EP2 - CCMI - 2007 - v. 1.0
Paredes 2: A missão

Paulo J. S. Silva

18 de outubro de 2007

1 Descrição

Vamos incrementar a implementação do primeiro EP para levar em conta várias paredes menores dentro da sala e várias partículas. Mais uma vez o programa deve simular a trajetória das partículas até que uma delas saia da sala ou sejam percorridos 200 segundos. Nesse EP o intervalo máximo de simulação será sempre de 0.05 segundo. Note que se uma partícula sai da sala no meio desse intervalo você deve descobrir o momento "exato" em que ela saiu.

A sala continua igual, mas o formato do arquivo de entrada mudou para poder descrever outras paredes e mais de uma partícula.

O programa deve ler um arquivo de entrada chamado
entrada.txt que possui na primeira linha a palavra "paredes" e em seguinda uma seqüência de linhas no formato:

```
<v ou h> <posicao> <inicio> <fim>
```

O significado de cada um dos itens acima é:

`v ou h:` Um caracter que indica se a parede é vertical (v) ou horizontal (h).

`posicao:` Posição da dimensão constante da parede (se a parede é vertical dá a sua posição x e vice-versa).

`inicio` **e** `fim:` Posição de início e fim da parede na direção que varia. Vale que $0.0 \leq inicio \leq fim \leq 1.0$.

A descrição das paredes acaba com uma linha com a palavra `particulas`. A partir deste ponto cada uma das novas linhas está no formato

```
<posição x> <posição y> <velocidade x> <velocidade y>
```

que descreve a posição inicial da partícula e sua velocidade.

Um exemplo de arquivo completo é dado a seguir:

```
paredes
v 0.25 0.25 0.75
v 0.75 0.25 0.75
h 0.1 0.25 0.75
h 0.9 0.25 0.75
particulas
0.5 0.5 0.57 0.023
0.1 0.1 0.17 0.19
0.9 0.9 0.23 0.05
0.9 0.9 0.05 0.137
```

2 Exemplos de execução

2.1 Uma partícula sai da sala

Arquivo de entrada dado acima.

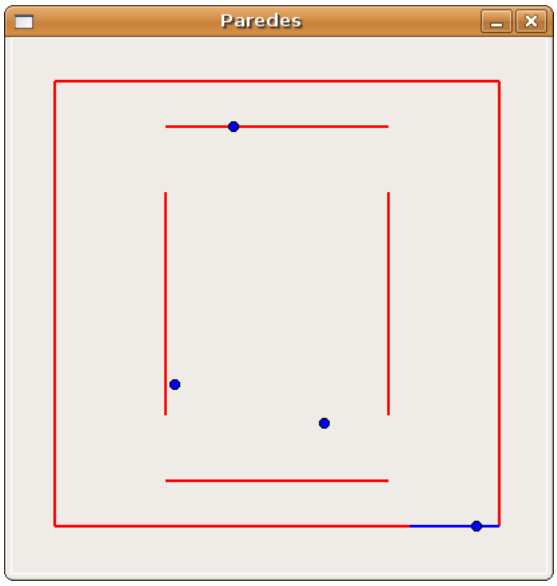

Figura 1: Imagem ao final de uma simulação em que a partícula escapa sala.

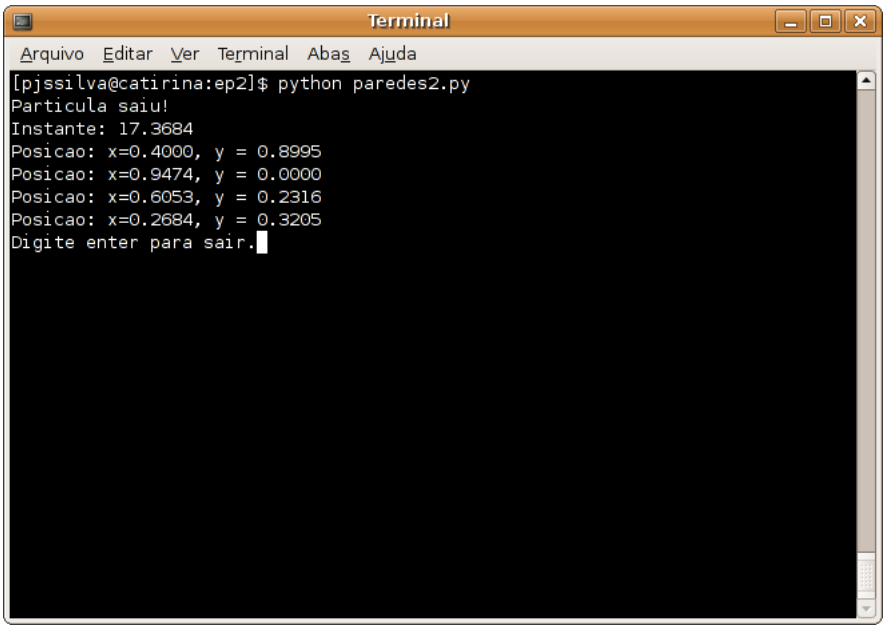

Figura 2: Mensagens no terminal quando a partícula deixa a sala.

2.2 Partícula fica da sala

Arquivo de entrada:

```
paredes
v 0.25 0.25 0.75
v 0.75 0.25 0.75
h 0.1 0.25 0.75
h 0.9 0.25 0.75
h 0.05 0.8 1.0
v 0.8 0.02 0.05
particulas
0.5 0.5 0.57 0.12
0.1 0.1 0.17 0.2
0.9 0.9 0.23 0.5
0.9 0.9 0.05 0.137
```

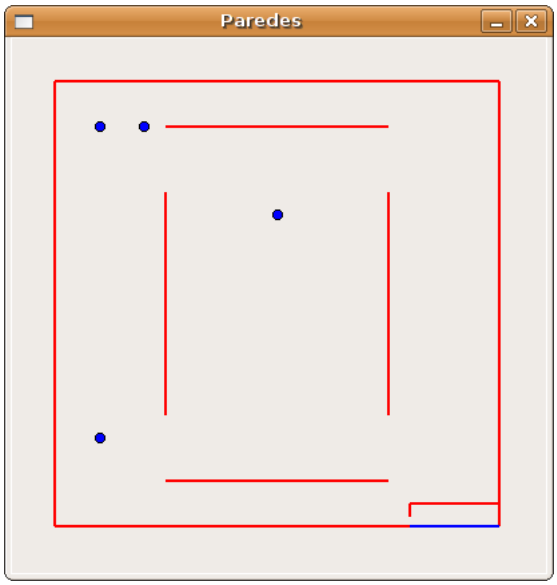

Figura 3: Imagem ao final de uma imagem em que a partícula fica na sala.

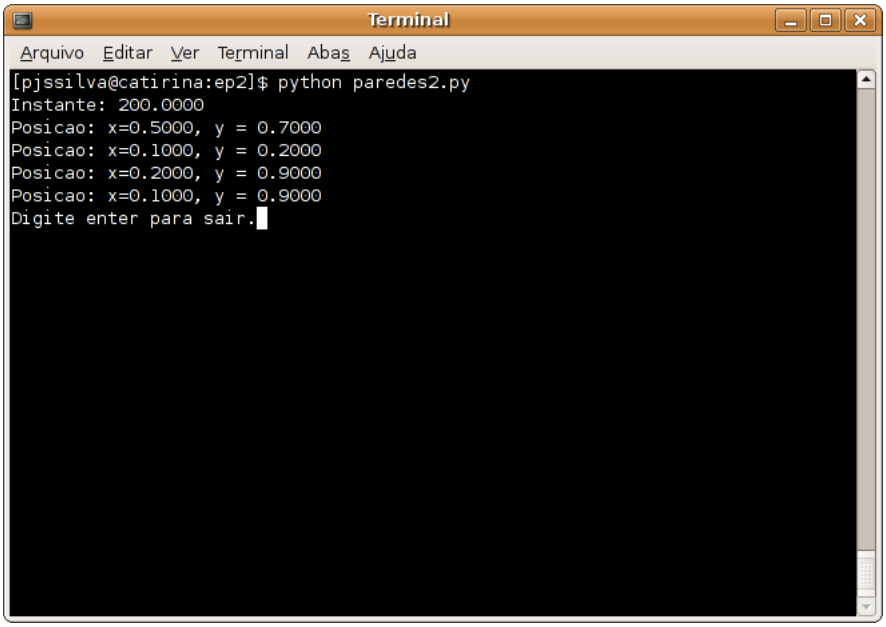

Figura 4: Mensagens no terminal quando a partícula fica na sala.

3 Detalhes de número em ponto flutuante

Vocês deve tomar cuidado com um detalhe importante de implementações que usam números reais (de ponto flutuante). Isso parece mais importante agora com várias bolas e várias paredes.

Como já comentei em sala, as contas e a representação dos números nesse sistema não é exata, incorrendo sempre em pequenos erros. Nesse em caso, não faz muito sentido comparar dois números que são resultados de contas para verem se são iguais. Assim, código que envolvem números em ponto flutuante geralmente comparam igualdades usando faixas, ou seja, define-se um $epsilon > 0$ e para verificar se $x = y$ usa-se um código do tipo

```
if math.fabs(x - y) < epsilon:
    print 'iguais'
else:
    print 'diferentes'
```

Ainda, para evitar problemas de precisão em situações extremas você pode assumir que duas paredes paralelas estão a pelo menos 0.01 metro de distância e que todas as velocidades são menores ou iguais a 100 m/s.

4 Considerações finais

Você deve organizar bem o seu programa, inclusive usando (várias) funções. Não é obrigatório o uso de classes, mas eu creio que isso irá facilitar bem o programa. Procure desenvolver o código de forma sistemática. É claro que dá para usar, parcialmente o código do EP passado, mas não tenha medo de jogar algo fora e começar de novo.

Além disso, se você sentir necessidade, comente o seu código. Para novatos, como vocês, é melhor pecar pelo excesso de comentários do que pela falta destes. No *mínimo* comente cada uma das funções da forma comentada em classe.

Ainda, o seu arquivo deve conter um cabeçalho contendo o nome do arquivo, o nome do autor e seu número USP e uma breve descrição dizendo o que o programa faz.

Por fim, lembrem de usar o fórum de discussão na página da disciplina. Isso é fundamental. No fórum posso disponibilizar correções no enunciado ou esclarecer dúvidas e definir detalhes que passaram despercebidos.

Não custa lembrar que o EP é um trabalho *individual*. Você pode discutir com os colegas o programa mas nunca trocar trechos de código. Sugiro nem mesmo olhar código do colega, apenas conversar. Caso encontremos trabalhos com trechos iguais os dois alunos receberão 0 no EP e a média final de EPs ainda receberá um desconto extra de 10%.