UNIVERSIDADE FEDERAL DO MARANHÃO
DEPARTAMENTO DE ENGENHARIA DA COMPUTAÇÃO

P1

Curso: Engenharia da Computação	**Ano / Semestre:**	**Data:**
Disciplina: Linguagens Formais e Autômatos	**Professor:** Thales Levi Azevedo Valente	
Aluno:	**Código:**	

OBS: RESPOSTAS DEVEM SER ENTREGUES À CANETA AZUL OU PRETA. CASO O ALUNO USE LÁPIS, O MESMO NÃO TERÁ DIREITO A REVISÃO DAS QUESTÕES

1. Por exemplo, sejam os conjuntos A = {1,2,3}, B = {x,y,z} e R = {(1, y),(1,z),(3,y)}. Diga quem é o domínio, imagem e a inversa da relação R

Dado:

- **Domínio de R:** conjunto dos primeiros elementos de cada par da relação
  ⇒ **domínio** D={1,3}
- **Imagem de R:** conjunto dos segundos elementos de cada par da relação
  ⇒ **imagem** Im={y,z}
- **Inversa de R (R^{-1}):** troca a ordem dos pares
  ⇒ **R^{-1} = {(y, 1), (z, 1), (y, 3)}**

2. Dado o Grafo G2, defina a tupla G2, o conjunto V2, o conjunto A2, classifique quanto a sua orientação, defina a ramificação de saída de cada só Ns(x), a ramificação de entrada de cada nó Ne(x), o vértice-raíz e o vértice-folha (se existirem).

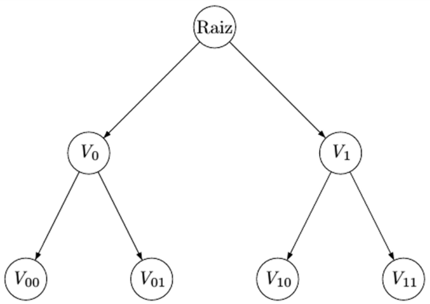

Figura: Árvore

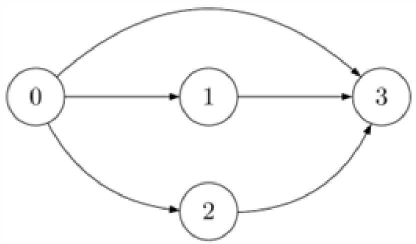

Figura: Grafo G_2

Tupla que define G_2:

- G2=(V2,A2), onde:
  - **V_2 (vértices)** = {0, 1, 2, 3}
  - **A_2 (arestas orientadas)**: {(0,1), (0,2), (0,3), (1,3), (2,3)}
- O grafo é **orientado (dirigido)**, pois as arestas possuem **sentido definido** (setas).
- Ramificação de saída Ns(x): Número de arestas que saem de cada vértice:
  - Ns(0) = 3 → {(0,1), (0,2), (0,3)}
  - Ns(1) = 1 → {(1,3)}
  - Ns(2) = 1 → {(2,3)}
  - Ns(3) = 1 → ∅
- Ramificação de entrada Ne(x): Número de arestas que entram em cada vértice:
  - Ne(0) = 0 → ∅
  - Ne(1) = 1 → {(0,1)}
  - Ne(2) = 1 → {(0,2)}
  - Ne(3) = 3 → {(0,3), (1,3), (2,3)}
- Vértice-raiz: **Não existe vértice raiz**, pois todos os nós possuem pelo menos uma aresta de entrada.
- Vértice-folha: **Não existe vértice folha**, pois todos os nós possuem pelo menos uma aresta de saída.

3. Explique porque o grafo "Figura Árvore" é classificado como uma árvore.

A figura à esquerda mostra um grafo em forma de **árvore binária**. Esse grafo é classificado como **árvore** pelos seguintes motivos:

- ✅ **Conectado**: há caminho entre a raiz e qualquer outro nó.
- ✅ **Arestas direcionadas para os filhos** : não há retorno para os vértices anteriores.
- ✅ **Sem ciclos**: nenhum vértice é visitado mais de uma vez.
- ✅ **Estrutura hierárquica**: possui um **único nó raiz** (no topo), de onde partem as demais conexões, e **vários nós-folha**, que não possuem filhos.
- ✅ **Unicidade de caminhos**: para qualquer vértice na árvore, existe **apenas um caminho** que o conecta diretamente à raiz, o que garante a ausência de ambiguidade nas relações.

4. Considerando o alfabeto Σ1 = {a,b,c}, a linguagem $L = \{a^i b^i c^i | i \geq 1\}$, o alfabeto Σ2 = {x,y,z}, e as regras de substituição s(a) = {x}, s(b) = {y,yy}, s(c) = {z,zz,zzz}. Digamos w = abc, pergunta-se: xyyyyzzz pertence a s(abc) ? Explique sua resposta.

Dado:

- Alfabeto Σ1={a,b,c}
- Regras de substituição:

    - s(a)={x}s(a)
    - s(b)={y,yy}
    - s(c)={z,zz,zzz}
- Palavra original: w=abc
- Palavra-alvo: xyyyyzzz

Vamos verificar as possibilidades:

- s(a)=x
- s(b)=y ou yyyyyy
- s(c)=z, zzzzzz, ou zzzzzzzzz

Agora, quebrar xyyyyzzz:

- Começa com **x**, que só pode vir de **a**
- Seguido de **yyyy**: só pode ser yy+yyyy + yyyy+yy → então, duas vezes bbb substituído por yyyyyy? **Não é permitido pois temos só um b**

Logo, NÃO pertence a s(abc), pois não é possível obter **4 y's** com **apenas um b**, mesmo considerando yy

5. Considere a gramática G2 = (V2,Σ2,P2,S), com: V2 ={a,b,c,S,B,C}, Σ2 ={a,b,c}, P2 ={S →aSBC,S →abC,CB →BC,bB →bb,bC →bc,cC → cc}. A linguagem definida por essa gramática é $L_2(G_2) = \{a^n b^n c^n | n \geq 1\}$. Pergunta-se: como a sentença **aabbcc** foi gerada utilizando G2.

Gramática:

- Variáveis: V2={a,b,c,S,B,C}
- Terminais: Σ2={a,b,c}
- Produções:
    - S→aSBC
    - S→abC
    - CB→BC
    - bB→bb
    - bC→bc
    - cC→cc

Objetivo: gerar aabbcc

Vamos fazer a derivação passo a passo:

1. **S** (S ⇒ aSBC)
2. ⇒ a**S**BC (S ⇒ abC)

3. ⇒ aab**CB**C (CB ⇒ BC)
4. ⇒ aa**bB**CC (bB ⇒ bb)
5. ⇒ aab**bC**C (bC ⇒ bc)
6. ⇒ aabb**cC** (cC ⇒ cc)
7. ⇒ aabbcc

✅ **Sentença aabbcc pode ser gerada por G2.**