+---+---+---+---+---+	25	2	50.6569	50.7822
o             o	24	2	47.5511	47.6878
	23	0	-1.0000	-1.0000
o             o	22	2	44.1757	44.3238
	21	0	-1.0000	-1.0000
+---+o---+---+---+---+---+---+o---+---+	20	2	40.5308	40.6902
	19	0	-1.0000	-1.0000
o               o	18	2	36.6162	36.7871
	17	0	-1.0000	-1.0000
o                 o	16	2	32.4321	32.6144
+---+---+---+---+---+---+---+---+---+	15	0	-1.0000	-1.0000
o                 o	14	2	27.9785	28.1721
A	13	0	-1.0000	-1.0000
o                 o A	12	2	23.2552	23.4602
A	11	0	-1.0000	-1.0000
+---+---+---+---+---+---+---+---+---+	10	0	-1.0000	-1.0000
o                 o	9	2	18.2624	18.4788
	8	0	-1.0000	-1.0000
E                 o	7	2	13.0000	13.2278
	6	0	-1.0000	-1.0000
+---+---+---+---+---+---+---+---+---+	5	0	-1.0000	-1.0000
	4	0	-1.0000	-1.0000
	3	0	-1.0000	-1.0000
	2	0	-1.0000	-1.0000
	1	0	-1.0000	-1.0000
+---+---+---+---+---+---+---+---+---+	0	0	-1.0000	-1.0000

Voce nao acertou o alvo nessa jogada.

====> Faca o lancamento 3 / 5: 0.7 0.65

Grafico do lancamento com deslocamento/orientacao = [ 0.7000, 0.6500]

Condicoes iniciais desse lancamento:

ang (rad) = 1.0210 cos = 0.5225 sin = 0.8526  
 velocidade = 35.0000 velX = 18.2874 velY = 29.8424  
 tempo total = 6.0151 incremento temporal = 0.1504 entre colunas

	lin	nInters	y1	y2
+---+---+o---+---+---+---+---+o---+	25	2	50.0508	50.2754
o           o	24	2	47.7001	47.9452
o           o	23	2	45.1459	45.4114
	22	0	-1.0000	-1.0000
o             o	21	2	42.3882	42.6741
+---+o---+---+---+---+---+---+o---+	20	2	39.4270	39.7333
	19	0	-1.0000	-1.0000
o             o	18	2	36.2623	36.5890
o	17	1	33.2412	-1.0000
o	16	1	32.8940	-1.0000
+---o---+---+---+---+---+---+---o---	15	2	29.3223	29.6898
	14	0	-1.0000	-1.0000
o               A	13	1	25.5470	-1.0000
A	12	0	-1.0000	-1.0000
o               A	11	1	21.5682	-1.0000
+---+---+---+---+---+---+---+---+	10	0	-1.0000	-1.0000
o	9	1	17.3858	-1.0000
	8	0	-1.0000	-1.0000
E	7	1	13.0000	-1.0000
	6	0	-1.0000	-1.0000
+---+---+---+---+---+---+---+---+	5	0	-1.0000	-1.0000
	4	0	-1.0000	-1.0000
	3	0	-1.0000	-1.0000
	2	0	-1.0000	-1.0000
	1	0	-1.0000	-1.0000
+---+---+---+---+---+---+---+---+	0	0	-1.0000	-1.0000

Voce nao acertou o alvo nessa jogada.

====> Faca o lancamento 4 / 5: 0.7 0.67

Grafico do lancamento com deslocamento/orientacao = [ 0.7000, 0.6700]

Condicoes iniciais desse lancamento:

ang (rad) = 1.0524 cos = 0.4955 sin = 0.8686  
 velocidade = 35.0000 velX = 17.3410 velY = 30.4021  
 tempo total = 6.3433 incremento temporal = 0.1586 entre colunas

	lin	nInters	y1	y2
+---+---o---+---+---+---+---+---+	25	1	49.8958	-1.0000
o           o	24	2	47.2247	48.8608
o	23	1	46.0997	-1.0000
o           o	22	2	44.3273	43.1123
o	21	1	41.2036	-1.0000
+---+---+---+---+---+---+---+o---+	20	1	39.8986	-1.0000
o	19	1	37.8535	-1.0000
o	18	1	36.4585	-1.0000
o	17	1	34.2771	-1.0000
o	16	1	32.7921	-1.0000
+---o---+---+---+---+---+---+---+	15	1	30.4744	-1.0000
o	14	1	28.8994	-1.0000
o               A	13	1	26.4453	-1.0000
o	12	1	24.7803	-1.0000
o               A	11	1	22.1899	-1.0000
+---+---+---+---+---+---+---+---+	10	0	-1.0000	-1.0000
o	9	1	17.7081	-1.0000
	8	0	-1.0000	-1.0000
E	7	1	13.0000	-1.0000
	6	0	-1.0000	-1.0000
+---+---+---+---+---+---+---+---+	5	0	-1.0000	-1.0000
	4	0	-1.0000	-1.0000
	3	0	-1.0000	-1.0000
	2	0	-1.0000	-1.0000
	1	0	-1.0000	-1.0000
+---+---+---+---+---+---+---+---+	0	0	-1.0000	-1.0000

Parabens! Voce acertou o alvo nessa jogada.

O jogo terminou.