Exame de recurso: pl2025-recurso

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

21 de Janeiro de 2026, 14h-16h, Ed.1: sala 2.18

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.) Uma string binária (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`;
- **b)** (1 val.) Todas as strings binárias que contêm pelo menos três carateres e o terceiro é `0`;
- **c)** (1 val.) Todas as strings binárias que contêm pelo menos três `1`.

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

Considere o seguinte excerto de um ficheiro HTML:

```
<!DOCTYPE html>
<html lang="pt-PT">
<head>
    <meta charset="UTF-8">
    <title>Arquivo de Exames: Processamento de Linguagens</title>
</head>
<body>
    <h1>Recursos de Processamento de Linguagens</h1>
    <p>Lista de enunciados de exames e testes práticos:</p>
    <ul>
        <li>
            <a href="pl101e1.pdf">Exame de 2002 (1ª Chamada)</a>:
            Gramática para histórias do "Museu da Pessoa" e filtros de
texto.
        </li>
        <li>
            <a href="pl2023-normal.pdf">Teste Normal 2023</a>:
            Expressões regulares para comentários C (ChatGPT) e gramática
para regras Prolog.
        </li>
        ...
    </ul>
</body>
</html>
```

- **a)** (1,5 val.) Especifique um programa em Python, usando o módulo `re`, que processa o texto HTML na entrada e:

- Extrai os URL de todos os links;
- Ordena alfabeticamente esses URL;
- Coloca na saída a lista de URL numa linha separados por `;`.

- **b)** (1,5 val.) Crie um programa em Python, usando o módulo `re`, que codifica um texto atendendo às seguintes regras:

- os dígitos ficam com o seu código ASCII rodado dum valor inteiro especificado na variável `rotn`;
- as letras minúsculas ficam com o seu código ASCII rodado dum valor inteiro especificado na variável `rotm`;
- as letras maiúsculas ficam com o seu código ASCII rodado dum valor inteiro especificado na variável `rotM`;
- os restantes carateres permanecem inalterados;
- notas adicionais: em Python `chr(n)` dá como resultado o caráter com código ASCII `n`, e `ord(c)` faz o inverso, devolve o código ASCII (um inteiro) correspondente ao caráter `c`;
- Exemplo: a frase de entrada `Dia 8, vamos ter...`, com `rotn=3; rotm=2; rotM=2`, colocaria na saída: `Fkc 1, xcoqu vgt...`.

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 uma lista de compras de acordo com as seguintes definições:

- Uma lista de compras, como o nome indica é uma lista de categorias de produtos a comprar: Fruta, Legumes, Laticínios, Padaria, etc;
- Cada categoria, tem uma designação (Fruta, Legumes, etc) e uma lista de produtos que é preciso comprar;
- Cada produto é caraterizado por uma designação e uma quantidade, por exemplo: `maçã, 2`.

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

- **c)** (1,5 val.) Desenhe o estado inicial do autómato LR(0) e os respetivos estados adjacentes e indique se são visíveis conflitos nesses estados.

Questão 4: Compilador (6 val.)

Considere a seguinte GIC que define uma linguagem para descrever as provas e os concorrentes de um concurso de equitação:

```
NT = { Concurso, Provas, Prova, Concors, Concor, Outros, Atleta. Cav, Pts
}
T = { '.', '(', ')', ',', id, num, CAVALEIRO, CAVALO, PONTOS }
P = {
p1: Concurso --> Provas
p2: Provas   --> Prova '.'
p3:            | Provas Prova '.'
p4: Prova    --> PROVA id Concors
p5: Concors  --> Concor Outros
p6: Outros   --> ε
p7:            | Concor Outros
p8: Concor   --> '(' Atleta ',' Cav ',' Pts ')'
p9: Atleta   --> CAVALEIRO id
p10: Cav     --> CAVALO id
p11: Pts     --> PONTOS num
}
```

- **a)** (1 val.) Escreva uma frase válida da linguagem gerada por G, apresentando a respetiva árvore de derivação;
- **b)** (1.5 val.) Supondo que era possível desenvolver um parser RD (recursivo-descendente) para a linguagem gerada pela gramática apresentada, escreva uma função para reconhecer qualquer símbolo terminal e outra função para reconhecer o símbolo não-terminal `Outros`;
- **c)** (1.5 val.) Escreva em Python usando o módulo `ply.lex` um analisador léxico para reconhecer as frases da linguagem definida pela gramática apresentada (especificação dos tokens, dos carateres a ignorar e o tratamento de erros).
- **d)** (2 val.) Escreva em Python usando o módulo `ply.yacc` um analisador sintático para reconhecer as frases da linguagem definida pela gramática apresentada. Acrescente ações semânticas às produções da gramática para calcular o número de provas, o total de concorrentes e ainda indicar o vencedor de cada prova (o que obtiver o maior número de pontos).

Questão 5: Assembly da VM (3 val.)

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

```
    PUSHI 0
START
    PUSHS "Introduza o valor: "
    WRITES
    READ
    ATOI
    STOREG 0
    PUSHI 0
    PUSHG 0
    PUSHA f1
    CALL
    POP 1
```

```
    WRITEI
STOP

f1:
    PUSHL -1
    PUSHI 2
    MUL
    STOREL -2
    RETURN
```

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 3 instruções antes da instrução `CALL`.
- **c)** (1 val.) Explique o que mudava no comportamento do programa se a instrução `POP 1` fosse retirada.

Bom trabalho e boa sorte

A equipe docente