Insper

Elementos de Sistema - Projeto J - VM Translator

Rafael Corsi - rafael.corsi@insper.edu.br

Maio - 2018

> **!** `Prazo original : Terça Feira - 22/5/2018`

Descrição

Nesse projeto iremos criar o programa *VM translator* que é responsável por traduzir os códigos escrito em linguagem VM de pilha para a linguagem assembly.

Instruções

As instruções técnicas referente ao projeto estão no documento H-Ferramental.pdf

Módulos a serem implementados

O projeto no total possui 5 módulos, sendo que o módulo VMTranslator.java, VMtranslate.java e Error.java já estão foram entregues implementados.

> Os módulos estão listados de maneira Top - Down

- VMTranslator
  - **Arquivo** : VMTranslator.java
  - **Descrição** : Main do projeto. Recebe como parâmetro o nome do arquivo .vm (ou diretório) e o nome do arquivo binário assembly (.nasm) a ser escrito, passa essas informações para a classe VMtranslate.
  - **Dependências** : VMtranslate.java
- VMtranslate

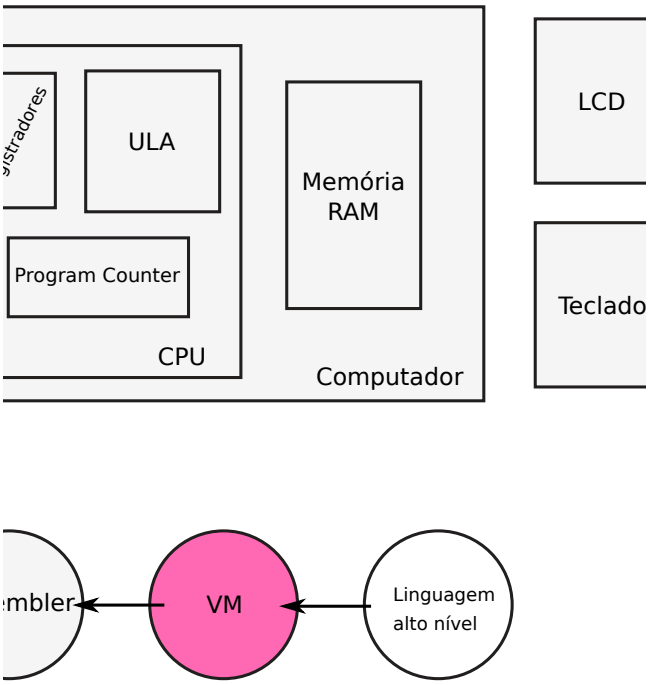

Figure 1: Assembly

  - **Arquivo** : VMtranslate.java
  - **Descrição** : Classe responsável por criar o código assembly, ela que **efetivamente** faz a varredura do arquivo .vm de entrada e escreve o arquivo .nasm de saída, gerando a tradução vm -> nasm.
  - **Dependências** : Code.java, Parser.java
- Code
  - **Arquivo** : Code.java
  - **Descrição** : Traduz comandos da linguagem vm para os comandos em nasm que a executam.
  - **Dependências** : none
- Parser
  - **Arquivo** : Parser.java
  - **Descrição** : Encapsula o código de leitura. Carrega as instruções na linguagem vm, analisa, e oferece acesso as partes da instrução (campos e símbolos). Além disso, remove todos os espaços em branco e comentários.
  - **Dependências** : none

Sugestão de implementação (partes)

Sugerimos que o vmtranslator seja implementado em duas partes, a primeira implementa somente o *Parser.java* e operações aritméticas e push/pop. A segunda parte faz a implementação de funções, goto e chamada de funções.

Parte 1

Implementar :

- Parser.java
- Code.writeArithmetic()
- Code.writePushPop()

Parte 2

Implementar :

- Code.writeGoto()
- Code.writeIf()
- Code.writeCall()
- Code.writeReturn()
- Code.writeFunction()

Rubricas para avaliação de projetos

Cada integrante do grupo irá receber duas notas: uma referente ao desenvolvimento total do projeto (Projeto) e outra referente a sua participação individual no grupo (que depende do seu papel).

Projeto

Conceito	
I	- Menos da metade dos módulos funcionando
D	- Teste unitário ou Teste integração não passa
C	- Criado a partir de estrutura de código disponibilizada - Todos os testes unitários passam no teste - Os testes de integração passam nos testes - Travis configurado corretamente
B	- Implementando modo verbose (-v) que possibilita analisar o vm e suas etapas
A	- Código bem documentado e explicado

- – Entregue atrasado = Conceito C