MAC122 Princípios de Desenvolvimento de Algoritmos
EP no. 2

Prof. Dr. Paulo Miranda
Instituto de Matemática e Estatística (IME)
Universidade de São Paulo (USP)

O XML (*eXtensible Markup Language*) é um formato recomendado pela W3C para a criação de documentos com dados organizados de forma hierárquica, cujo propósito principal é facilitar o compartilhamento de informações através da internet.
Um documento em XML é um texto que apresenta marcações, chamadas tags (etiquetas), que são usadas para organizar o conteúdo do documento. Tags são identificadas como trechos do texto compreendidos entre um caracter de início `'<'` e outro caracter de fim `'>'`. Genericamente falando, existem dois tipos de tags - tags de abertura: `<comando>` e tags de fechamento: `</comando>`. A diferença entre elas é que na tag de fechamento existe uma barra `'/'`. Abaixo é mostrado um exemplo de documento em XML:

```
<?xml version="1.0"?>
<note>
    <to>Jane</to>
    <from>Tarzan</from>
    <heading>Reminder</heading>
    <body>Jane must stay with Tarzan!</body>
</note>
```

Você deve ter notado a partir do exemplo anterior que a declaração `<?xml ... ?>` da primeira linha não tem uma tag de fechamento correspondente. Isso não é um erro. Essa primeira declaração não tem por padrão nenhuma marca de fechamento. Ela apenas informa a versão do documento XML.

As tags de abertura podem possuir atributos na forma de pares `nome="valor"`. Exemplo:

```
<?xml version="1.0"?>
<note date="12/11/2007">
  <to>Jane</to>
  <from>Tarzan</from>
</note>
```

No exemplo acima `date="12/11/2007"` é um atributo da tag `note`.

Um documento em XML apresenta as seguintes regras de sintaxe:

1. Os documentos XML são sensíveis à letras maiúsculas e minúsculas.
   Ex: `<to>Jane </TO>` está errado.

2. Todos os elementos XML devem ter uma tag de fechamento correspondente devidamente aninhada.
   Exemplo de documento incorreto:
   ```
   <?xml version="1.0"?>
   <note date="12/11/2007">
     <to>Jane
     <from>Tarzan
   </note>
   ```

   Um segundo exemplo de documento incorreto:
   ```
   <?xml version="1.0"?>
   <note date="12/11/2007">
     <to>Jane</note>
     <from>Tarzan</from>
   </to>
   ```

3. Todos os documentos XML devem ter um único elemento raiz que contém todos os outros elementos. Por exemplo, o elemento raiz dos exemplos acima é a tag `<note>`.

Para um dado arquivo texto, contendo um documento no formato XML, faça um programa que verifica se as tags de abertura e fechamento estão balanceadas. O seu programa deve imprimir uma mensagem de erro, indicando a primeira ocorrência de uma violação das regras de 1 a 3.
Para a resolução do problema, utilize uma implementação de pilha com listas ligadas. O programa deve receber como entrada o nome do arquivo texto no formato XML (ex: “example.xml”) e deve produzir mensagens tais como indicadas nos exemplos de execução abaixo. Nos exemplos, as entradas do usuário correspondem aos textos em vermelho e as saídas do programa aos textos em azul.

Exemplo de documento simples que viola a segunda regra do arquivo “example01.xml”:

```
<?xml version="1.0"?>
<informacao>
  <pais>Chile</capital>
  <capital>Santiago</pais>
</informacao>
```

Entrada e saída esperada do programa:

```
example01.xml
Violacao da regra #2:
Tags incompativeis (abertura 'pais' na linha 3 e fechamento 'capital' na linha 3)
```

Exemplo de documento que viola a segunda regra do arquivo “example02.xml”:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note>
  <to>Tove</to>
  <from>Jani
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

Entrada e saída esperada do programa:

```
example02.xml
Violacao da regra #2:
Tags incompativeis (abertura 'from' na linha 4 e fechamento 'note' na linha 7)
```

Exemplo de documento mais complexo com tags balanceadas do arquivo “example03.xml”:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<receita nome="pao" tempo_de_preparo="5 minutos" tempo_de_cozimento="1 hora">
  <titulo>Pao simples</titulo>
  <ingredientes>
    <ingrediente quantidade="3" unidade="xicaras">Farinha</ingrediente>
    <ingrediente quantidade="7" unidade="gramas">Fermento</ingrediente>
    <ingrediente quantidade="1.5" unidade="xicaras" estado="morna">Agua</ingrediente>
    <ingrediente quantidade="1" unidade="colheres de cha">Sal</ingrediente>
  </ingredientes>
  <instrucoes>
    <passo>Misture todos os ingredientes, e dissolva bem.</passo>
    <passo>Cubra com um pano e deixe por uma hora em um local morno.</passo>
    <passo>Misture novamente, coloque numa bandeja e asse num forno.</passo>
  </instrucoes>
</receita>
```

Entrada e saída esperada do programa:

```
example03.xml
Tags balanceadas
```

Exemplo de documento que viola a primeira regra do arquivo “example04.xml”:

```
<?xml version="1.0"?>
<informacao>
  <pais>Chile</PaiS>
  <capital>Santiago</CapiTaL>
</informacao>
```

Entrada e saída esperada do programa:

```
example04.xml
Violacao da regra #1:
Tags incompativeis (abertura 'pais' na linha 3 e fechamento 'PaiS' na linha 3)
```

Exemplo de documento que viola a terceira regra do arquivo “example05.xml”:

```
<?xml version="1.0"?>
<informacao>
  <pais>Chile</pais>
  <capital>Santiago</capital>
</informacao>
<nome apelido="Fenomeno"> Ronaldo
</nome>
```

Entrada e saída esperada do programa:

```
example05.xml
Violacao da regra #3:
Raiz adicional ('nome' na linha 6)
```

Observações:

Note que a indentação (recuo) é geralmente usada em arquivos XML, pois torna o documento mais legível para humanos. No entanto, ela não é necessária em documentos XML. O seu programa não precisa verificar questões relativas a indentação do documento.

Assuma que os caracteres `'<'` e `'>'` sempre aparecem aos pares e nessa ordem (ou seja, para todo caracter `'<'` o próximo caracter de marcação é sempre um `'>'`).

O nome do elemento deve seguir imediatamente a marca de início `'<'`. Isso significa que não há espaços em branco entre o nome e `'<'`. Por exemplo, se você estiver criando um elemento chamado de `message`, ele deve ler: `<message>` e não `< message>`. O seu programa pode assumir que os elementos foram digitados corretamente seguindo essa convenção.

Por questões de simplicidade, assuma o padrão ASCII.

Algumas funções para a manipulação de arquivos como a função `getc`, `rewind`, `fseek`, e `ftell` podem ser úteis. Consulte a documentação dessas funções na internet ou em livros especializados.