Exame de época normal: pl2025-normal

UC: Processamento de Linguagens e Compiladores (LCC, MEFIS)

16 de Dezembro de 2025, 11h-13h, CP2: salas 1.03

Engenharia Informática (3º ano) e Mestrado em Engenharia Física (1º ano)

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

Escreva uma expressão regular que "apanhe":

- **a)** (1 val.) Todos os números reais, incluindo o sinal opcional e o separador decimal obrigatório (ponto ou vírgula). Exemplos: `-3.14`, `+0,5`, `100.0`, `-7.3E-2`, `5.333E+8`;
- **b)** (1 val.) Linhas de um ficheiro de registo (log file) que contenham um endereço de email válido (formato `nome@dominio.ext`), onde o `nome` e o `dominio` consistem em letras minúsculas ou dígitos e o `ext` tem 2 ou 3 letras minúsculas;
- **c)** (1 val.) Uma bitstring (string formada por 0's e 1's) que representa um número que é potência de 2 (considere apenas expoentes inteiros: `[0,1,...]`). Exemplos: `00010`, `1`, `10000`.

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

Considere o seguinte excerto de um ficheiro de configuração simulado:

```
config_data = """
# Configurações do Servidor
Porta=8080;
Timeout=300;
# Outros parâmetros
User=admin;
LogFile=/var/log/server.log;
"""
```

- **a)** (1,5 val.) Escreva um programa Python que utilize o módulo `re` para extrair e imprimir apenas os pares `chave=valor` que contenham um valor numérico (inteiro). Deve ignorar linhas vazias e comentários (linhas que começam com `#`). O resultado deve ser uma lista de strings;
- **b)** (1,5 val.) Escreva uma função Python que utilize o re.sub para substituir todos os identificadores de campos (chaves) que terminam em `Porta` ou `Port` (com ou sem capitalização) pela nova chave `Listen`, por exemplo: `Listen=8080`.

A partir daqui responda numa segunda folha de exame separando as perguntas 1 e 2 das 3,4 e 5.

Questão 3: Gramáticas (5v)

- **a)** (2,5 val.) 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:

- A programação diária é uma lista de programa (sigla, 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;

- **b)** (1 val.) Terminada a especificação apresente uma frase válida na linguagem especificada, juntamente com a sua árvore de derivação.

- **c)** (1,5 val.) 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.

Questão 4: Compilador (6 val.)

Considere a seguinte Gramática Independente de Contexto (G) que define uma estrutura de uma lista simples, onde `id` é um terminal variável (sequência de letras) e o axioma é `Lista`.

```
NT = { Lista, Elementos, Elemento }
T = { '(', ')', ',', id }
P = {
p1: Lista       --> '(' Elementos ')'
p2: Elementos   --> Elemento RestoElementos
p3: RestoElementos --> ',' Elementos
p4: | ε
p5: Elemento    --> id
}
```

id 1
id 1

- **a)** (1 val.) Escreva uma frase válida da linguagem gerada por G, apresentando a respetiva árvore de derivação;

- **b)** (1.5 val.) Escreva, em linguagem algorítmica ou em Python, a função de um parser RD (recursivo-descendente) apropriada para reconhecer o símbolo não-terminal `RestoElementos`;

- **c)** (1.5 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 total de elementos `id` que aparecem na lista. O valor semântico final deve ser o número de elementos.

Questão 5: Assembly da VM (3 val.)

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

```
    PUSHN 2
    PUSHI 100
    STOREG 0
    PUSHI 5
    STOREG 1
START
    JUMP m

f1:
    PUSHL -2
    PUSHL -1
    DIV
    STOREL -3
    RETURN

m:
    PUSHI 0
    PUSHG 0
    PUSHG 1
    PUSHA f1
    CALL
    POP 2
    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 `CALL`.
- **c)** (1 val.) Explique o que mudava no comportamento do programa se a instrução `POP 2` fosse retirada.

Bom trabalho e boa sorte

A equipe docente