Processamento/Teoria de Linguagens e Compilação
LCC (3ºano) + MEFis (1ºano)

Teste final
10 de Janeiro de 2024 (09h00)

Dispõe de **2:00 horas** para realizar este teste

Questão 1: Expressões Regulares (5v = 1+1+1+1+1)

Responda a cada uma das alíneas seguintes:

a) Analise com atenção a expressão regular $e2$:

$$e2 \;=\; (b\ a \mid b\ c)^{+}\ (c\ d \mid c\ b)\ d^{*}$$

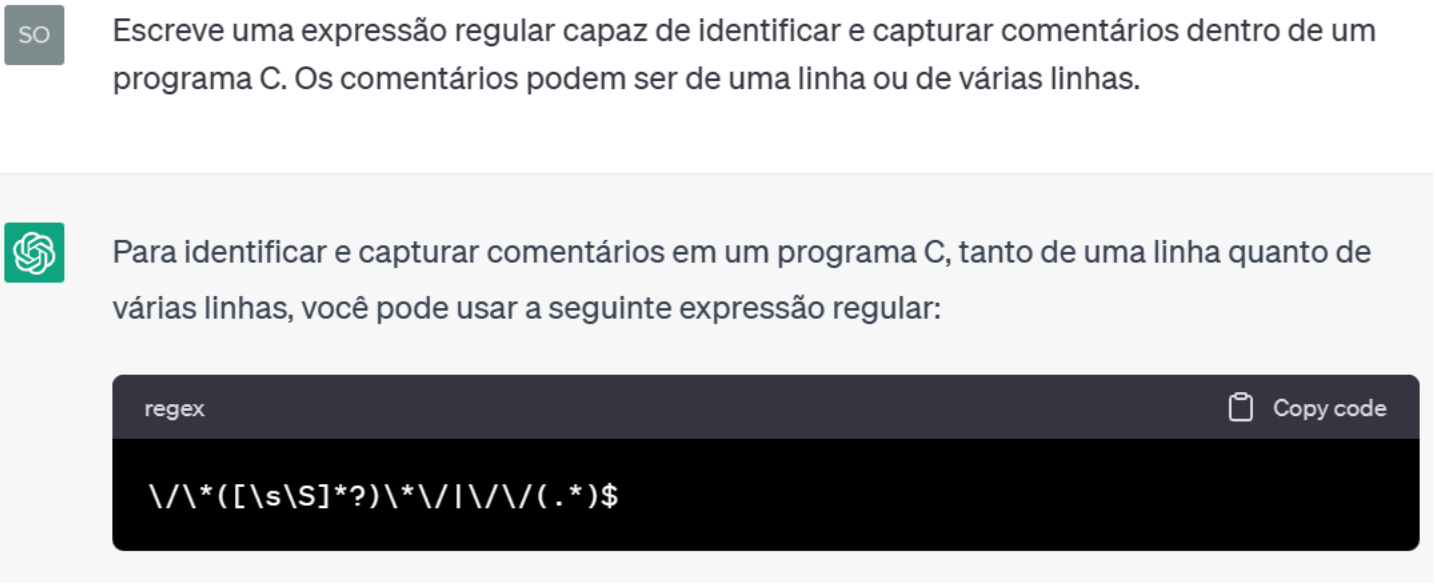
Reduza-a à sua forma mais simples (a sua forma canónica) e depois explique por palavras suas qual a linguagem (quais as frases) definida por esta ER $e2$.

b) Considere a Expressão Regular $e1$ e escreva uma frase válida, derivada de $e1$, e outra que não seja válida justificando as suas respostas

$$e1 \;=\; (a^{*}\ b \mid c^{+})\ d\ c^{*}$$

c) Construa intuitivamente um autómato determinista correspondente a $e1$.

d) Foi colocada ao ChatGPT, o famoso `chatbot` desenvolvido pela OpenAI, a questão que se pode ver na Figura abaixo, onde se mostra também a resposta imediatamente recebida.



Comente a resposta dada pelo ChatGPT, explicando o seu funcionamento e corrigindo-a se necessário; sugira possíveis formas de validar esta resposta. Apresente ainda uma solução alternativa que não recorra aos tokens '$\backslash s$' e '$\backslash S$'.

e) Escreva uma expressão regular que aceite (reconheça) linhas com a estrutura mostrada nos exemplos seguintes:

```
Joao Matias : Q1-T;Q2-F;Q3-F.
Fernando: Q1-T;Q2-T.
Maria Rita Santos: Q1-F;Q2-T;Q3-T; Q4-T.
Marta Mendes de Campos :Q1-F.
```

Questão 2: módulo re (5v = 2+1+1+1)

Recordando o que aprendeu sobre o uso de Expressões Regulares em Python com recurso ao módulo '`re`', responda às alíneas seguintes:

a) Escreva uma função Python que, dado um texto, devolva um novo texto em que as subfrases colocadas entre asteriscos sejam colocadas dentro do comando LaTeX para enfatizar texto em e as subfrases colocadas entre um par de asteriscos sejam colocadas dentro do comando LaTeX para escrever em *Boldface* (*Negrito*).
Exemplo:

```
In: Aqui vai *um* bom **exemplo**
Out: Aqui vai \emph{um} bom \textbf{exemplo}
```

b) Considere o seguinte extrato de um filtro de texto em *Python*:

```
import re
linha = input()
y = re.findall(r'[0-9]+.[0-9]{2}'), linha)
if(len(y)>0):
    print(y[0]," é a primeira de ", len(y), " ocorrências")
else:
    pass
```

e responda às alíneas seguintes:

b1) Diga qual a resposta dada pelo filtro acima se o texto de entrada for

```
linha = "3 hh-+. 3,45.ola.102.34 rrr.1.3rnhh .89-98.  ghhh ."
```

b2) Supondo que o filtro acima produziu a seguinte saída

```
"1.44 é a primeira de 2 ocorrências"
```

Dê um exemplo de um possível texto de entrada atribuído a 'linha'.

b3) Supondo que a expressão regular que define o padrão de pesquisa fosse alterada para

```
r'[0-9]+\.[0-9]{2}'
```

Diga justificando se o comportamento do filtro se mantinha o mesmo ou se o resultado passava a ser diferente para as mesma entradas.

Questão 3: Gramáticas (3v)

OntoDL é uma DSL que serve para descrever ontologias. Para isso permite declarar o conjunto de conceitos (idenficadores começados por uma maiúscula) o conjunto das relações (identificadores começados por uma minúscula) e o conjunto de triplos, sendo que cada triplo é formado por um conceito,uma relação e um conceito, comose exemplifica a seguir
Exemplo de uma frase válida em OntoDL:

```
CONCEITOS { Ca, Cb, Cc, Cd }
RELACOES { r1, r2 }
TRIPLOS { Ca = r1 => Cb;  Ca = r1 => Cc; Ca = r2 => Cd; }
```

Neste exercício pede-se que:

a) Escreva, em BNF-puro, uma Gramática Independente de Contexto (GIC) para definir formalmente a linguagem OntoDL acima descrita.

Questão 4: Compilador (7v = 2+1+1+1+2)

Considere o Terminal variável "num" (número decimal) e a seguinte Gramática Independente de Contexto (G) em que 'S' é o Axioma e '&' representa a string nula.

```
T  = { '.', ';', '[', ']', num }
NT = { S, Is, RI, I }
P = {    p1: S  : Is '.'
         p2: Is : I RI
         p3: RI : &
         p4: RI : I RI
         p5: I  : '[' num ';' num ']'    }
```

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

a) Escreva uma *frase válida* da linguagem gerada por G, apresentando a respetiva *árvore de derivação*.

b) Justifique porque é que G não apresenta **Conflitos LL(1)**.

c) Escreva, em linguagen algorítmica, as 4 funções de um parser RD (recursivo-descendente), uma para reconhecer cada símbolo não-terminal.

d) Escreva em Python usando o módulo '`ply.lex`' um analisador léxico para reconhecer as frases da linguagem definida por G.

e) 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 associar ao valor semântico do Axioma (em Python seria `p[0]` da produção $p1$) o número total de intervalos reconhecidos na frase de entrada.