Compiladores, 2022/2023

Trabalho prático, parte 1
– Expressões aritméticas para a linguagem Mar –

1 Introdução

Pretende-se que faça um compilador de uma linguagem muito simples, que apenas permite o cálculo de expressões aritméticas. Um programa nesta linguagem consiste numa ou mais instruções. A linguagem só permite um tipo de instrução: a palavra reservada 'print', seguido de uma expressão aritmética, seguido de um ponto e vírgula. Eis um exemplo de um programa válido:

```
print (1 + 2) * (3 + 4);
print -2 + 3.14;
```

A expressão aritmética pode ter os 4 operadores usuais: +, -, * e /, bem como os habituasi parentesis curvos.

Os operandos podem ser literais inteiros ou reais, sem sinal. Note que o símbolo menos (-) tanto pode ser usado quer como operador binário, quer unário.

O seu compilador deve geral código para uma máquina virtual de stack, onde todas as operações aritméticas são realizadas no stack de execução. Internamente, quer os inteiros quer os reais são manipulados pela máquina virtual como se fossem um double.

O compilador deverá ler o input a partir de um ficheiro que contém o código fonte a compilar, e deverá ter a extensão '.mar'. O compilador deverá gerar os byte-codes correspondentes num ficheiro com o mesmo nome mas com extensão 'marbc'. Por exemplo, dado o input 'teste.mar', deverá ser gerado (caso não haja erros de compilação) um ficheiro de output com o nome 'teste.marbc'.

Para além do compilador, deverá também programar uma máquina virtual que receba como input um ficheiro '.marbc' e execute o código nele contido.

A título de exemplo, se o ficheiro de input '.mar' tivesse apenas as duas instruções apresentadas anteriormente, a execução dos bytecodes contidos no '.marbc' correspondente deveria produzir o seguinte output:

```
21.0
1.14
```

2 Instruções da máquina virtual

A máquina virtual tem apenas 7 instruções:

```
DCONST
ADD
SUB
MUL
DIV
UMINUS
PRINT
HALT
```

Destas 7 instruções, apenas DCONST tem um argumento que consiste num double (representado por 8 bytes, tal como em Java).

Numa máquina de stack, as expressões artiméticas são realizadas no stack: os operandos são empilhados no stack. Para se realiar uma operação, faz-se pop do(s) operando(o), efectua-se a operação, e empilha-se o resultado no stack.

A isntrução DCONST empilha o seu argumento no stack de execução.

A instrução PRINT deve desempilhar o valor que está no topo do stack e escrevê-lo para o ecrã.

A instrução HALT termina o programa.

2.1 Exemplo

O seguinte input:

```
print (1 + 2) * (3 + 4);
print -2 + 3.14;
```

Deveria gerar o seguinte output (assemby).

```
 0: DCONST 1.0
 1: DCONST 2.0
 2: ADD
 3: DCONST 3.0
 4: DCONST 4.0
 5: ADD
 6: MULT
 7: PRINT
 8: DCONST 2.0
 9: UMINUS
10: DCONST 3.14
11: ADD
12: PRINT
13: HALT
```

3 Condições de realização

O projeto deve ser realizado em grupo, de acordo com as inscrições em grupo do laboratório. Deverão submeter o vosso código num ficheiro ZIP por via eletrónica, através da tutoria, até às 23:59 do dia 29/03/2023.

Apenas é necessário que 1 dos elementos do grupo submeta o trabalho.