Compiladores

Linguagem *QLang*

Departamento de Electrónica, Telecomunicações e Informática
Universidade de Aveiro

Abril de 2024

Objectivos

O objectivo geral deste trabalho é o desenvolvimento de uma linguagem de programação compilada – i.e. que crie programas equivalentes ao programa a compilar numa linguagem de programação genérica (Java, Python, ...) – para construção de questionários de avaliação (especialmente dirigidos para o ensino de programação).

Esta linguagem permite a definição de vários tipos de perguntas, e depois executar um questionário (apenas para um aluno).

Por exemplo, podemos definir as seguintes perguntas:

```
hole Question.q1 is   # Question.q1 is the full question id
  println "A atribuicao de valor em PIL usa o operador " ans->":=" "."
end; # automatic grading

open OpenQuestion is
  println "Defina a estrutura de dados lista ligada."
end; # manual grading

code-open Question.Code1 is
   uses code from "even-odd.pil" end;
   print "Implemente um programa que, pedindo um numero inteiro ao ";
   print "utilizador com o texto 'Number: ', escreva na consola ";
   println "se este e' par (even) ou impar (odd).";
end; # automatic grading
```

Depois podemos executar essas perguntas:

```
q: question; # question type variable
q := new Question.q1;
execute q;
execute new OpenQuestion;
q := new Question.Code1;
execute q;
```

Por fim, podemos exportar todo o questionário (com todas as respostas e as avaliações automáticas):

```
export result to "result.txt";
```

A descoberta da sintaxe desta linguagem deve ser feita recorrendo os programas de exemplo.

Sendo esta linguagem dirigida especialmente para o ensino da programação, faz parte do projecto a definição de uma linguagem secundária interpretada, denominada *Procedural Imperative Language* (PIL), que implementa uma linguagem genérica (simplificada).

Como exemplo de um programa simples nesta linguagem:

```
— Procedural Imperative Language — conditional

n := integer(read "Number: "); — type conversion: type(expression)
write "Number ",n, " is ";
if n % 2 = 0 then — = is the comparison operator (as in math)
   writeln "even"
else
   writeln "odd"
end;
```

Características da solução

Apresentam-se a seguir um conjunto de características que a solução desenvolvida pode ou deve contemplar. Essas características estão classificadas a 3 níveis:

- mínima – característica que a solução tem obrigatoriamente que implementar;
- desejável – característica não obrigatória, mas fortemente desejável que seja implementada pela solução (apenas considerada se as mínimas forem cumpridas);
- avançada – característica adicional apenas considerada para avaliação se as obrigatórias e as desejáveis tiverem sido contempladas na solução.

Características mínimas

Os exemplos `p1.q`, `p2.q`, `p1.pil`, `p2.pil` e `p3.pil` indicam algum código fonte que tem de ser aceite (e devidamente compilado e interpretado) pelas linguagens a desenvolver.

A linguagem deve implementar:

- Instrução para definir perguntas dos tipos: **hole**, **open**, **code−open**, **code−hole**.

  Nesta definição cada pergunta tem de ter uma identificação única (no programa), qualquer uma delas do tipo **question**.

  A identificação de cada pergunta pode ser hierarquizada numa especie de definição de subtipos. Por exemplo, uma pergunta com a identificação A.B.C, pertence aos grupos A e A.B. Assim, uma referência ao grupo A selecciona (aleatoriamente) uma das perguntas desse grupo, e assim sucessivamente.

- Instrução para definir código (em linha ou importado de um ficheiro externo).

  Tal como acontece com as perguntas, nesta definição o identificador tem de ser único (incluindo os identificadores das perguntas), e o código é do tipo **code**.

- Os tipo de de dados inteiro, texto, e fracção. Este último corresponde a uma fracção inteira.

- Aceitar expressões aritméticas standard para os tipos de dados numéricos. Deve também aceitar a operação de concatenação de texto (sem nenhum operador, isto é, dois textos seguidos correspondem à sua concatenação).

- Instrução de escrita no *standard output* (com e sem mudança de linha no fim).

- Instrução de leitura de texto a partir do *standard input*.

- Operadores de conversão entre tipos de dados (por exemplo, **text**(10) para converter para texto; ou **integer**("10") para converter para inteiro).

- Instrução para executar código ou perguntas.

- Instrução para exportar resultados do questionário.

- Verificação semântica do sistema de tipos.

A linguagem interpretada deve aceitar os programas `pil` acima referidos. Deve também incluir:

- A definição de expressões booleanas (predicados) contendo, pelo menos relações de ordem e operadores booleanos (conjunção, disjunção, etc.).

- Instrução condicional.

- Instrução de iteração (loop).

Características desejáveis

Os exemplos `p3.q` e `p4.pil` indicam algum código fonte que se enquadra nas características desejáveis.

- Permitir a definição de expressões booleanas (predicados) contendo, pelo menos relações de ordem e operadores booleanos (conjunção, disjunção, etc.).

- Incluir a instrução condicional (operando sobre expressões booleanas).

- Incluir a instrução iterativa (operando sobre expressões booleanas).

- Perguntas dos tipos: **multi−choice** e **code−output**.
- A definição de buracos em código definido em linha (ver `p3.q`).
- Definição de pergunta composta (que pode incluir a execução de uma ou mais perguntas).

Características avançadas

- Implementar funções e variáveis locais às mesmas.
- Implementar uma tabela de símbolos que resolva o problema dos contextos de declaração.
- ...