

Exame de época especial: pl2025-especial

UC: Processamento de Linguagens (LEI)

15 de Julho de 2025, 9h00-11h00, CP2: sala 0.03

Engenharia Informática (3º ano)

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

Escreva uma expressão regular que "apanhe":

- **a)** (1 val.) Todos os endereços IP (IPv4), que são sequências de quatro números de 0 a 255 separados por pontos (e.g., **192.168.1.1** ou **10.0.0.255**);
 - **b)** (1 val.) As matrículas de veículos portuguesas atuais, no formato AA-NN-AA ou NN-AA-NN, onde 'A' representa uma letra maiúscula e 'N' um dígito;
 - **c)** (1 val.) Linhas de um ficheiro de configuração que definem uma porta (e.g., Port = 8080 ou ListenPort: 443), onde o nome do campo pode ser "Port" ou "ListenPort" (case-insensitive), o separador pode ser '=' ou ':' e o valor é um número inteiro.
-

Questão 2: Expressões Regulares (módulo re) (4v = 2+2)

Considere o seguinte excerto de um ficheiro de registo de eventos:

```
log_data = """
[2024-07-20 10:00:15] INFO: Aplicação iniciada com sucesso.
[2024-07-20 10:05:30] WARNING: Espaço em disco baixo (15% disponível).
[2024-07-20 10:10:05] ERROR: Falha na conexão com a base de dados.
[2024-07-20 10:15:20] INFO: Utilizador 'admin' autenticado.
[2024-07-20 10:20:45] ERROR: Erro de escrita em ficheiro: Permissão
negada.
"""
```

- **a)** (2 val.) Escreva um programa Python que utilize o módulo re para extrair todas as linhas que contêm mensagens de nível "ERROR" (incluindo o timestamp e a mensagem), e as imprima. O resultado deve ser uma lista de strings impressas ou enviadas para a saída (**stdout**);
 - **b)** (2 val.) Escreva uma função Python que utilize o re.sub para substituir todos os níveis "INFO" por "DEBUG" no **log_data**, e retorne a string modificada.
-

Questão 3: Gramáticas (5v = 3.5+1.5)

- **a)** (3.5 val.) Escreva, em BNF-puro, uma Gramática Independente de Contexto (GIC) para definir formalmente uma linguagem específica para descrever um "Dicionário de Sinónimos Simples", de acordo com as seguintes definições:

- Um dicionário é uma lista não vazia de entradas;
- Cada entrada tem uma palavra principal (uma sequência de letras minúsculas), seguida por uma lista de sinónimos;
- A palavra principal e a lista de sinónimos são separadas por um sinal de dois pontos `::`;
- A lista de sinónimos é uma sequência não vazia de palavras (sequência de letras minúsculas) separadas por vírgulas `,`;
- Cada entrada deve terminar com um ponto e vírgula `;`;
- Exemplo: `casa:lar,morada;carro:automovel,viatura;`
- **b)** (1.5 val.) Identifique os símbolos terminais que usou na GIC da alínea anterior e escreva o respetivo analisador léxico associando a todos esses símbolos terminais as respetivas expressões regulares.

Questão 4: Compilador (6v = 1+2+1+2)

Considere os terminais variáveis `nome_equipa` (sequência não nula de letras, dígitos e hífen), `data` (data no formato ANSI ISO: AAAA-MM-DD), `max_eq` (string concatenada com um inteiro que representa o número máximo de equipas que se podem aceitar no torneio) e a seguinte Gramática Independente de Contexto (G) em que o axioma é `ListaTorneios` e `"&"` representa a string nula.

```
NT = { ListaTorneios, Torneios, Torneio, Equipas, Equipa }
T = { '(', ')', '[', ']', ',', ':', nome_equipa, DATA, MAX_EQ }
P = { p1:    ListaTorneios  --> '[' Torneios ']'
      p2:    Torneios       --> Torneio Torneios_Resto
      p3, p4: Torneios_Resto --> ',' Torneios | &
      p5:    Torneio        --> '(' data ':' max_eq Equipas ')'
      p6:    Equipas        --> nome_equipa Equipas_Resto
      p7, p8: Equipas_Resto --> ',' nome_equipa Equipas_Resto | & }
```

Neste contexto e após analisar a G dada e a Linguagem L por ela definida, responda às alíneas seguintes:

- **a)** (1 val.) Considere a frase:

```
[ (2025-01-15:MAX_EQ_10 equipa-A, equipa-B),
  (2025-02-20:MAX_EQ_5 equipa-C) ]
```

Apresentando a respetiva árvore de derivação, mostre que esta frase pertence à linguagem definida por G.

- **b)** (2 val.) Prove que G é uma gramática LL1.
- **c)** (1 val.) Escreva, em linguagem algorítmica, a função de um parser RD (recursivo-descendente) apropriada para reconhecer o símbolo não-terminal `Equipas_Resto`.
- **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:
 1. Calcular o número total de equipas registadas em todos os torneios;
 2. Criar uma lista com todos os `nome equipas` únicos presentes na gramática.

Questão 5: Assembly da VM (3v = 2+1)

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

```
PUSHN 3
PUSHI 1
STOREG 0
PUSHI 10
STOREG 1
PUSHI 0
STOREG 2
START
JUMP main

sum_up: NOP
PUSHG 0
PUSHG 2
ADD
STOREG 2
RETURN

main: NOP
loop: NOP
PUSHG 0
PUSHG 1
INF
JZ end_loop
PUSHA sum_up
CALL
PUSHG 0
PUSHI 1
ADD
STOREG 0
JUMP loop

end_loop: NOP
PUSHG 2
WRITEI
STOP
```

Responda, então, às alíneas seguintes:

- **a)** (2 val.) Qual é o comportamento geral deste programa? Descreva-o em linguagem algorítmica, explicando o que calcula e imprime.
- **b)** (1 val.) Suponha que a segunda instrução após "**loop: NOP**", atualmente "**PUSHG 1**", era "**PUSHI 5**". O que mudaria no comportamento do programa e qual seria o resultado final impresso?

Bom trabalho e boa sorte

A equipe docente