Perfil de EL - Engenharia de Linguagens
(1º ano do MEI)

Trabalho Prático 2 (TP2) de
EG – Engenharia Gramatical

Ano Letivo 2022/23

1 Analisador de Código Fonte

Como estudou no 1º Trabalho Prático, existem várias *ferramentas avançadas de análise de código* (ou seja, de *programas-fonte em Linguagens de Programação de alto-nível*) com vista a ajudar em tarefas diferentes: embelezar textualmente a escrita do programa; detetar situações que infringem as boas-práticas de codificação na linguagem em causa ou que podem ser vulneráveis durante a execução; sugerir *melhores* formas de codificar sem alterar o significado do programa-fonte; avaliar a performance do programa estática ou dinamicamente; etc.

Agora, neste 2º Trabalho Prático, pretende-se que desenvolva um Analisador de Código para uma evolução da sua Linguagem de Programação Imperativa (`LPI`), definida na resolução do TPC que lhe foi proposto no início desta UC (na forma de um *trabalho de grupo*).

Recorda-se que essa Linguagem `LPI` deve permitir: declarar variáveis e manipular valores dos tipos *Inteiro, Booleano, Array, Tuplo, String, Lista*; escrever instruções tais como *Atribuição, Leitura, Escrita, Seleção (SE, CASO), Repetição (ENQ-FAZER, REPETIR-ATE, PARA-interv-FAZER*; escrever expressões para definir valores com base em operadores associados aos tipos tais como *operadores aritméticos, operadores lógicos, operadores relacionais, operadores de indexação de arrays, seleção de elementos de tuplos e de listas, inserção em listas, teste de pertença em listas.*

Concretamente, deve escrever em `Python`—usando o `Parser` e os `Visitors` do módulo para geração de processadores de linguagens `Lark.Interpreter`—uma ferramenta que: analise programas escritos na sua linguagem `LPI`; e gere em HTML(exemplo na secção 2 (de maneira a que seja possível visualizar da melhor forma as informações pedidas abaixo) um relatório com os resultados dessa análise, nomeadamente:

1. Lista de todas as variáveis do programa indicando os casos de: *redeclaração*[1] ou *não-declaração*[2]; variáveis usadas[3] mas *não inicializadas*[4]; variáveis declaradas e *nunca mencionadas.*
2. Total de variáveis declaradas versus os Tipos de dados estruturados usados.
3. Total de instruções que formam o corpo do programa, indicando o número de instruções de cada tipo (*atribuições, leitura e escrita, condicionais* e *cíclicas*).
4. Total de situações em que estruturas de controlo surgem aninhadas em outras estruturas de controlo do mesmo ou de tipos diferentes.

Além dessas métricas, adicione informação acerca das situações em que surjam *estruturas de controle aninhadas.* Quando essas situações envolvam `SE` aninhados, indique se há a possibilidade desses aninhamentos poderem ser substituídos por um só `SE`.

Como é habitual, o TP será entregue na forma de um relatório desenvolvido em LaTeX, utilizando para isso o template de relatório que se encontra no Material de Apoio à disciplina da Blackboard.

[1] Uma variável declarada 2 vezes.
[2] Uma variável mencionada no código sem ter sido inicialmente declarada.
[3] Uma variável que está a ser mencionada como operando de uma operação ou como parametro na chamada de uma função.
[4] Uma variável à qual nunca foi atribuído um valor.

2 Html de exemplo

```
<!DOCTYPE html>
<html>
<style>
    .error {
        position: relative;
        display: inline-block;
        border-bottom: 1px dotted black;
        color: red;
    }

    .code {
        position: relative;
        display: inline-block;
    }

    .error .errortext {
        visibility: hidden;
        width: 200px;
        background-color: #555;
        color: #fff;
        text-align: center;
        border-radius: 6px;
        padding: 5px 0;
        position: absolute;
        z-index: 1;
        bottom: 125%;
        left: 50%;
        margin-left: -40px;
        opacity: 0;
        transition: opacity 0.3s;
    }

    .error .errortext::after {
        content: "";
        position: absolute;
        top: 100%;
        left: 20%;
        margin-left: -5px;
        border-width: 5px;
        border-style: solid;
        border-color: #555 transparent transparent transparent;
    }

    .error:hover .errortext {
        visibility: visible;
        opacity: 1;
    }
</style>

<body>

    <h2>Análise de código</h2>
```

```
    <pre><code>
<p class="code">
    int z;
</p>
<p class="code">
    if(<div class="error">cond<span class="errortext">Variável não inicializada</span></div>){
</p>
<p class="code">
            if(<div class="error">y<span class="errortext">Variável não inicializada</span></div>) {
</p>
<p class="code">
                    if(<div class="error">z<span class="errortext">Variável não inicializada</span></div>) {
</p>
<p class="code">
                            z=1;k=3}
</p>
<p class="code">
            if(k) {
</p>
<p class="code">
                    k=2; z=9; x=5;}
</p>
<p class="code">
            }
</p>
<p class="code">
    }
</p>
<p class="code">
    int x;
</p>
<p class="code">
    x=3;
</p>

</code></pre>
</body>

</html>
```

Que produz o seguinte output :

Análise de código

```
int z;

if(cond){


        if(y) {


                if(z) {


                        z=1;k=3}

        if(k) {

                k=2; z=9; x=5;}

        }

}

int x;

x=3;
```

Figura 1: Exemplo de output.