Universidade de Évora
Departamento de Informática
Linguagens de Programação
Ano letivo 2019 - 2020

Leituras de Programas TISC

Alunos:
Luís Ressonha - 35003
Rúben Teimas - 39868

Docente:
Teresa Gonçalves

2 de Maio de 2020

Indice

1 Introdução

Pretende-se que ao longo da UC de Linguagens de Programação seja desenvolvida uma máquina *TISC*.

Para um desenvolvimento gradual da máquina desta arquitetura o trabalho foi dividido em partes, sendo que nesta 1ª parte é apenas necessário ler as instruções de um programa *TISC* através do *standard input*(stdin) e carregá-las para a memória de instruções da máquina.

Foi-nos dada total liberdade quanto à escolha da linguagem de programação ainda que tenha sido aconselhada a linguagem *Java*, que acabámos por utilizar.

Para além de Java utilizámos também um analisador léxical (*JLex*) e um analisador sintático (*Cup*) compativeis com a linguagem.

Para compilar o trabalho deve ser executado o comando "*make*"na pasta **ficheiros** que irá compilar o programa. De seguida, na mesma pasta deve ser executado o seguinte comando: *make run < ../exemplos/programa* , em que *programa* é o ficheiro de input que se pretende passar à máquina.

2 Desenvolvimento

2.1 Estrutura da Máquina

A máquina de Arquitetura *TISC* possuí 3 zonas distintas de memória: a memória de instruções, a pilha de avaliação e a memória de execução. Possui também 2 registos: o *Program Counter* e o *Environment Pointer*.

A estrutura base de execução bastante simples: recebe um input do *stdin*, passa esse input pelo *Lexer* (*JLex*), o resultado dessa passagem é enviado para o *Parser* (*Cup*) que ao identificar a instrução coloca-a na memória de instruções. Este processo pode ser observado na *Figura 1*.

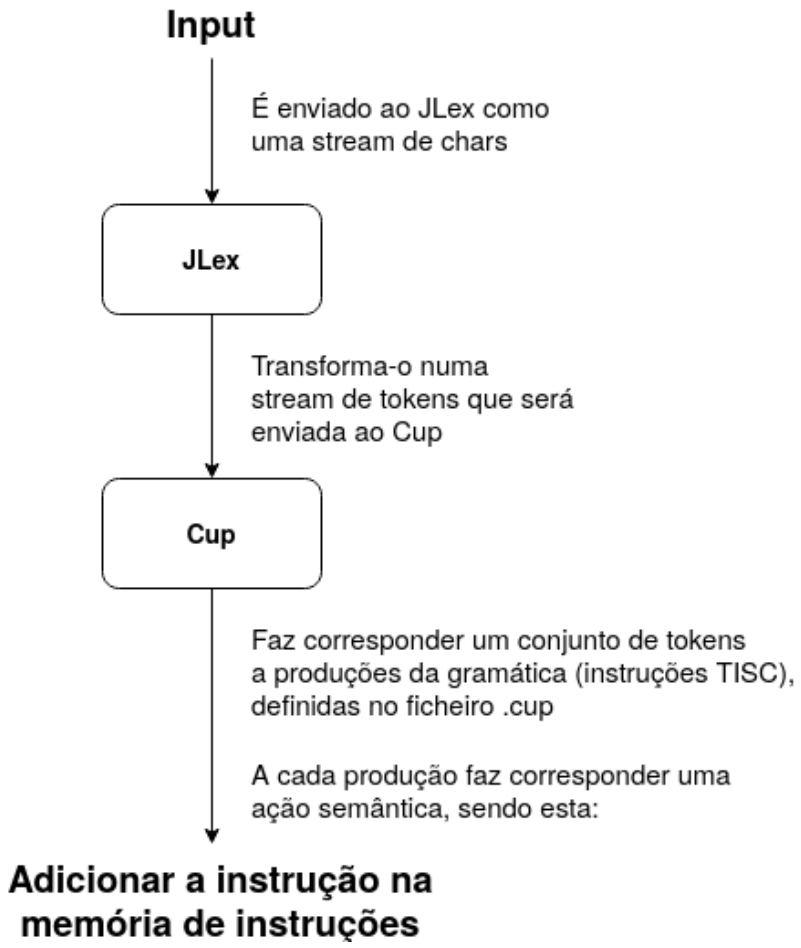

Figura 1: Estrutura base da execução Máquina

Para esta 1ª fase do trabalho, a única ação realizada sobre as instruções em memória será o *print* das mesmas.

Posteriormente, noutra fase do trabalho, as instruções serão executadas garantindo assim a execução de programas *TISC*.

2.2 Implementação da Máquina

Para esta fase do trabalho implementámos apenas a **memória de instruções** e a **pilha de avaliação**, tendo utilizado um *ArrayList* e uma *Stack* respetivamente.

Sendo a memória de instruções preenchida com instruções, optámos por criar uma classe abstrata **Instrução** que define o tipo mais abrangente de uma instrução tendo declarado unicamente os métodos abstratos *executa(TISC maquina)* e o *toString()*, dado que todas as instruções terão estes métodos.

Dividimos as instruções por grupos, adicionando mais uma camada de abstração, sendo esta divisão a indicada no enunciado:

- Instruções Aritemétic as: classe *Aritmetica*;
- Instruções de Acesso a Variáveis: classe *AcessVar*;
- Instruções de Acesso a Argumentos: classe *AcessArg*;
- Instruções para chamada de funções: classe *ChamaFunc*;
- Instruções de Salto: classe *Salto*;
- Instruções de Saída: classe *Output*;

Todas as classes acima extendem a classe mais abstrata *Instrução*.

A classe de cada instrução extende a respetiva classe abstrata e implementa o método *toString()*. O método *executa(TISC maquina)* foi por enquanto deixado incompleto propositadamente, pois o funcionamento de cada instrução pertence à fase seguinte do trabalho. Ainda assim, alguns construtores e métodos possuiem pedaços de código comentado que fizemos a pensar na próxima fase.

Também a pensar na próxima fase decidmos criar na máquina uma *Hashtable* que tem como chave o nome da *Label* e como valor a posição em que esta se encontra, para que quando exista um jump para uma *label* saibamos qual o valor que o *program counter* deve passar a ter.

3 Conclusão

Acreditamos ter concluído o objetivo desta fase do trabalho e tendo conseguido ler e carregar as instruções para a memória de instruções.

Ainda que com bastantes aspetos em abertos pensamos também ter chegado a uma estrutura da máquina que se adequa à fase seguinte do trabalho, de modo a que não tenhamos de proceder a qualquer alteração.