PL

2025-05-26

Exame de época normal: pl2025-normal

UC: Processamento de Linguagens (LEI)

26 de Maio de 2025, 16h30-19h30, CP2: salas 1.01, 1.03, 1.05 e 1.07

Engenharia Informática (3º ano)

Questão 1: Expressões Regulares (especificação) (3 val.)

Escreva uma expressão regular que "apanhe":

- **a)** (1 val.) Dado um programa Python, apanhe a primeira linha de cada função python;
- **b)** (1 val.) Dado um programa Python, apanhe as duas primeiras linhas de cada função python;
- **c)** (1 val.) Um parágrafo (sequência de linhas não vazias até uma linha em branco ou fim de ficheiro).

Questão 2: Expressões Regulares (módulo re) (3 val.)

Considere o seguinte programa Python, no qual uma expressão regular é usada para decifrar uma mensagem codificada:

```
import re

codigo_secreto =
"""YQDRGLIWFSPGNGARTNOH0IILGZRA9290CL5AGE61H907PS770TSYXCVGT6AMAMJF6KQGRI7
Q3YPDOIOOTJUN1T7EXMVOSCQZWE99ZRGLPX0E"""

mensagem = re.findall(r"[A-GP-Z](\w)\d", codigo_secreto)
print(''.join(mensagem))
```

- **a)** (1,5 val.) Apresente o resultado produzido pelo programa.
- **b)** (1,5 val.) Apresente uma possível forma de codificar a mensagem "PL25".

Questão 3: Gramáticas (5v)

Escreva, em BNF-puro, uma Gramática Independente de Contexto (GIC) para definir formalmente uma linguagem específica para descrever a programação diária do conjunto não-vazio de canais televisivos de acordo com as seguintes definições (3,5 val.):

- A programação diária é uma lista de programa (nome, descrição, hora de início e de fim e tipo);
- O tipo de um programa pode ser entretenimento, cinema, noticiário, concurso, etc;
- Cada canal tem um nome e é identificado por uma sigla.

Como a ideia é construir um parser *TopDown* a GIC a escrever não deve ter conflitos LL(1). Prove-o no fim de a especificar (1,5 val.).

Questão 4: Compilador (6 val.)

A gramática independente de contexto, G, abaixo escrita em *BNF*, define uma linguagem de domínio específico para descrição de uma enciclopédia.

O Símbolo Inicial é **Encic**, os Símbolos Terminais são escritos só em maiúsculas se forem palavras-reservadas, só em minúsculas se forem terminais-variáveis (como: **letra** que representa uma só letra maiúscula entre parêntesis retos, **pal** que representa uma só palavra, **int** que representa um número inteiro, **str** que representa um texto entre aspas, e **data** que representa uma data) ou entre apóstrofes (sinais-de-pontuação, ou literais); os restantes (sempre começados por maiúsculas) serão os Símbolos Não-Terminais.

```
p1:     Encic     -->  FrontMat Caps
p2:     FontMat -->  Titulo Autor Versao data
p3:     Titulo  -->  str
p4:     Autor   -->  str
p5:     Versao  -->  VERSION ':' int
p6:     Caps    -->  Capitulo
p7:                |  Caps Capitulo
p8:     Capitulo-->  letra Entrs
p9:     Entrs   -->  Entrada '.'
p10:               |  Entrs Entrada '.'
p11:    Entrada -->  pal ':' Classe ';' Sin Sins
p12:    Sin     -->  pal
p13:    Sins    -->  €
p14:               | ',' Sin Sins
p15-17:  Classe  -->  'Sub' | 'Ver' | 'Adj'
```

Neste contexto e após analisar G, responda às alíneas seguintes:

- **a)** (1 val.) Escreva uma *frase válida* da linguagem gerada por **G**, apresentando a respetiva *árvore de derivação*.
- **b)** (2 val.) Escreva, em linguagen algorítmica, a função de um parser RD (recursivo-descendente) apropriada para reconhecer o símbolo não-terminal **Sins**.
- **c)** (1 val.) Escreva em Python usando o módulo **ply.lex** um analisador léxico para reconhecer as frases da linguagem definida por **G**.
- **d)** (2 val.) Escreva em Python usando o módulo **ply.yacc** um analisador sintático para reconhecer as frases da linguagem definida por **G**. Acrescente Ações Semânticas às produções da gramática para calcular o número de capítulos e o número de entradas por capítulo da encliclopédia; adicionalmente crie uma lista com todas as palavras de abertura de cada entrada.

Questão 5: Assembly da VM (3 val.)

Considere o seguinte programa escrito em Assembly da máquina virtual VM:

```
    PUSHI 0
    PUSHI 0
    PUSHN 10
    PUSHI 0
    START
    PUSHI 0
    STOREG 0
aqui: NOP
    PUSHG 0
    PUSHI 10
    INF
    JZ f
    PUSHG 1
    READ
    ATOI
    ADD
    STOREG 1
    PUSHG 0
    PUSHI 1
    ADD
    STOREG 0
    JUMP aqui
f: NOP
    PUSHG 1
    PUSHI 10
    DIV
    WRITEI
    STOP
```

Responda, então, às alíneas seguintes:

- **a)** (1 val.) Escreva um algoritmo que descreva o comportamento do programa.
- **b)** (1 val.) Explique com clareza qual a lógica das 4 instruções antes da instrução **START**.
- **c)** (1 val.) Explique o que mudava no comportamento do programa se a instrução **JUMP f** fosse substituída pela instrução **NOP**.

Bom trabalho e boa sorte

A equipe docente