

MINI TESTE 1

Pergunta 1

Considere a seguinte expressão regular:

```
bin = r'^1*(0|01)*$'
```

E das strings binárias seguintes assinale aquelas que fariam match com a expressão regular apresentada:

- Respostas corretas:
- b. 01010101
 - c. 10001000
 - f. 11110000

Pergunta 2

Considere o seguinte extrato de um filtro de texto em Python (em que o operador ':=' calcula a expressão, atribui o valor à variável e verifica se esse valor é verdadeiro ou falso)

```
import re
import sys

for linha in sys.stdin:
    if s := re.search(r'<[A-Z]+>', linha):
        print(s.group())
    else:
        print("noops!")
```

e selecione as alíneas abaixo que são afirmações verdadeiras:

- Respostas corretas:
- se o texto de entrada for LINHA COM marcas <A> ou <BODY>
 - a resposta do programa é: <BODY>
 - se o texto de entrada for aqui vai <HEAD> bla-bla-BLA </HEAD>
 - a resposta do programa é: noops!

Pergunta 3

Considere o seguinte programa em Python, louco.py:

```
import re
import sys

for linha in sys.stdin:
    if re.search(r'(louc|LOUC)', linha):
        print('0', end='')
    elif re.search(r'[Ll][Oo][Uu][Cc]', linha):
        print('1', end='')
    elif re.search(r'\.', linha):
        print('2', end='')
    elif re.search(r'CURA', linha):
        print('3', end='')
    elif re.search(r'LOUCURA|loucura', linha):
        print('4', end='')
    else:
        pass
```

E considere o seguinte texto de input (baseado num poema de Fernando Pessoa), texto.txt:

```
D. SEBASTIÃO
Rei de Portugal

Louco, sim,
louco, porque quis grandeza
Qual a Sorte a não dá.
Não coube em mim minha certeza;
Por isso onde o areal está
Ficou meu ser que houve, não o que há.

Minha LOUCURA, outros que me a tomem
Com o que nela ia.
Sem a louCURA que é o homem
Mais que a besta sadia,
Cadáver adiado que procria?

Fernando Pessoa, em Mensagem
```

Se o programa fosse invocado da seguinte forma:
\\$ cat texto.txt | python louco.py
Irías obter uma sequência numérica no output.
Preenche a resposta com essa sequência **sem deixar qualquer espaço entre os dígitos**.
Resposta correta: 21.122.021
Intervalo de resposta +/- 0 (21122021 - 21122021)

Pergunta 4

Considere a seguinte frase
 $a = 135 + 512 \cdot a * b$
e selecione as ER abaixo que a podem gerar :

Respostas corretas:

- a. `[a-z] =]* ([a-z]+| [0-9]+| []+| [+\\-*/*])+`
- b. `([a-z] | [0-9]+| []+| [=+\\-*/*])+`
- c. `[a-z] []*=([a-z]+| [0-9]+| []+| [=+\\-*/*])+`

Pergunta 5

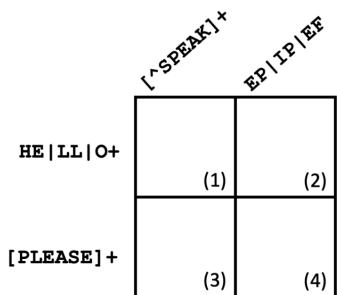
Considere o seguinte input:

```
1.111
-223.145
123
8.88e10
2,000,000
332.333
4.321
-34.123
2.43
```

Respostas corretas: c. `^ -? \d+ (, \d+)* (\. \d+ (e \d+)?) ? $`

Pergunta 6

Considere o seguinte quadro correspondente a umas "palavras cruzadas" de expressões regulares (cada expressão regular define o que pode aparecer numa linha ou numa coluna).



Cada quadrícula, de 1 a 4, deverá ser associada a um carácter.

Faça as associações que achar corretas.

Pergunta	Correspondência correta	Correspondência selecionada
Qual o carácter para (1)?	<input checked="" type="checkbox"/> f. H	<input checked="" type="checkbox"/> c. O
Qual o carácter para (2)?	<input checked="" type="checkbox"/> h. E	<input checked="" type="checkbox"/> c. O
Qual o carácter para (3)?	<input checked="" type="checkbox"/> b. L	<input checked="" type="checkbox"/> b. L
Qual o carácter para (4)?	<input checked="" type="checkbox"/> a. P	<input checked="" type="checkbox"/> h. E

Pergunta 7

A cada cidadão do país Utopia é dado um novo número de cidadão.

Este número segue o seguinte formato:

1. A string pode começar por letras minúsculas até um máximo de 3 (0, 1, 2 ou 3);
2. A seguir deverá conter uma sequência de dígitos. Este segmento deverá ter entre 2 a 8 caracteres inclusive;
3. Depois dos dígitos deverá ter, pelo menos, 3 letras maiúsculas.

Das expressões regulares apresentadas a seguir, assinale aquelas que poderiam reconhecer apenas estes números:

Respostas corretas:

- a. `[a-z]?[a-z]?[a-z]?[0-9]{0,6}[0-9][0-9][A-Z][A-Z][A-Z]*`
- b. `[a-z]{0,3}\d{2,8}[A-Z]{3,}`
- c. `[a-z]{0,3}\d{2,8}[A-Z]{3,}`
- d. `[a-z]{0,3}\d{2,8}[A-Z]{3,}`
- e. `[a-z]?[a-z]?[a-z]?[0-9][0-9][0-9][0-9]{0,6}[A-Z][A-Z][A-Z]*`

MINI TESTE 2

Pergunta 1

Considere o seguinte excerto de código, inserido nas duas imagens seguintes:

```
import ply.lex as lex
import sys

tokens = ['PLUS', 'MINUS', 'SEPARATOR','NUM']
states = [('state1','inclusive')]

def t_state1_NUM (t):
    r'\d+'
    t.lexer.soma+=int(t.value)
    return t

def t_NUM (t):
    r'\d+'
    return t

def t_SEPARATOR (t):
    r','
    return t

def t_PLUS (t):
    r'\+'
    t.lexer.begin("state1")
    return t

def t_state1_PLUS (t):
    r'\+'
    t.lexer.begin("INITIAL")
    return t
    t_ignore='\n\t '

lexer = lex.lex()
lexer.soma = 0

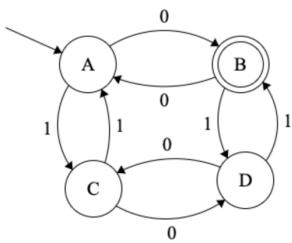
frase = "1,+1,2,+1,1,+3,+"
lexer.input(frase)
for tok in lexer:
    pass
print(lexer.soma)
```

Das afirmações seguintes assinale as que estão corretas:

- Respostas corretas: b. O programa dá como resultado 6.
 d. A declaração de `state1` como `inclusive` é necessária para o programa não abortar com uma situação de erro.

Pergunta 2

Considere o seguinte autómato finito determinista:



E das afirmações seguintes, assinale as que são verdadeiras:

- Respostas corretas: a. A frase "01100001111" pertence à linguagem definida pelo autómato.
 e. A frase "0" pertence à linguagem definida pelo autómato.

Pergunta 3

Considere o seguinte texto:

```
Oh Laurindinha, vem à janela.
Verso 1 de 4
Oh Laurindinha, vem à janela.
Verso (2 de 4)
Ver o teu amor, (ai ai ai) que ele vai p'ra guerra. Verso 3
de 4
Ver o teu amor, (ai ai ai) que ele vai p'ra guerra.
Verso 4 de 4
```

Considere agora o seguinte programa, onde a variável `texto` é o texto acima.

```
import re

# a função subn retorna um tuplo onde o primeiro elemento
# é o texto resultante de executar substituições no texto
# e o segundo elemento é o número de substituições feito.
(texto, n) = re.subn(r'Verso (\d+) de \1', '#####', texto)
print(n)
```

- Respostas corretas: A ER captura apenas 1 grupo (sem contar com o grupo 0).
 Quando executado, o programa imprime '1'.

Pergunta 4

Observe com atenção o seguinte filtro de texto implementado com o LEX do Python:

```
import ply.lex as lex
import sys

states = [('marcado', 'inclusive')]

tokens = ('MA', 'MF', 'MARA', 'IGN')

def t_MA(t):
    r'<[^>]+>'
    t.lexer.begin('marcado')

def t IGN(t):
    r'.|\n'

t_ignore = ' \t\r\n'

def t_marcado_MF(t):
    r'</[^>]+>'
    print(t.lexer.conteudo)
    t.lexer.conteudo = ""
    t.lexer.begin('INITIAL')

def t_marcado_MARA(t):
    r'.|\n'
    t.lexer.conteudo += t.value

t_marcado_ignore = ''

def t_error(t):
    t.lexer.skip(1)
```

Respostas corretas:

- c. Se o texto de entrada for:
agora <aqui attr="8">sim vai fica
O resultado será formado por 1 linha com "sim vai".
- d. Se o texto de entrada for:
agora <aqui>sim vai</aqui> fica <ab>123 ola 456</ab>
O resultado será formado por 2 linhas uma com "sim vai" e a outra com "123 ola 456".

Pergunta 5

Considere o seguinte analisador léxico:

```
import ply.lex as lex

tokens = ( 'INI', 'FIM', 'nome', 'real', 'int', 'sinal' )

t_sinal = r'[=+\-*()]*'

def t_INI(t):
    r'(?:begin)|\{'
    return t

def t_FIM(t):
    r'\}|(?:end)|;|'
    return t

def t_nome(t):
    r'[a-zA-Z][a-zA-Z0-9]*'
    return t

def t_real(t):
    r'[0-9]+\.[0-9]+'
    return t

def t_int(t):
    r'[0-9]+'
    return t

t_ignore = " \n\t"

def t_error(t):
    # ignorar erro e continuar a análise
    t.lexer.skip(1)

lexer = lex.lex()
```

Respostas corretas:

- Se o texto de entrada for:
{x == ab12-End}
a sequência de símbolos retornados é:
INI nome sinal sinal nome sinal FIM FIM

- Se a sequência de símbolos retornados for:
int sinal real sinal nome
o texto de entrada pode ser
4*5.6 = FIM

Pergunta 6

Considere a seguinte expressão regular:

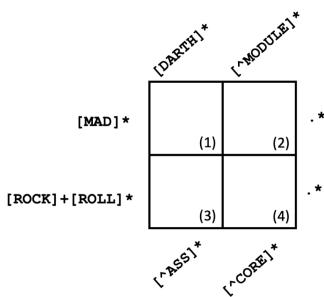
$\text{bin} = r'^0*(1|10)*\$$

E das strings binárias seguintes assinale aquelas que fariam match com a expressão regular apresentada:

- Respostas corretas:
- a. 01010101
 - c. 01110111
 - d. 110110110110

Pergunta 7

Considere o seguinte quadro correspondente a umas "palavras cruzadas" de expressões regulares (cada expressão regular define o que pode aparecer numa linha ou numa coluna).



Cada quadrícula, de 1 a 4, deverá ser associada a um carácter.

Faça as associações que achar corretas.

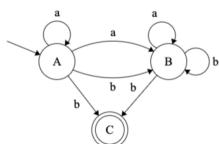
- | Pergunta | Correspondência correta | Correspondência selecionada |
|---------------------------|--|--|
| Qual o carácter para (1)? | <input checked="" type="checkbox"/> c. D | <input checked="" type="checkbox"/> c. D |
| Qual o carácter para (2)? | <input checked="" type="checkbox"/> b. A | <input checked="" type="checkbox"/> b. A |
| Qual o carácter para (3)? | <input checked="" type="checkbox"/> g. R | <input checked="" type="checkbox"/> g. R |
| Qual o carácter para (4)? | <input checked="" type="checkbox"/> a. K | <input checked="" type="checkbox"/> a. K |

Pergunta 8

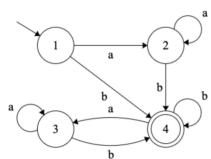
Considere a seguinte expressão regular: $a^* (a \mid b)^* b$

Considere os seguintes autómatos:

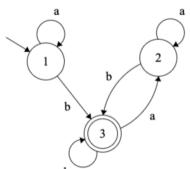
Autómato A1



Autómato A2



Autómato A3



- Respostas corretas:
- a. O automáto A3 é um Autómato Finito Determinista (AFD) capaz de reconhecer a linguagem definida pela expressão regular.
 - d. O automáto A2 é um Autómato Finito Determinista (AFD) capaz de reconhecer a linguagem definida pela expressão regular.

MINI TESTE 3

Pergunta 1

Considere o seguinte excerto dum módulo em Python que faz o reconhecimento de uma árvore binária de números inteiros.

Considere que o analisador léxico retorna o valor associado a num no tipo inteiro.

Preencha as ações semânticas necessárias de modo a que o parser escreva o somatório de todos os nodos contidos na árvore.

Com a entrada: (23 (5 () ()) (47 () ()))

deverá escrever: 75

Não insira espaços em branco desnecessários nas expressões que vai escrever.

```
def p_Axioma(p):
    "Axioma : ABin"
    print([Exp1])

def p_ABin_empty(p):
    "ABin : (' )"
    [Exp2]

def p_ABin(p):
    "ABin : (' num ABin ABin )"
    [Exp3]
```

Respostas corretas para: Exp1

Método de avaliação

Correspondência exata

Resposta correta

p[1]

Respostas corretas para: Exp2

Método de avaliação

Correspondência exata

Resposta correta

p[0]=0

Respostas corretas para: Exp3

Método de avaliação

Correspondência exata

Resposta correta

p[0]=p[2]+p[3]+p[4]

Pergunta 2

Considere o seguinte excerto de um programa Python que utiliza a biblioteca PLY para gerar um processador da linguagem definida pela GIC inserida no código.

NOTA IMPORTANTE: `print('c' * n)` em Python significa imprimir 'n' vezes o caractere 'c'.

```
def p_query_composition(p):
    "query : 't' composition 'NUM'"
    print(p[2] + str(p[4]) + ')'
    p.parser.conta_paren += 1

def p_query_function(p):
    "query : ID NUM"
    print(p[1] + '(' + str(p[2]) + ')')

def p_composition_list(p):
    "composition : composition '.' ID"
    p.parser.conta_paren += 1
    p[0] = p[1] + p[3] + '.'

def p_composition_one(p):
    "composition : ID '.' ID"
    p.parser.conta_paren = 2
    p[0] = p[1] + p[3] + '.'

def p_error(p):
    print("Syntax error: ", p)
    p.parser.success = False
```

Das alíneas seguintes selecione as que são verdadeiras.

- Respostas corretas:
- c. Se o input for: (f . g . h) 2 o programa imprimirá: f(g(h(2)))
 - d. A frase f (2) seria um possível resultado da execução deste processador.

Pergunta 3

Considere a seguinte GIC descrita pelas suas produções, em que as maiúsculas representam símbolos não terminais e as minúsculas, os símbolos terminais:

```
p1: A -> B C
p2: B -> D E
p3: C -> ε
p4: C -> a A
p5: C -> b A
p6: D -> ε
p7: D -> f
p8: D -> g A h
p9: E -> ε
p10: E -> c B
p11: E -> d B
```

A seguir associe a cada produção indicada o seu lookahead.

- | Pergunta | Correspondência correta | Correspondência selecionada |
|---------------------------|---|---|
| lookAhead(p1: A -> B C) | <input checked="" type="radio"/> d, {e, f, g} | <input checked="" type="radio"/> d, {e, f, g} |
| lookAhead(p3: C -> ε) | <input checked="" type="radio"/> c, {§, h} | <input checked="" type="radio"/> h, {§, a, b} |
| lookAhead(p9: E -> ε) | <input checked="" type="radio"/> h, {§, a, b} | <input checked="" type="radio"/> g, {§, c, d} |
| lookAhead(p8: D -> g A h) | <input checked="" type="radio"/> a, {g} | <input checked="" type="radio"/> a, {g} |

Pergunta 4

6,666 de 10 pontos

Considere as seguintes produções duma GIC que define uma linguagem para a especificação de listas, em que para os símbolos se seguiram as convenções usadas nas aulas teóricas:

```

Lista -> '[' ICont
ICont -> ']'
ICont -> num ElemsCont ']'

ElemsCont -> ',' num ElemsCont
ElemsCont -> e

Depois de analisar a gramática assinale as afirmações verdadeiras.

Respostas corretas: Se construirmos um parser recursivo descendente, o reconhecimento do símbolo ICont seria:
def rec_ICont():
    global prox_simb
    if prox_simb == ']':
        rec_term(']')
    elif prox_simb == 'num':
        rec_term('num')
        rec_ElemsCont()
        rec_term(']')
    else:
        erro()

✓ a.

```

Pergunta 5

Considere as seguintes produções duma GIC, em que os símbolos não terminais são representados por maiúsculas e os símbolos terminais por minúsculas:

```

S -> A a S
S -> A
A -> B b A
A -> B
B -> c S c
B -> d

```

Depois de analisar a gramática assinale as afirmações verdadeiras.

Respostas corretas: ✓ a. A gramática não é LR(0) porque apresenta conflitos redução/transição em vários estados do seu autómato.
✓ c. Follow(S) = {c, \$}

Pergunta 6

0 de 10 pontos

Considere o seguinte excerto de um ficheiro Python (escrito usando o módulo PLY) que implementa um processador que reconhece uma sequência de intervalos fechados de números naturais e diz que falhou ou escreve um resultado (neste caso, a dimensão do intervalo).

Considere que o analisador léxico retorna o valor associado a NUM no tipo inteiro.

```

def p_sequencia(p):
    "sequencia : sentido intervalos"
    if p[2] == -1:
        print("falhou!")
    else:
        print("resultado: ",p[2])

def p_sentidoA(p):
    "sentido : '+'"
    global flag
    flag = True
    p.parser.last = 0

def p_sentidoD(p):
    "sentido : '-'"
    global flag
    flag = False
    p.parser.last = 100000000

def p_intervalos_intervalo(p):
    "intervalos : intervalo"
    p[0] = p[1]

def p_intervalos_intervalos(p):
    "intervalos : intervalos intervalo"
    p[0] = p[2] if (p[1]==-1) else -1

def p_intervalo(p):
    "intervalo : '[' NUM ',' NUM ']'"
    p[0] = -1
    if (flag and (p[2] > p.parser.last)) and (p[4] >= p[2]):
        p[0] = p[4] - p[2]
    p[0] = (not flag) and (p[2] < p.parser.last)) and (p[4] <= p[2]):
        p[0] = p[2] - p[4]
    p.parser.last = p[4]

Respostas corretas: Se o texto de entrada for:
✓ +[3,3] [4,16] [15,20]
o processador deverá escrever:
falhou!
✓ b.
Se o texto de entrada for:
✓ +[3,10] [11,14] [20,30]
o processador deverá escrever:
✓ d. resultado: 10

```

Pergunta 7

Dadas as seguintes duas frases:

```

email:{ jj@jcr : email:{ jj@jcr : Envio de conteudo para as aulas. .}
email:{ jj@jcr : Envio de conteudo para as aulas. }

```

Selecione a gramática que poderia reconhecer ambas as frases.

Considere o terminal TEXTO que representa strings(com espaços incluídos), assim como ORIGEM e DESTINO que representam uma sequência de um ou mais caracteres sem espaço.

Respostas corretas: ✓ a. p1: Email -> 'email:{' Conteudo '}'
✓ p2: Conteudo -> ORIGEM '@' DESTINO ':' Corpo '.'
✓ p3: Corpo -> TEXTO
✓ p4: Corpo -> Email
✓ b.
✓ p1: Email -> Abre Conteudo Fecha
✓ p2: Conteudo -> ORIGEM '@' DESTINO ':' Corpo '.'
✓ p3: Corpo -> TEXTO
✓ p4: Corpo -> Email
✓ p5: Abre -> 'email:{'
✓ p6: Fecha -> '}'
✓ c.

Pergunta 8

Considere os Terminais str (texto entre aspas), texto (sequência de caracteres exceto '<' e '>') e id (sequência não nula de letras) e a seguinte Gramática Independente de Contexto (GIC):

```
p1: Anota -> Abre texto Fecha  
p2: Abre -> '<' id '>'  
p3:           | '<' id' LstA '>'  
p4: Fecha -> '<' '/' id '>'  
p5: LstA -> Atr  
p6: LstA -> LstA Atr  
p7: Atr -> id '=' str
```

Selecione, então, das alíneas abaixo a afirmação verdadeira:

Respostas corretas:

A frase:
<ttt a="1">bla bla</ttt>
pertence à linguagem L(G) gerada por esta gramática.

- b.
A frase:
<ttt>ola ole oli</z>
pertence à linguagem L(G) gerada por esta gramática.
- d.

MINI TESTE 4

Pergunta 1

Suponhamos que a máquina virtual tem uma instrução nova:

ITE - retira da pilha os valores b, y, x. Se b não for nulo, coloca na pilha x, caso contrário coloca na pilha y.

Assinale as respostas verdadeiras:

Respostas corretas:	- Suponha que o endereço de uma variável a é 0. O seguinte código máquina: PUSHI 1 PUSHI 2 ADD PUSHI 1 PUSHI 2 SUB PUSHG 0 PUSHI 2 EQ ITE poderia representar: a. $a + 2$ if $a == 2$ else $1 - 2$ b. Este operador não poderia ser utilizado para implementar um bloco de instruções condicional (<i>if/else</i>), porque estes podem não retornar valores. Suponha que o endereço da variável a é 0. PUSHI 0 PUSHI 0 SUB PUSHG 0 PUSHG 0 PUSHI 0 INF ITE d. calcula o valor absoluto de a.
---------------------	--

Pergunta 2

Considere o seguinte programa escrito numa linguagem de programação LP, semelhante à que se trabalhou nas aulas:

```
int n quad
n = int(input("Introduza um inteiro positivo:"))
repeat(n)
    quad = n*n
    print(n, ":", quad)
    n = n - 1
}
```

Considere também o seguinte código gerado pelo compilador onde faltam as instruções I1, I2, I3, I4 e os argumentos A1, A2, A3.
Complete o código preenchendo os espaços correspondentes às instruções e argumentos em falta (use minúsculas sempre que escrever letras).

```
pushi 0
pushi 0
start
pushs "Introduza um inteiro positivo:"
writes
read
atoi
storeg 0
pushg 0
label1:
pushl 0
[A1] [A1]
pushg 0
pushg 0
mul
storeg 1
pushl 0
writei
pushs "::"
writes
pushg 1
writei
[I4] [A3]
pushi 1
[I3]
storeg 0
pushl 0
pushi 1
sub
storel 0
[I2] [A2]
label12:
stop
```

Respostas corretas para: I1

Método de avaliação	Resposta correta
Contém	jr

Respostas corretas para: A1

Método de avaliação	Resposta correta
Contém	label2

Respostas corretas para: I4

Método de avaliação	Resposta correta
Contém	pushg

Respostas corretas para: A3

Método de avaliação	Resposta correta
Contém	0

Respostas corretas para: I3

Método de avaliação	Resposta correta
Contém	sub

Respostas corretas para: I2

Método de avaliação	Resposta correta
Contém	jump

Respostas corretas para: A2

Método de avaliação	Resposta correta
Contém	label1

Pergunta 3

2,5 de 10 pontos

Para descrever o output gerado por programas diversos, quando sujeitos a testes, definiu-se uma linguagem específica que permite identificar o teste e o valor esperado à saída. Mostram-se abaixo 5 exemplos de frases válidas dessa linguagem:

```
01: 12.35
Sai1: rosa
Out3: SEQ ( rosa, tulipa, camelia, jasmim )
04: TUP < TUP<10,20>, 4, SEQ(1, 2.5, 3, 4.7) >
Sai5: SEQ( TUP<1,2>, TUP<3,4>, TUP<5,6> )
```

O Objetivo é desenvolver um processador para essa linguagem que reconheça cada frase e produza algumas informações sobre o valor esperado, como se exemplifica abaixo:

```
Teste 01 valor atómico (real)
Teste Sai1 valor atómico (pal)
Teste Out3 valor composto (sequencia com 4 elementos)
Teste 04 valor composto (tuple com 3 elementos)
Teste Sai5 valor composto (sequencia com 3 elementos)
```

Especifique uma gramática para a linguagem indicando o conjunto de símbolos terminais (T), o conjunto de símbolos não-terminais (N), o axioma (S) e o conjunto de produções (P).

Pergunta 4

Considere o excerto abaixo de uma gramática, escrita na notação do ply-yacc, para definir conjuntos em comprehensão. Onde "aexpr" é uma expressão aritmética e "bexpr" é uma expressão booleana, tal como foram mostradas nas aulas.

Notas:

1. tanto aexpr como bexpr retornam uma string com a expressão e não o resultado. Por exemplo, se no texto estiver '1 + 2', o valor retornado por aexpr é '1 + 2' e não 3.

2. Em Python, o uso de """ ... """ representa strings multi-linha.

```
def p_setc_no_pred(p):
    "setc : '{' expr '||' ID IN expr '}'"
    p[0] = f"""
        "output_expr": "{p[2]}",
        "var": "{p[4]}",
        "input_set": "{p[6]}"
    """

def p_setc_pred(p):
    "setc : '{' expr '||' ID IN expr ',' predicates '}'"
    p[0] = f"""
        "output_expr": "{p[2]}",
        "var": "{p[4]}",
        "input_set": "{p[6]}",
        "predicates": {p[8]}
    """

def p_predicates_list(p):
    "predicates : predicates ',' bexpr"
    p[0] = p[1] + [p[3]]

def p_predicates_one(p):
    "predicates : bexpr"
    p[0] = [p[1]]

def p_expr(p):
    "expr : aexpr | bexpr"
    p[0] = p[1]
```

Selecione as afirmações verdadeiras:

Respostas corretas:

- Se o texto de entrada for
{ x - 1 | x in N, x > 0, x % 2 == 0 }
o resultado será:
"output_expr": "x - 1",
"var": "x",
"input_set": "N",
c. "predicates": ["x > 0", "x % 2 == 0"]

Pergunta 5

Considere a gramática sobre Expressões amplamente trabalhada nas aulas à qual se adicionaram ações semânticas para a construção duma AST:

```

def p_Exp(p):
    "Z : Exp"
    p[0] = Exp('expressao', p[1])

def p_Exp_ad(p):
    "Exp : Exp '+' Term"
    p[0] = Operacao('operacao', 'soma', [p[1], p[3]])

def p_Exp_sub(p):
    "Exp : Exp '-' Term"
    p[0] = Operacao('operacao', 'subtracao', [p[1], p[3]])

def p_Exp_term(p):
    "Exp : Term"
    p[0] = p[1]

def p_Term_mul(p):
    "Term : Term '*' Factor"
    p[0] = Operacao('operacao', 'multiplicacao', [p[1], p[3]])

def p_Term_div(p):
    "Term : Term '/' Factor"
    p[0] = Operacao('operacao', 'divisao', [p[1], p[3]])

def p_Term_factor(p):
    "Term : Factor"
    p[0] = p[1]

def p_Factor_num(p):
    "Factor : num"
    p[0] = Operando('operando', p[1])

```

Considere que se definiram as seguintes classes para suportar a AST:

```

class Exp:
    def __init__(self, tipo, exp):
        self.tipo = tipo
        self.exp = exp

    def calcVal(self, nodo):=
```

```

class Operacao:
    def __init__(self, tipo, operacao, operandos=None):
        self.tipo = tipo
        self.operacao = operacao
        if operandos:
            self.operandos = operandos
        else:
            self.operandos = []
```

```

class Operando:
    def __init__(self, tipo, valor):
        self.tipo = tipo
        self.valor = valor
```

Uma das vantagens da construção da AST é podermos obter o resultado final que pretendemos com uma simples travessia. Completa a seguinte travessia que calcula o valor da expressão guardada na AST (este método está inserido na classe Exp), **sem utilizar espaços brancos nas respostas**:

```

class Exp:
    ...
    def calcVal(self, nodo):
        if nodo.tipo == 'expressao':
            return self.calcVal([Var1])
        elif nodo.tipo == 'operacao':
            if nodo.operacao == 'soma':
                return self.calcVal([Var2]) + self.calcVal([Var3])
            elif nodo.operacao == 'subtracao':
                return self.calcVal(...) - self.calcVal(...)
            elif nodo.operacao == 'multiplicacao':
                return self.calcVal(...) * self.calcVal(...)
            elif nodo.operacao == 'divisao':
                return self.calcVal(...) / self.calcVal(...)
        elif nodo.tipo == [Var4]:
            return int([Var5])

```

Respostas corretas para: Var1

Método de avaliação	Resposta correta
Contém	nodo.exp

Respostas corretas para: Var2

Método de avaliação	Resposta correta
Contém	nodo.operandos[0]

Respostas corretas para: Var3

Método de avaliação	Resposta correta
Contém	nodo.operandos[1]

Respostas corretas para: Var4

Método de avaliação	Resposta correta
Contém	'operando'
Contém	"operando"

Respostas corretas para: Var5

Método de avaliação	Resposta correta
Contém	nodo.valor

Pergunta 6

Considere a seguinte gramática:

```
Z : S $  
S : A a  
| b  
A : A a  
| ε
```

Selecione as afirmações seguintes que são verdadeiras:

- Respostas corretas:
- b. A gramática tem um conflito redução/redução no seu autômato LR(0).
 - c. A gramática tem um conflito no estado inicial do seu autômato LR(0).
 - d. O autômato SLR(1) para a mesma gramática já não apresenta conflitos.

Pergunta 7

Considere a seguinte regra gramatical escrita na notação do ply-yacc, que foi acrescentada à linguagem das expressões aritméticas que foi estudada nas aulas:

```
def p_expr_or(p):  
    "expr : expr \"|\" term"  
    p[0] = f"""  
    {p[1]}  
    DUP 1  
    JZ next{count}  
    JUMP end{count}  
    next{count}:  
    POP 1  
    {p[3]}  
    end{count}:  
    """
```

Nota: em Python, o uso de """ ... """ representa strings multi-linha.

Assinale as afirmações verdadeiras:

- Respostas corretas:
- a. Ao ser executado, o programa P abaixo imprimirá 2
a = 0
b = 0
 - b. print (a | b | 2)
Ao ser executado, o programa P abaixo imprimirá -1
a = 2
 - c. print (-1 | a)

Pergunta 8

5 de 10 p

Considere a seguinte gramática correspondente a expressões numéricas LISP muito simplificadas:

```
S : Exp ;  
Exp : int  
| '(' Funcao ')' ;  
Funcao : '*' Lista  
| '**' Lista ;  
Lista : Exp Lista  
|
```

Preencha as lacunas no texto que se segue conforme é pedido.
É [V1] (preencha com V ou F) que a gramática apresentada é LL(1).

Escreva a frase da linguagem [V2] correspondente à expressão aritmética $2 * 3 + 8 * 2$ (coloque apenas um espaço entre os símbolos terminais).

E [V3] (preencha com V ou F) que o lookahead(Lista :), última produção, é LA(Lista :) = ('.'

Escreva a frase da linguagem [V4] correspondente à expressão aritmética $2 * 2 * 2 * (5 + 2)$ (coloque apenas um espaço entre os símbolos terminais crie apenas os níveis de profundidade necessários).

Respostas corretas para: V1

Método de avaliação	Resposta correta	Sensibilidade a maiúsculas e minúsculas
<input checked="" type="checkbox"/> Correspondência de padrão	[]*[Vv][]*	

Respostas corretas para: V2

Método de avaliação	Resposta correta	Sensibilidade a maiúsculas e minúsculas
<input checked="" type="checkbox"/> Correspondência de padrão	[]*[]*[]*[]*[]+2[]+3[]*[]*[]*[]+8[]+2[]*[]*[]*[]*[]*	

Respostas corretas para: V3

Método de avaliação	Resposta correta	Sensibilidade a maiúsculas e minúsculas
<input checked="" type="checkbox"/> Correspondência de padrão	[]*[Ff][]*	

Respostas corretas para: V4

Método de avaliação	Resposta correta	Sensibilidade a maiúsculas e minúsculas
<input checked="" type="checkbox"/> Correspondência de padrão	[]*[]*[]*[]*[]+2[]+2[]*[]*[]*[]+5[]+2[]*[]*[] []*[]*	