MAC 110/115 – Introdução à Ciência da Computação – 1º/2008

Segundo Exercício-Programa: GERADOR DE POESIA PSEUDO-DADAÍSTA

Data de entrega: até 30 de maio de 2008 às 23h55, pelo sistema PACA.

Professores: Marcelo Finger, Marcelo Queiroz e Roberto Marcondes

1 Introdução

Neste exercício-programa, você irá criar um programa de computador que compõe automaticamente pérolas da literatura universal, como as que seguem:

Exemplo 1	Exemplo 2
O céu gosta da beterraba como o inverno pensa na lasanha. O garfo come a idéia porque o pássaro da diarréia ri. Passa pelo amor a água toda vez que a árvore sai do frango. Enquanto o mosquito a enchente lambe o calor entra no dicionário.	Oculta o asno a cerveja se o ferro na nebulosa rasteja. Como ri da chacrete a besta e toca o fanfarrão o planeta. Quando o coelho a fogueira assa o joelho pelo fraco passa. Usurpa a diarréia a fome se a viola o bagulho come. A ordem teme a idéia mesmo quando o micróbio o zoológico desafia.

Notem a riqueza vocabular e a multiplicidade de imagens e significados. Observem as inversões de frases e as rimas ricas no segundo exemplo. Quantas sutilezas! Que perspicácia!

2 Os segredos do ofício

Para a produção de tão plurissignificativas preciosidades, o computador vai precisar de algum material de onde partir. Parte deste material será fornecido pelo usuário, enquanto outra parte fará parte do próprio programa. Veremos todos os detalhes em detalhe.

Para simplificar nossa vida, restringiremos um pouco o universo lingüístico do nosso "poeta automático". Ele utilizará apenas substantivos no singular e verbos transitivos diretos ou indiretos, na 3a pessoa do singular, no modo indicativo. Assim estão excluídos pronomes pessoais, verbos intransitivos, verbos transitivos direto e indireto, verbos reflexivos, voz passiva, subjuntivo, etc., etc., etc.

Especificamente, o usuário deverá fornecer ao computador uma lista de substantivos no singular, juntamente com o respectivo artigo, como "a árvore", "o morcego" ou "a imensidão", e também uma lista de verbos, conjugados na 3a pessoa do singular, juntamente com uma preposição (no caso dos verbos transitivos indiretos), como "passa por", "pensa em" ou "machuca -" (o hífen simboliza a ausência de preposição, pois trata-se de um verbo transitivo direto).

O usuário também vai informar se deseja ou não utilizar rimas, e quantos versos o poema deverá ter. Seu programa poderá gerar poemas tão longos quanto Os Lusíadas e tão concentrados quanto um haikai:

A goiabada com o cão viaja

a nuvem ofusca a inveja.

A risada trafega pelo pêssego.

Nos exemplos de poemas fornecidos na primeira página, podemos observar as principais características das frases:

- elementos: um sujeito, um verbo e um objeto (direto ou indireto);
- inversões: sujeito+verbo+objeto ou sujeito+objeto+verbo ou verbo+objeto+sujeito;
- conjunções: como, e, enquanto, mesmo quando, porque, quando, se, toda vez que;
- rimas: no segundo exemplo, os finais dos versos pares "rimam" com o final do verso anterior;

Na seção "Implementação" são fornecidos mais detalhes sobre estas características e como o seu programa pode utilizar as palavras fornecidas pelo usuário na geração de poemas.

3 Implementação

Neste EP você deverá fazer um programa que peça ao usuário uma lista de substantivos e uma lista de verbos, e permita ao usuário gerar poemas pseudo-dadaístas com ou sem rima. Para isso você deverá criar uma classe Java chamada `GeradorDadaista.java`, que possui *obrigatoriamente* os seguintes métodos:

- `static void produzVersos(int númeroDeVersos, boolean rima)`: gera (na tela) um poema com o número de versos desejado, com ou sem rima, dependendo do valor da variável booleana `rima`.
- `public static void main (String [] args)`: programa principal, que lê as listas de palavras, pergunta se o poema deve ou não usar rimas e qual o número de versos desejados, e gera o poema.

Com a função `main` seu programa poderá ser executado através da instrução `java GeradorDadaista`, tanto na janela de interações do DrJava quanto em um interpretador de comandos (como o bash ou o prompt de Comando). Em princípio, todos os métodos e atributos utilizados pela função main devem possuir o qualificador `static`, e dessa forma pertencem à classe (não são particulares a cada objeto da classe).

Você pode organizar o seu programa da maneira que achar melhor, através da definição dos atributos da classe, de outros métodos e até de outras classes, se julgar necessário.

3.1 Leitura e armazenamento das palavras

O usuário deverá informar a quantidade de substantivos e em seguida digitar uma lista de substantivos com os artigos. Depois ele deve informar a quantidade de verbos e digitar os verbos com as preposições correspondentes (ou '-' se o verbo não pede preposição). Um exemplo de entrada possível é:

```
Quantos substantivos você deseja utilizar?
3
Digite um substantivo (com artigo) por linha:
o guarda-chuva
a cebola
a interrogação

Quantos verbos você deseja utilizar?
4
Digite um verbo (com preposição) por linha:
gosta de
engana -
viaja com
reza por
```

Para armazenar os substantivos você pode usar dois vetores de String's: um para os artigos ("o" ou "a") e outro para os substantivos propriamente ditos. O mesmo pode ser feito com os verbos e as preposições.

Para ler palavras acentuadas (que utilizam o padrão UTF-8) você pode usar a seguinte receita:

```
Scanner sc = new Scanner(System.in,"UTF-8");
String palavra = sc.next();
```

3.2 Construção das frases

A construção das frases envolve os seguintes aspectos, não necessariamente nesta ordem:

- a escolha das palavras;
- a declinação das preposições;
- a escolha da ordem dos termos na frase; e
- o tratamento das rimas

A escolha das palavras será feita *obrigatoriamente* de maneira aleatória e evitando repetições. Já utilizamos a classe Random no primeiro EP:

```
Random gerador = new Random();
gerador.nextInt(N) // gera um número inteiro entre 0 e N-1
```

Para saber se uma palavra já foi usada ou não, você pode manter um vetor de indicadores booleanos para cada classe de palavras (substantivos e verbos). Cada sorteio pode então ser repetido um certo número de vezes até que se encontre uma palavra que ainda não foi usada.

É importante observar que a não-repetição não é uma condição obrigatória, do contrário não seria possível gerar poemas longos a partir de um conjunto pequeno de palavras. Considere que cada sorteio pode ser repetido até um máximo de `M=10*(númeroDeSubstantivos+númeroDeVerbos)` vezes; se todas as palavras sorteadas já tiverem sido usadas, fica-se com a última palavra sorteada.

As frases dos poemas também podem começar com conjunções, tais como "como", "e", "enquanto", "mesmo quando", "porque", "quando", "se", "toda vez que", e outras mais que você quiser incluir. Estas conjunções deverão ser sorteadas aleatoriamente, e utilizadas pelo programa em 30% das frases do poema, sempre na posição inicial da frase.

Quando a frase possui um verbo transitivo indireto, devemos declinar a preposição correspondente de acordo com o artigo do objeto, através das regras habituais: a+a=à, a+o=ao, de+o=do, de+a=da, em+a=na, em+o=no, por+a=pela, por+o=pelo, enquanto outras preposições mantém-se separadas dos artigos (com a, com o, para a, para o).

A escolha da ordem dos elementos principais da frase (com exceção da conjunção, quando existir) deve ser feita aleatoriamente, com probabilidade igual para os 3 casos seguintes (você pode considerar outras inversões, se quiser):

sujeito + verbo + objeto

sujeito + objeto + verbo

verbo + objeto + sujeito

A rima na realidade é um conceito complicado, mas para descomplicar nossa vida vamos considerar que duas palavras rimam quando elas terminam com as mesmas duas letras (fácil, né?). Se o usuário pedir poemas com rimas, utilizaremos o esquema de rimas AABBCCDD... ou seja, ao final das frases pares a palavra deve rimar com a última palavra da frase imediatamente anterior. Em outras palavras, nas frases ímpares não nos preocuparemos com rima, mas guardaremos a última palavra para gerar a frase seguinte; nas frases pares, após

definir a ordem dos termos, vamos procurar uma palavra da classe correspondente ao terceiro termo (pode ser substantivo ou verbo) que rime com a palavra que guardamos da frase anterior.

Assim como a não-repetição, o obtenção de uma rima também será tratada com flexibilidade (afinal pode acontecer de não existir rima em alguns casos): você poderá sortear até `M` palavras tentando satisfazer as condições de não-repetição e rima. Se após M tentativas não for obtida nenhuma palavra nova com a rima certa, fica-se com a última palavra sorteada.

3.3 Formatação da saída

Para deixar os poemas visualmente mais bonitos, cuide dos seguintes detalhes:

- as frases ímpares devem começar com letra maiúscula.
- as frases pares devem terminar com ponto final; a última frase do poema também deve terminar com ponto final, mesmo se for uma frase ímpar.
- pule uma linha no final de cada frase, e deixe uma linha em branco a cada 4 frases.

4 Exemplo de execução

No PACA você encontrará um programa executável com uma implementação deste EP. Use e experimente.

Há também arquivos com substantivos e verbos que você pode usar como entrada, para evitar ficar digitando sempre as mesmas palavras. Selecione o texto em um editor de texto e copie na janela de execução do GeradorDadaista.

Você pode criar arquivos com outros substantivos e verbos, de temas particulares (por exemplo futebol, política, astrologia...)

5 Observações importantes

5.1 Sobre a elaboração:

Este EP poderá ser elaborado por equipes de dois alunos, desde que sejam respeitadas as seguintes regras:

- Os alunos devem trabalhar sempre juntos: a idéia é que deve existir uma cooperação;
- Caso em um grupo exista um aluno com maior facilidade, este deve explicar as decisões tomadas. E o seu par deve participar e se esforçar para entender o desenvolvimento do programa (chamamos isso de *programação pareada*, que é uma excelente prática que vocês devem se esforçar para adotar);
- Mesmo a digitação do EP deve ser feita em grupo: enquanto um digita, o outro acompanha o trabalho e participa das decisões.

Recomendamos fortemente que o exercício seja desenvolvido da forma descrita nesta observação. No entanto, se desejado, o exercício também poderá ser feito individualmente.

5.2 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 tenha comentários 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, melhor será a disposição dele para dar-lhe uma nota generosa.

5.3 Sobre a entrega:

- O prazo de entrega é o dia 30/5/2008, às 23h55. Não será possível o envio de arquivos após este horário.
- O nome do arquivo da sua classe deve ser `GeradorDadaista.java`.
- No início do arquivo, acrescente um cabeçalho bem informativo, como o seguinte:

```
/*****************************************************************/
/**   MAC 110/115 - Introdução à Computação                     **/
/**   IME-USP/IF-USP - Primeiro Semestre de 2008                **/
/**   <turma> - <nome do professor>                             **/
/**                                                             **/
/**   Segundo Exercício-Programa  --  Gerador de Poesia Dadaista  **/
/**   Arquivo: GeradorDadaista.java                             **/
/**                                                             **/
/**   <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.

- Junto com o programa deve ser entregue uma simulação de 2 poemas, de 12 linhas cada, um com rima e outro sem rima.
- As simulações devem vir ao final do arquivo, na forma de comentários. Indique claramente o final de uma simulação e o início da próxima.
- Não é necessário usar mais de uma classe para a realização do EP. No entanto, se você quiser usar mais de uma classe, todas deverão estar contidas no mesmo arquivo `GeradorDadaista.java`. Se quiser, você poderá implementar outros métodos auxiliares além dos métodos obrigatórios.
- Para a entrega, utilize o moodle em `paca.ime.usp.br`. Para enviar o programa, você precisa ter um cadastro de usuário e estar inscrito na página do curso.
- 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 a ajuda do moodle para obter mais detalhes.
- Guarde uma cópia eletrônica do seu EP pelo menos até o fim do semestre!

Bom Trabalho!