MAC 115 – Introdução à Computação para Ciências Exatas e Engenharia

FÍSICA – PRIMEIRO SEMESTRE DE 2007

Segundo Exercício-Programa — Data de entrega: até 18 de maio de 2007.

SIMULADOR DE CRESCIMENTO POPULACIONAL

1 Introdução

O objetivo deste EP é realizar algumas simulações de crescimento populacional [1], usando uma simples equação não-linear. Utilizaremos um modelo da Teoria do Caos que é utilizada para explicar uma série de fenômenos naturais ligados à Física, Biologia e outras ciências. Neste EP específico vamos estudar as mudanças populacionais de determinados animais ao longo do tempo, isto é, o número de animais existentes em diferentes gerações de uma certa população.

Tomemos um exeplo de crescimento na população de pombas em um zoológico. Suponhamos que o número máximo de pombas que poderiam manter-se no zoológico seja de 1000 e que existam atualmente 200. Vamos padronizar o tamanho da população existente de maneira a facilitar a comparação com outras populações. A padronização consiste em atribuir-se o valor 1 à população máxima. Assim a *população inicial* (X_0) é dada por $200/1000$ ou 0.2.

A *população seguinte* (X_{n+1}), contabilizada na próxima geração, depende de uma *taxa de crescimento* (a). Essa taxa de crescimento depende de diferentes fatores como a fertilidade das pombas, a quantidade de alimentos, doenças, etc.

P.F. Verhulst, em 1845, propôs um modelo para o crescimento populacional usando a seguinte equação:

$$X_{n+1} = a * X_n * (1 - X_n)$$

em que a é a taxa de crescimento, X_n a quantidade de elementos da população na geração n, e X_{n+1} a quantidade de elementos da população na geração $n + 1$.

Chamaremos de *iteração* populacional a aplicação reiterada da equação acima descrita. Se, no início, X_n for 0.2 e a taxa de crescimento for de 3, a geração seguinte da população será de $3 * 0.2 * (1 - 0.2) = 0.48$, ou o equivalente no nosso exemplo a 480 pombas.

Neste EP, vamos realizar diferentes simulações de crescimento populacional a fim de examinar como X_{n+1} varia em cada iteração. Deverão ser escolhidos valores crescentes de taxa de crescimento 'a' em cada simulação. O i-ésimo simulador terá uma taxa de crescimento de $i/2$. Dessa forma, a simulação 1 terá uma taxa de crescimento de 0.5, a simulação 2 terá uma taxa de 1, e assim sucessivamente.

Mostramos abaixo o resultado desejável da execução de simulações válidas para uma população inicial de 0.2 e 1000 gerações.

Simulador	Taxa	População final	Média	Variância
1	0.5	1.2773909359785701E-302	3.51479656708514E-4	4.799720494280577E-5
2	1.0	9.896821088562584E-4	0.005196871673622111	1.7180500968484776E-4
3	1.5	0.33333333333333337	0.33297156687119517	3.1657856942141297E-5
4	2.0	0.49999999999999994	0.49944721361110017	1.2623747180803526E-4
5	2.5	0.6	0.5994005994006106	1.9944091871673583E-4
6	3.0	0.6739969172672533	0.6656424548062693	5.288533963830337E-4
7	3.5	0.8269407065914387	0.6459911561525289	0.04403179953118893
8	4.0	0.13822325807715316	0.4992911610938014	0.1251076587334914

A coluna `Taxa` representa a taxa de crescimento usada na simulação enquanto `População final` representa a última geração (após 1000 iterações). `Média` e `Variância` representam, respectivamente, o valor da média e variância

[1] Por simplicidade, no contexto do EP, o decrescimento populacional será considerado como um crescimento negativo.

da população considerando todas as gerações, inclusive a população inicial. A população média e a variância de n gerações são calculadas como:

$$\text{média} \qquad \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\text{variância} \qquad \frac{\sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2/n}{n}$$

É interessante notar como a variância cresce bastante depois do sétimo simulador, indicando uma grande variação no tamanho da população para taxas maiores que 3.0. Vale salientar que muitos estudos posteriores aos de Verhulst constataram que a simplicidade da sua equação era enganosa. Em resumo, a conduta da população nas gerações recai em um dos casos seguintes dependendo do valor da taxa de crescimento (a):

- Extinção: A população morre se $a < 1$.
- Ponto fixo: A população tende a um valor fixo se $1 < a < 3$.
- Periódica: A população flutua entre 2 o mais valores discretos para $3 < a < 3.57$.
- Caótica: A população flutua com nenhum ordem aparente para valores de $a > 3.57$.

2 Implementação

A implementação do EP será feita em quatro partes: (1) escrita de uma classe população; (2) criação de uma classe de simulação populacional; (3) criação de uma classe de teste de simulações; e (4) criação de uma classe de gerenciamento de simulações. Cada classe deve conter pelo menos os atributos e métodos abaixo (outros atributos e métodos podem ser criados caso você julgue necessário):

1. Classe: `População`
   - `double numeroDeElementos`: Atributo que indica o número de elementos da população na geração atual.
   - `double taxaDeCrescimento`: Atributo que indica a taxa de crescimento da população.
   - `void inicializar(double populaçãoInicial, double taxaDeCrescimento)`: Inicializa os valores da população inicial conjuntamente com a taxa de crescimento.
   - `double geracaoSeguinte()`: Calcula o número de elementos da geração seguinte usando a equação de Verhulst.
2. Classe: `SimuladorPopulacional`
   - `População populaçãoEstudada`: Atributo que armazena a população associada ao simulador.
   - `void simular(double populaçãoInicial, double taxaDeCrescimento, int iterações)`: Inicia a simulação usando como população inicial `populaçãoInicial`, taxa de crescimento de `taxaDeCrescimento`, e número de gerações a serem simuladas (`iterações`).
   - `double populaçãoMédia()`: Devolve o valor médio de elementos na população ao longo de toda a simulação considerando todas as gerações, inclusive a população inicial.
   - `double variânciaDaPopulação()`: Devolve a variância da população considerando todas as gerações, inclusive a população inicial.

3. Classe: `TestaSimuladorPopulacional`

   - `void testePontualPopulaçãoFinal(double populaçãoInicial, double taxaDeCrescimento, int numIterações, double respostaEsperada)`: Realiza um teste de simulação depois de `numIterações` gerações usando como população inicial `populaçãoInicial` e taxa de crescimento de `taxaDeCrescimento`.

     Um teste será válido se o valor absoluto das diferenças entre a população final obtida (número de elementos na última geração) e a `respostaEsperada`, for menor que 10^{-10}. Você pode usar os valores da tabela do exemplo acima para testar seus programas.

4. Classe: `GerenciadorDeSimulações`

   - `void executar(double tamanhoPopulação, int iterações)`: Executa as simulações usando `tamanhoPopulação` como tamanho da população inicial em um número de gerações igual a `iterações`. A taxa de crescimento usada no simulador i será de $i/2$.

     Como resultado da execução, deve ser mostrada uma tabela de resultados das simulações válidas, como mostrado no exemplo anterior. Caso você queira, pode usar algum programa de visualização de gráficos para gerar gráficos da evolução do tamanho da população (por exemplo o Octave (`http://www.gnu.org/software/octave`) ou o Scilab (`http://www.scilab.org`, ambos programas de código aberto)).

Importante: Não esqueça de verificar o fórum do PACA periodicamente. Eventuais dicas e correções serão publicados nesse fórum.

3 Observações importantes

Sobre a elaboração:

- Este EP pode ser elaborado por equipes de dois alunos, desde que sejam respeitadas as seguintes regras (chamamos isso de *programação pareada*).
  - Os alunos devem trabalhar sempre juntos, a idéia é que deve existir uma cooperação.
  - Mesmo a digitação do EP deve ser feita em grupo, enquanto um digita, o outro fica acompanhando o trabalho.
  - Caso em um grupo exista um aluno com maior facilidade, este deve explicar e discutir as decisões tomadas. E o seu par deve participar e se esforçar para entender o desenvolvimento do programa.
- Recomendamos fortemente que o exercício seja desenvolvido da forma descrita na observação acima, ou seja, com programação pareada, mas também pode ser feito individualmente. Não são permitidas equipes de mais de 2 alunos.
- É absolutamente proibida a entrega do EP em equipes de 2 sem que seja usada programação pareada. Os professores não tem como fiscalizar isso mas contam com a ética e honestidade dos alunos para garantir que isso não aconteça.

Sobre a avaliação:

- No caso de exercícios feitos em dupla, a mesma nota da correção será atribuída aos dois alunos do grupo.
- **Não serão toleradas cópias**. Exercícios copiados (com ou sem eventuais disfarces) receberão nota zero (inclusive o original).
- **Exercícios atrasados não serão aceitos**.
- Exercícios com erros de sintaxe (ou seja, erros de compilação) receberão nota zero.

- É muito importante que seu programa use **nomes tão claros quanto possível** para seus métodos e variáveis, tenha **comentários** elucidativos e esteja **bem indentado**, ou seja, digitado de maneira a ressaltar a estrutura de subordinação dos comandos do programa (conforme visto em aula). A qualidade do seu trabalho sob esse ponto de vista influenciará sua nota.
- As informações impressas pelo seu programa na tela devem aparecer da forma mais clara possível. Este aspecto também será levado em consideração no cálculo da sua nota.
- Uma regra básica é a seguinte: do ponto de vista do monitor responsável pela correção dos trabalhos, quanto mais convenientemente apresentado estiver o seu programa e quanto mais fácil for para entendê-lo, melhor será a disposição do monitor para dar-lhe uma nota generosa.

Sobre a entrega:

- O prazo de entrega é o dia **18/5/2007**.
- No início dos arquivos, acrescente um cabeçalho bem informativo, como o seguinte:

```
/*****************************************************************/
/**   MAC 115 - Introdução à Computação                         **/
/**   Física-USP - Primeiro Semestre de 2007                    **/
/**   <turma> - <nome do professor>                             **/
/**                                                             **/
/**   Segundo EP -- Simulador de Crescimento Populacional       **/
/**                                                             **/
/**   <nome do(a) aluno(a)>               <número USP>          **/
/**   <nome do(a) aluno(a)>               <número USP>          **/
/**                                                             **/
/**   <data de entrega>                                         **/
/*****************************************************************/
```

  Não é obrigatório que o cabeçalho seja idêntico a esse, apenas que contenha pelo menos as mesmas informações.

- Para a entrega utilize o paca.ime.usp.br. Crie um arquivo compactado (no formato .zip, .tgz ou .rar) contendo os arquivos java e submeta ao paca este arquivo compactado. Você pode entregar várias versões de um mesmo EP até o prazo, mas somente a última será armazenada pelo sistema. Encerrado o prazo, o sistema não aceitará mais os EP's. Os procedimentos de entrega para trabalhos individuais e em grupo diferem um pouco, consulte seu monitor em caso de dúvida.
- Guarde uma cópia do seu EP pelo menos até o fim do semestre.