Instituto Tecnológico de Costa Rica

Departamento de Ingeniería en Computación

Curso: Lenguajes de Programación

Primer Tarea Programado:

Paradigma Funcional, Pollito Poeta

Estudiante:

Eduardo Chavarría Rey 2013012779

Profesor:

Kirstein Gätjens Soto.

Sede Central, ITCR

6 de abril del 2015

Índice

Análisis de Resultados

A- Utilidad concluida exitosamente.

B- Utilidad concluida con leves problemas.

C- Utilidad concluida con graves problemas.

D- Utilidad únicamente concluida en su diseño.

E- Utilidad no concluida.

Funcionalidad	Resultado
Generar Versos	A
Generar Párrafos	A
Generar Poemas	A
Generar Rimas	A
Imprimir Poema	A
Aleatoriedad de Estructura	A
Validación de versos	A

Diseño de la solución

La resolución del programa en general, consistió en la división de las funcionalidades de cada utilidad del proyecto tomándolas como funciones individuales llamadas en forma conjunta, con el objetivo de simplificar las soluciones, y al mismo tiempo reciclar el código lo más posible.

Lo primero que se hizo en fue la generación de versos bases de forma aleatoria (a los cuales se les hace su respectiva validación en relación a la cantidad de sílabas), y estos se usan como base para la generación de versos rimas. Con lo que finalmente en un algoritmo “molde”, se les da la forma o estructura solicitada por el profesor.

La estructura para la oración de cada verso se elige de forma aleatoria, donde la elección de cada palabra también se elige de forma aleatoria. Ambas funciones reciben de parámetro una lista, la cual se usa para filtrar la lista de palabras a elegir; cosa que es particularmente útil para la elección de palabras que rimen.

La base de la impresión, es un algoritmo llamado “imprimirVersos”, la cual se mapea sobre un párrafo en la función “imprimirParrafos”, la cual a su vez es también mapeada sobre la función “imprimirPoema”.

Finalmente vale la pena destacar que aunque la elección de palabras y estructuras se genera de forma aleatoria, se decidió optimizar el código mediante el uso de parámetros de listas filtradas.

La gramática BNF utilizada para la creación de las estructuras de los versos fue la siguiente:

1) <s>::= <sustantivoObligado> <verbo> <sustantivoOpcionalconPrep>
2) <s>::= <literal-preposicion> <sustantivoObligado> <verbo> <sustantivoOpcionalconPrep>
3) <s>::= <verbo> <sustantivoObligado>
4) <s>::= <literal-adverbioMente> <verbo> <sustantivoObligado>
5) <sustantivoOpcionalconPrep>
6) <verbo>::= <literal-verbo><adverbioMente>
7) <adverbioMente>::= <literal-adverbioMente>
8) <adverbioMente>::= épsilon
9) <sustantivoOpcional>::= <literal-preposicion> <sustantivoObligado>
10) <sustantivoOpcional>::= épsilon
11) <sustantivoObligado>::= <literal-articulo> <literal-sustantivo> <adjetivo>
12) <adjetivo>::= <literal-adjetivo>
13) <adjetivo>::= epsilon

Manual de Usuario

Para el correcto funcionamiento del programa se requiere:

- Tener instalado Chicken Scheme.
- Generar una copia del archivo “poeta”, en el path (C:\Program Files (x86)\Chicken Scheme\bin) o (C:\Program Files\Chicken Scheme\bin). Esto varía según el sistema operativo.
- Abrir el csi.exe de en el path descrito anteriormente.
- Ingresar (load “Pollito Poeta - Chavarria Eduardo.egg”).

Con todo lo anterior realizado, ya se podrá acceder a la aplicación y funciones de esta.

Entre las de mayor relevancia para el usuario se menciona:

- (Poeta paramInt): Genera un soneto, de 8 versos; compuesto de 3 párrafos (2 de 3 versos, 1 de 4 versos).
  Donde las rimas pueden ser asonantes o consonantes.
  E imprime la poesía como symbols.
- (createVerso): Genera un verso al azar.
- (crearRima paramVerso paramInt): Genera un verso que rime, con el verso ingresado como parámetro.
  Requiere de un número aleatorio de parámetro para definir el tipo de rima a realizar.
- (crearPoema paramInt): Genera un soneto, de 8 versos; compuesto de 3 párrafos (2 de 3 versos, 1 de 4 versos).
  Donde las rimas pueden ser asonantes o consonantes.
  Pero no imprime la poesía como symbols.

- (filtroRima paramLista paramRima): Genera una lista filtrada, a partir de una lista parámetro y una lista que contenga las letras de una rima (ya sea consonante o asonante).
  Por ejemplo: (filtroRima x ‘(a e)).
  Y el resultado será una lista con todas las palabras de la lista x, las cuales tengan rima asonante (a e).
- (filtro paramLista paramTipo): Genera una lista filtrada, a partir de una lista parámetro y un tipo de palabra (adjetivos, sustantivos, adverbios, etc).

Bibliografía

http://roble.pntic.mec.es/msanto1/lengua/proverso.htm

http://docs.racket-lang.org/reference/

http://wiki.call-cc.org/

http://www.rae.es/