Processamento de Linguagens e Compiladores (3º ano de Curso)

Processador de trasações da Via Verde

Relatório de Desenvolvimento

Carlos Silva (A62156) Paulo Sousa (A69318)

25 de Outubro de 2015

Resumo

Neste trabalho iremos usar o gerador de análise de texto FLEX para que através de regras e reconhecimento de padrões lexicais conseguir criar um filtro de texto capaz de receber um extrato mensal de um utente de Via Verde e exemplifica-lo calculando o valor total(em euros) gasto em portagens e parques, criando também um grafo que ligue nós de entrada e de saída (utilizando a linguagem Dotty e o sistema de constrção de grafos Dot). Pretendemos incentivar e puxar o leitor a conhecer mais sobre este rápido analizador lexical, valorizando a sua grande utilidade e a sua velocidade de processo.

Conteúdo

1 Introdução **2**

2 Análise e Especificação **3**
2.1 Descrição informal do problema 3
2.2 Especificação dos Requisitos 3
2.2.1 Dados 3
2.2.2 Pedidos 4

3 Concepção/desenho da Resolução **5**
3.1 Estruturas de Dados 5
3.2 Algoritmos 5

4 Codificação e Testes **6**
4.1 Alternativas, Decisões e Problemas de Implementação 6
4.2 Testes realizados e Resultados 6

5 Conclusão **7**

A Código do Programa **8**

Capítulo 1

Introdução

FLEX é uma ferramenta usada para gerar analizadores de texto(scanners), este programa lê arquivos de entrada especificados pelo utilizador, e através de expressões regulares e regras de código C filtra o ficheiro texto gerando um arquivo de origem C chamado "lex.yy.c"que pode ser compilado para produzir um executável. Assim este analizador percorre o texto procurando ocorrências de texto correspondentes às expressões ou regras, executando o código C correspondente. Vamos aplicar esta ferramenta FLEX na realização do nosso trabalho usando-a expecificamente para analizar um documento de texto, onde nele se encontra um qualquer estrato de um cliente do serviço de Via Verde. Temos primeiramente como objectivo o retorno do dinheiro total gasto em portagens e estacionamentos, fazendo uso, como referido em cima, de expressões regulares e regras de código C. De seguida apresentaremos o desenho de um grafo ligando os nós de entrada e saída do cliente, bem como os parques de estacionamento por ele visitado e o número de trajectos feitos. Para a realização desta segunda tarefa iremos recorrer à linguagem e sistema de construção de grafos Dot. Dot é uma linguagem simples que é usada para a criação de grafos. A saída é apresentada em desenho e em vários formatos possíveis (ps, jpeg, png, etc). De evidenciar que a realização da primeira tarefa irá ser apresentada em linguagem C e a segunda tarefa em linguagem C++, isto com a finalidade de obter os resultados pretendidos.

Estrutura do Relatório

O documento, de seguida, será estruturado em quatro Capítulo. No Capítulo 2 vamos analisar a descrição do problema e especificar os seus requisitos (dados propostos e pedidos apresentados). No Capítulo 3 iremos passar para a realização/resolução dos problemas propriamente ditos, apresentando e explicando as estruturas de dados e os algoritmos usados. No Capítulo 4 falaremos dos problemas de implementação, dos testes por nós realizados e dos resultados obtidos. Nestes dois últimos Capítulos temos como finalidade proporcionar ao leitor uma melhor compreenção de como chegamos à resolução final dos problemas apresentados. Por último no Capítulo 5 terminamos o relatório com uma síntese do que anteriomente foi dito, apresentando as devidas conclusões.

Capítulo 2

Análise e Especificação

2.1 Descrição informal do problema

Desenvolvimento de um Filtro de Texto usando Flex para ler um extrato mensal de um cliente de Via Verde no formato XML exemplificando-o primeiramente para calcular o total gasto em portagens e em parques e seguidamente desenhar um grafo do seu percurso.

2.2 Especificação dos Requisitos

Calcular o total gasto em portagens e o total gasto em parques, apresentando o valor em float com duas casas decimais. Desenvolvimento de um grafo que ligue os nós de entrada e saída entre si (no caso do parque cria um lacete), sem repetições de trajectos (se a ligação for repetida, registar o número de vezes como peso do ramo), recorrendo à linguagem Dot.

2.2.1 Dados

Extrato mensal da Via Verde fornecido no formato XML:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<EXTRACTO id="011114056/08/2015">
<MES_EMISSAO>Ago-2015</MES_EMISSAO>
<CLIENTE id="514714936">
<NIF>nnn nnn nnn</NIF>
<NOME>Nome Apelido</NOME>
<MORADA>RUA RRR</MORADA>
<CODIGO_POSTAL>CP...</CODIGO_POSTAL>
</CLIENTE>
<IDENTIFICADOR id="28876820811">
<MATRICULA>nn-ll-nn</MATRICULA>
<TRANSACCAO>
<DATA_ENTRADA>26-07-2015</DATA_ENTRADA>
<HORA_ENTRADA>11:33</HORA_ENTRADA>
<ENTRADA>Povoa N-S</ENTRADA>
<DATA_SAIDA>26-07-2015</DATA_SAIDA>
<HORA_SAIDA>11:42</HORA_SAIDA>
<SAIDA>Angeiras N-S</SAIDA>
<IMPORTANCIA>2,00</IMPORTANCIA>
<OPERADOR>I. de Portugal (N1)</OPERADOR>
<TIPO>Portagens</TIPO>
```

```
</TRANSACCAO>
<TRANSACCAO>
<DATA_ENTRADA>26-07-2015</DATA_ENTRADA>
<HORA_ENTRADA>11:45</HORA_ENTRADA>
<ENTRADA>Aeroporto</ENTRADA>
<DATA_SAIDA>26-07-2015</DATA_SAIDA>
<HORA_SAIDA>11:50</HORA_SAIDA>
<SAIDA>Ponte Pedra</SAIDA>
<IMPORTANCIA>0,65</IMPORTANCIA>
<OPERADOR>I. de Portugal (P3)</OPERADOR>
<TIPO>Portagens</TIPO>
</TRANSACCAO>
<TRANSACCAO>
<DATA_ENTRADA>06-08-2015</DATA_ENTRADA>
<HORA_ENTRADA>15:57</HORA_ENTRADA>
<ENTRADA></ENTRADA>
<DATA_SAIDA>06-08-2015</DATA_SAIDA>
<HORA_SAIDA>17:08</HORA_SAIDA>
<SAIDA>PQ A Sa Carn.I</SAIDA>
<IMPORTANCIA>3,75</IMPORTANCIA>
<OPERADOR>ANA - Aeroportos de Portugal. SA (AP)</OPERADOR>
<TIPO>Parques de estacionamento</TIPO>
</TRANSACCAO>
</IDENTIFICADOR>
</EXTRACTO>
```

2.2.2 Pedidos

Calcular o gasto total em portagens e o total gasto em parques, usando o analisador de texto Flex. Desenvolvimento de um grafo que ligue os nós de entrada e saída entre si com a ajuda da linguagem Dot.

Capítulo 3

Concepção/desenho da Resolução

3.1 Estruturas de Dados

3.2 Algoritmos

Capítulo 4

Codificação e Testes

4.1 Alternativas, Decisões e Problemas de Implementação

4.2 Testes realizados e Resultados

Mostram-se a seguir alguns testes feitos (valores introduzidos) e os respectivos resultados obtidos:

Capítulo 5

Conclusão

Síntese do Documento.
Estado final do projecto; Análise crítica dos resultados.
Trabalho futuro.

Apêndice A

Código do Programa

Lista-se a seguir o código Java [?] do programa Darius [?] que foi desenvolvido.

```
aqui deve aparecer o código do programa,
tal como está formato no ficheiro-fonte "darius.java"
```

Listing A.1: Exemplo de uma Listagem

```
1 ou então aparecer aqui neste sítio
2 como alternativa ao anterior.
```

É ainda possível importar diretamente o ficheiro: