Conceitos de Linguagens de Programação
MAC5754
Relatório das fases 1 e 2

Alantiel Freire Marins
Bárbara Tieko Agena
Rayssa Küllian Martins
Yuri David Santos

10 de Abril de 2014

Resumo

Nas fases 1 e 2, iremos apresentar o relatório do andamento do projeto de uma linguagem de programação, aqui se encontram a descrição do domínio e os elementos essenciais da nossa linguagem.

A seguir iremos descrever a futura linguagem, seu nome será MarkIME, o motivo deste nome se explica pelo foco da linguagem, pois a linguagem, como será visto, é uma facilitadora para a escrita de LaTeX, que é uma linguagem de markup, linguagem muito utilizada no IME para apresentação de trabalhos e artigos.

Conteúdo

1 Fase 1

1.1 Descrição do Domínio

Escrever LaTeXpode ter uma estrutura simplificada para o desenvolvimento quando se comparado a linguagens de programação, porém, para usuários com pouco conhecimento neste âmbito, esta tarefa pode exigir maior esforço.

Dentro do cenário de pesquisadordes que desejam escrever artigos em LaTeX, nem sempre existe o conhecimento em desenvolvimento de sistemas, isto é, muitos alunos tem foco teórico no seu campo de atuação, mas não em habilidades específicas de LaTeX.

Dessa forma, visamos facilitar o processo de escrita de artigos, relatórios e apresentações de slides em LaTeXem geral, evitando tarefas repetitivas, aumentando o poder de reutilização de código e criando um meio mais direto de escrevê-los.

1.2 Proposta da Linguagem de Programação

Escrever uma linguagem que facilite a manipulação de artigos, relatórios e apresentações em LateX, adicionando flexibilidade ao processo de edição por meio de uma sintaxe mais simples e flexível, facilitando tarefas comuns ao contexto e embutindo novos recursos.

Características da linguagem:

- Facilitar tarefas como escape de caracteres especiais, acentuação, etc.
- Simplificar a construção de tabelas e inclusão de imagens, legendas, etc.
- Comando de repetições e condições inexistentes em LaTeX pura para fazer tratamentos úteis como condições sobre elementos do texto, em formatos iterativos em LaTeX.

1.3 Elementos essenciais à linguagem

1. **Operadores**:
   - **Operações sobre texto**: Concatenação, uppsercase, etc.
   - **Operações atitméticas**: Soma, multiplicação, etc.
   - **Operações lógicas**: E / OU
2. **Declarações**:
   - **Constantes**: a linguagem suportará constantes numéricas e de string
   - **Variáveis**: a linguagem suportará variáveis numéricas e de string
3. **Funções**:
   - **Funções matemáticas**: para realizar operações sobre os números
   - **Funções para manipulação de texto**: para realizar operações sobre as strings, fazer tratamentos de case, busca, etc.

2 Fase 2

2.1 Tipos de Dados

A linguagem terá apenas tipo de dados primitivos, separamos em 2 grupos:

2.1.1 Numéricos

- **Inteiros**: 1, 2, -3, 42, etc...
- **Ponto flutuante**: 0.0, -0.5

2.1.2 Não numéricos

- **String**: , "Meu texto", etc...
- **Booleano**: true, false

2.2 Expressões

A linguagem terá 3 formas de expressões para os valores das variáveis:

2.2.1 Literais

- "Meu texto"equivale ao valor string "Meu texto".
- 1 equivale ao número inteiro 1.

2.2.2 Aplicação de funções

- upper("minhaString") equivale a string "MINHASTRING"
- concat("minhaString", "minhaNovaString") equivale a string "minhaStringminhaNovaString"

2.2.3 Valores associados a identificadores

- var i = 1 equivale a definir o inteiro 1 em i
- var s = "texto"equivale a definir a string "texto"em s

2.3 Avaliação de expressões

2.3.1 Ordem de avaliação

Mesma ordem da avaliação matemática usual, isto é, da esquerda para a direita, com ordem de precedência: (), *, /, +, -

2.4 Comandos

A linguagem terá recursos extras para tratamento de string e números, esses recursos são providos na maioria dos casos por funções, tais como.

2.4.1 Comandos de atribuição

```
var x = y
```

2.4.2 Blocos

```
head [...]
var x
#document{
    var y = 2
    $$z = x + y$$
}
```

2.4.3 Condicionais

```
#head [...]
var x = 0
#document{
    var y = 2
    $$if(y < x){ y } else { x }$$  //mostra o menor
}
```

2.4.4 Comandos de iteração

```
#head [...]
var x = 5
#document{
    $$for(var y = 0; y < x; y++){ y }$$  //itera 5 vezes
}
```

2.5 Sistema de tipos

Nossa linguagem dependerá apenas do tipo dos argumentos para definir a abstração a ser utilizada, isto é, não será dependente de contexto.

2.5.1 Verificação de tipos

A linguagem validará os tipos das variáveis ao aplicar alguma abstração, dessa forma, não permitindo ambiguidades em tempo de compilação.

2.6 Esboço de um artigo escrito em MarkIME

```
#head['Test article', 'Footnote',
      'http://test.com', 'Author1; Author2;Author3']

      var string = "string-test"
      var count = 1

#abstract{
   Nome do artigo   //just if want overhide the default
   ==============

   Article description, bla bla, bla
}

#document{
  Name of article    //just if want overhide the default
  ==============

  Section 1
  ---------

  Paragraph are separeted a blank line,
  a just break doesn't do anything.

  Text in *italic*,
  **bold**, 'monospaced'.

  Um [link](http://teste.com).

  Variables and methods
  $$string | concat(concat(string, string), "last")$$
  $count = $ $$count+1$$  //print count = 1
  $count = $ $$count+1$$  //print count = 1
  different of
  $$count+1$$             //print 2
  $$count = count+1$$     //print 2
  $$count = count+1$$     //print 3

  Unordered list:
    * list a
    * list g
    * list c

  Ordered list:
    1. list 1
    2. list 2
    3. list 3
}
```