UNIVERSIDADE DE ÉVORA
ESCOLA DE CIÊNCIAS E TECNOLOGIA
DEPARTAMENTO DE INFORMÁTICA

Linguagens de Programação

Carregamento de programas CIMS

por

Eduardo Romão nº35477,

João Dias nº35476

13 de Abril de 2018

Tabela de Contéudos

Introdução

Com a realização deste trabalho, pretendemos implementar uma memória na máquina CIMS, que ao ler um ficheiro carrega-o. Recorremos à utilização da linguagem de programação Java, visto que a professora fornece as ferramentas adequadas para a realização do presente trabalho.

Desenvolvimento

Como referido anteriormente, a máquina possui três zonas de memória. Assim sendo, foi criado um *ArrayList* para cada parte da memória.

```
public class CIMS {
        private ArrayList<Object> memoria;
    private ArrayList<Object> avaliacao;
    private ArrayList<Object> execucao;
```

Listing 2.1: Memória

2.1 Implementação

Nesta fase, verificamos que cada tipo de instrução tem os mesmos argumentos à excepção das instruções do tipo saída e chamada de funções. Para cada tipo de instrução foram criadas classes em que o construtor recebe os argumentos dessa mesma instrução.

Após a criação das classes, ao ficheiro CIMS.cup foram adicionados os objetos através de um método criado no CIMS.java, onde se encontra a nossa memória.

```
        public void add_Inst(Object obj)
    {
        memoria.add(obj);
    }

    public void add_Label(Label label)
    {
        memoria.add(label);
    }
```

Listing 2.2: Métodos de Adicionar à Memória da máquina

```
instrucao ::= CHAMA INTEIRO:d IDENTIFICADOR:p
                                {: maquina.add_Inst(new I_Chamada_Funcoes("Chama"
    ,

            Integer.parseInt(d.toString()),

            p.toString())); :}
          |   LOCAIS INTEIRO:a INTEIRO:v
```

```
                                {: maquina.add_Inst(new I_Chamada_Funcoes("Locais
",

          Integer.parseInt(a.toString()),

          Integer.parseInt(v.toString())); :}
        |   REGRESSA
                                {: maquina.add_Inst(new I_Chamada_Funcoes("
Regressa")); :}
        |   COLOCA_ARG INTEIRO:n
```

Listing 2.3: Exemplos no Ficheiro Cup

Instruções Aritméticas: A classe I_Aritméticas, nesta fase do trabalho, apenas contém uma string com o nome da operação aritmética (soma, sub, mul, div, mod e exp).

Instruções de Salto: A classe correspondente é a classe I_Salto, que tal como a anterior contém uma string com o nome, e contém outra string que corresponde à etiqueta.

Instruções para Manipulação de Inteiros: I_Manipulação_Inteiros, a classe contém uma string com o nome e um int para o inteiro.

Instruções de Acesso a Variáveis e a argumentos: Ambas as classes têm dois inteiros e uma string, I_Acesso_Variaveis e I_Acesso_Argumentos.

Instruções para Chamada de Funções: A classe I_Chamada_Funcoes contém três construtores. Cada construtor recebe os argumentos de cada instrução.

Instruções de Saída: I_Saida, esta classe, tal como a anterior, possui mais que um construtor.

2.2 Comandos

Os comandos para executar o trabalho são os seguintes:

compilar: make

limpar: make clean

correr ficheiro: make run caminho/nome_do_ficheiro.cims

Conclusão

No início do desenvolvimento do presente trabalho deparámo-nos com algumas dificuldades em entender o Makefile. Sem saber como utilizar o Makefile fornecido optámos então, por um desenvolvimento "*à bruta*", onde descrevemos as classes indicadas no trabalho. Inicialmente, uma vez que não sabíamos como utilizar o CUP, consultávamos o ficheiro colocando em memória cada instrução.

Na aproximação do prazo de entrega do referido trabalho, em diálogo com os nossos colegas, apercebemo-nos de que a principal causa de não conseguirmos utilizar o Make fornecido baseava-se no facto de o java_cup não funcionar estando em jar. Como tal, descomprimimos o jar tornando o Make utilizavél o que permitiu alterar o trabalho para a versão atual, concluindo assim a primeira etapa da sua execução.