

Lógica y Matemática Computacional
Licenciatura en Sistemas de Información

ARBOLES

Ing. JULIO C. ACOSTA

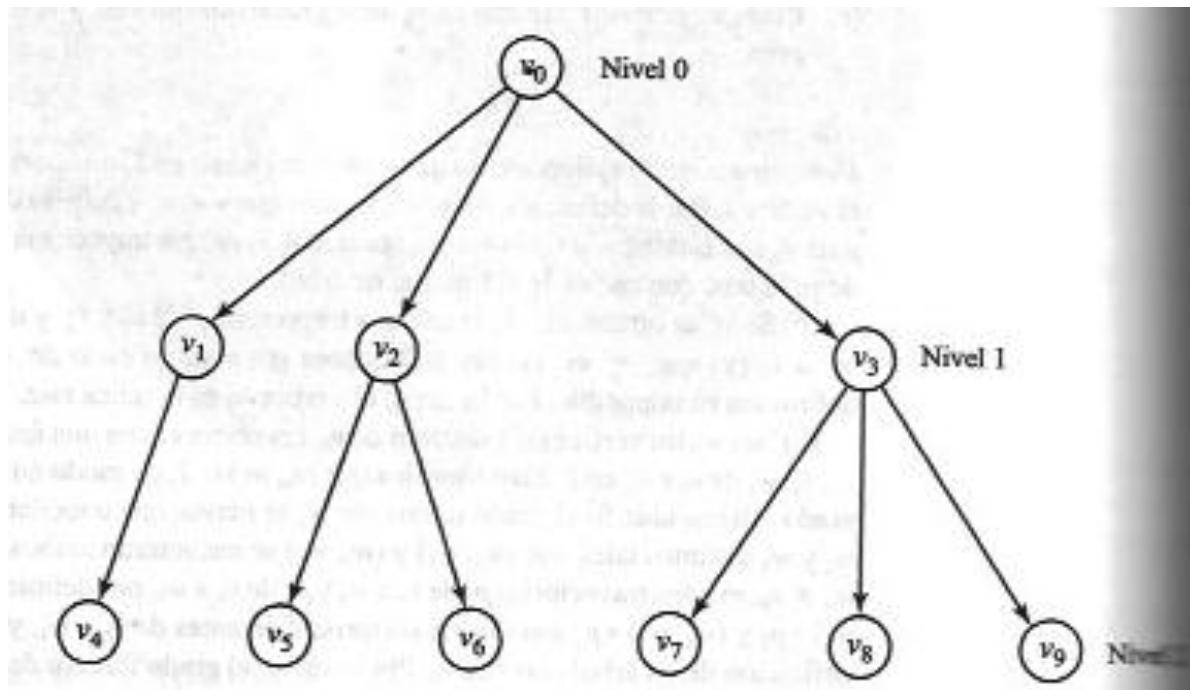
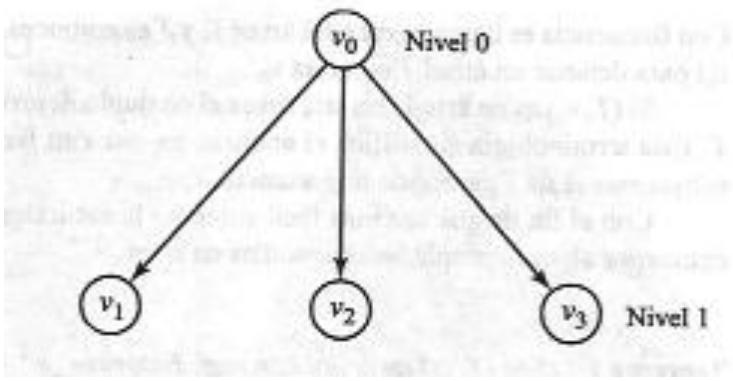
Facultad de Ciencias Exactas y Naturales y Agrimensura - UNNE

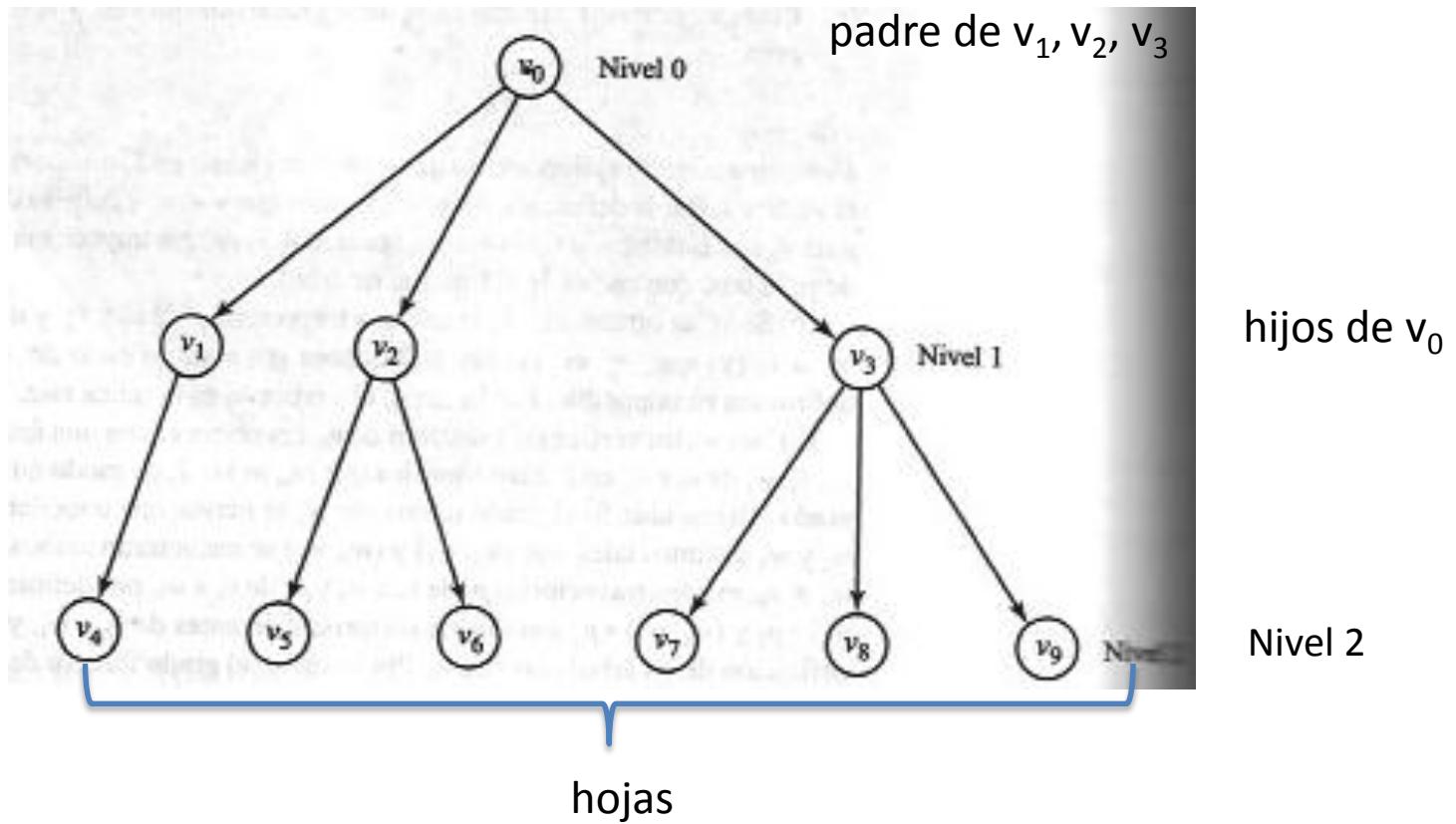
- Arboles con raíz.
- Arboles etiquetados.
- Arboles binarios.
- Búsquedas en árboles.
- Arboles no dirigidos.
- Arboles generadores o de expansión.
- Arbol de expansión mínima.
- Algoritmo de Prim.
- Algoritmo de Kruskal.
- Algoritmo de árboles de deducción de una fórmula de la lógica proposicional.

Sea un conjunto A y T una **relación** definida en A

T es un **árbol** (T, v_0) en A si existe un vértice v_0 en A con la propiedad que:

- 1) Existe una única trayectoria en T de v_0 a cualquier otro vértice v en A
- 2) No existe trayectoria de v_0 a v_0
- 3) v_0 es único y es llamado *raíz del árbol T*
- 4) Si escribimos (T, v_0) , designamos el árbol T con raíz v_0 sobre un conjunto A; un elemento v de A, es un *vértice en T*.





Ancestros

Nodos

Altura

Descendiente

Raíz

Determine en cada caso si R definida en A es un árbol.

$$A = \{ t, u, v, w, x, y, z \}$$

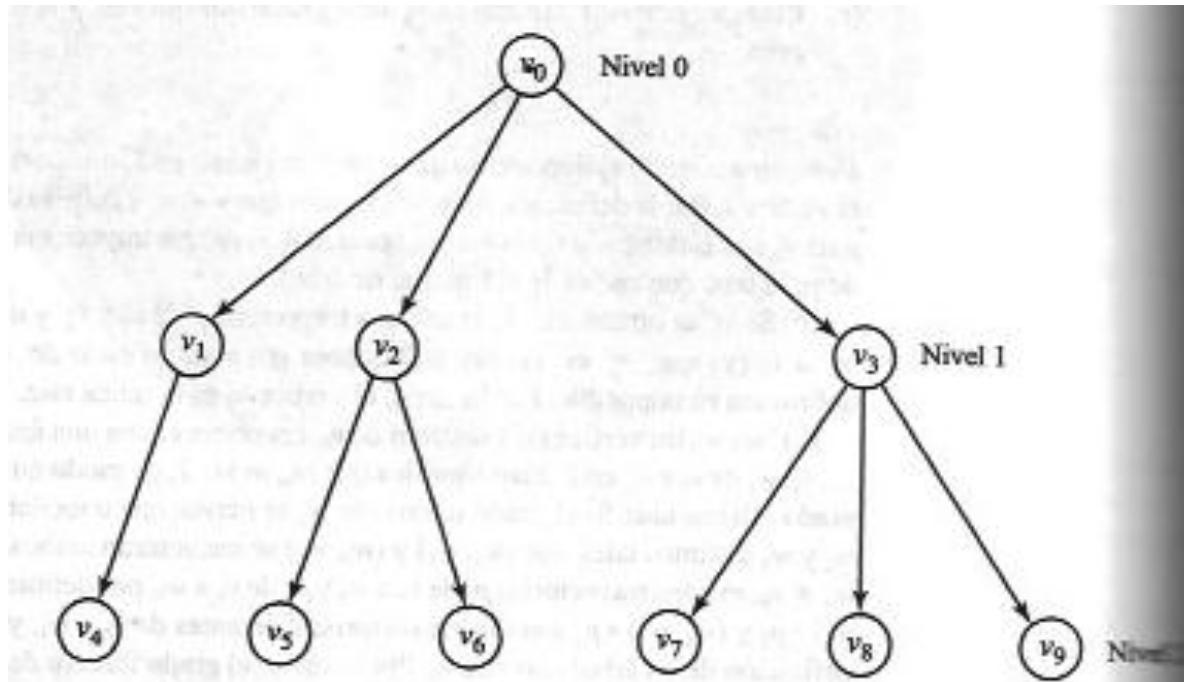
$$R = \{ (t,u); (u,w); (u,x); (u,v); (v,z); (v,y) \}$$

$$A = \{ a, b, c, d, e, f \}$$

$$R = \{ (a,b); (c,e); (f,a); (f,c); (f,d) \}$$

TEROREMA 1: Sea (T, v_0) un árbol con raíz.
Entonces

- (a) No existen ciclos en T .
- (b) v_0 es la única raíz en T .
- (c) Cada vértice en T distinto de v_0 tiene grado interno 1, y v_0 tiene grado interno 0



DEMOSTRACION: TEROREMA 1: Sea (T, v_0) un árbol con raíz. Entonces:

(a) No existen ciclos en T .

Suponga que existe un ciclo q en T que comienza y termina en v

Sabemos por definición que: $v \neq v_0$

Debe existir una trayectoria p de v_0 a v

Entonces:

$q \circ p$ es una trayectoria de v_0 a v diferente de p

Lo que contradice la definición de árbol

Por tanto, NO existen ciclos en T

TEROREMA 1: Sea (T, v_0) un árbol con raíz.

Entonces:

(b) v_0 es la única raíz en T .

Si v_0' es otra raíz de T , existe una trayectoria p que va de v_0 a v_0'
y una trayectoria q que va de v_0' a v_0

Entonces:

$q \circ p$ es un ciclo que va de trayectoria de v_0 a v_0

Lo que contradice la definición de árbol

Por tanto, v_0 es raíz única

TEROREMA 1: Sea (T, v_0) un árbol con raíz.

(c) Cada vértice en T distinto de v_0 tiene grado interno 1, y
 v_0 tiene grado interno 0

Sea w_1 un vértice en T , distinto de v_0

Entonces existe una trayectoria v_0, \dots, v_k, w_1 en T . $(v_k, w_1) \in T$
 w_1 tiene grado interno al menos 1.

Si w_1 tiene grado interno mayor que 1, deben existir vértices w_2 y w_3

$(w_2, w_1) \in T$ $(w_3, w_1) \in T$

$w_2 \neq v_0$ $w_3 \neq v_0$

Existen trayectorias p_2 de v_0 a w_2

p_3 de v_0 a w_3

$$(w_2, w_1) \circ p_2$$

$$(w_3, w_1) \circ p_3$$

Son trayectorias diferentes de v_0 a w_1

Lo que contradice la definición de árbol con raíz en v_0

Por tanto en grado interno de w_1 es uno

Ejercicio: Argumente que v_0 tiene grado cero

TEOREMA2: Sea (T, v_0) un árbol con raíz sobre u conjunto A.

Entonces

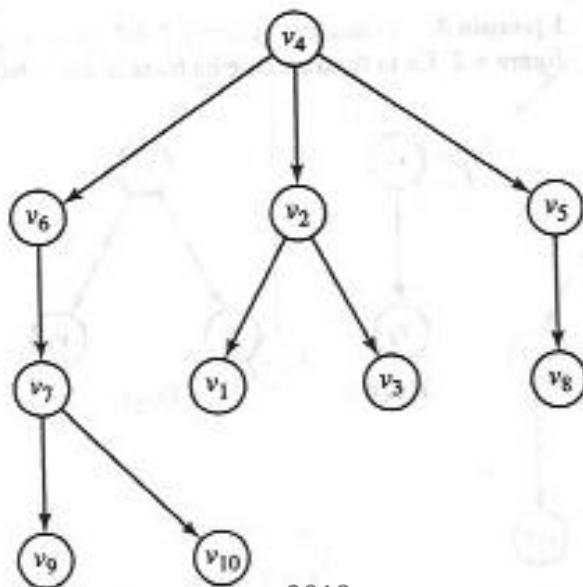
(a) T es Arreflexiva

(b) T es Asimétrica

(c) $T \text{ si } (a T b) \wedge (b T c) \text{ entonces } (a T c)$

Ejemplo: Sean $A = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}\}$

$T = \{(v_2, v_3); (v_2, v_1) (v_4, v_5) (v_4, v_6) (v_5, v_8) (v_6, v_7) (v_4, v_2) (v_7, v_9) (v_7, v_{10})\}$

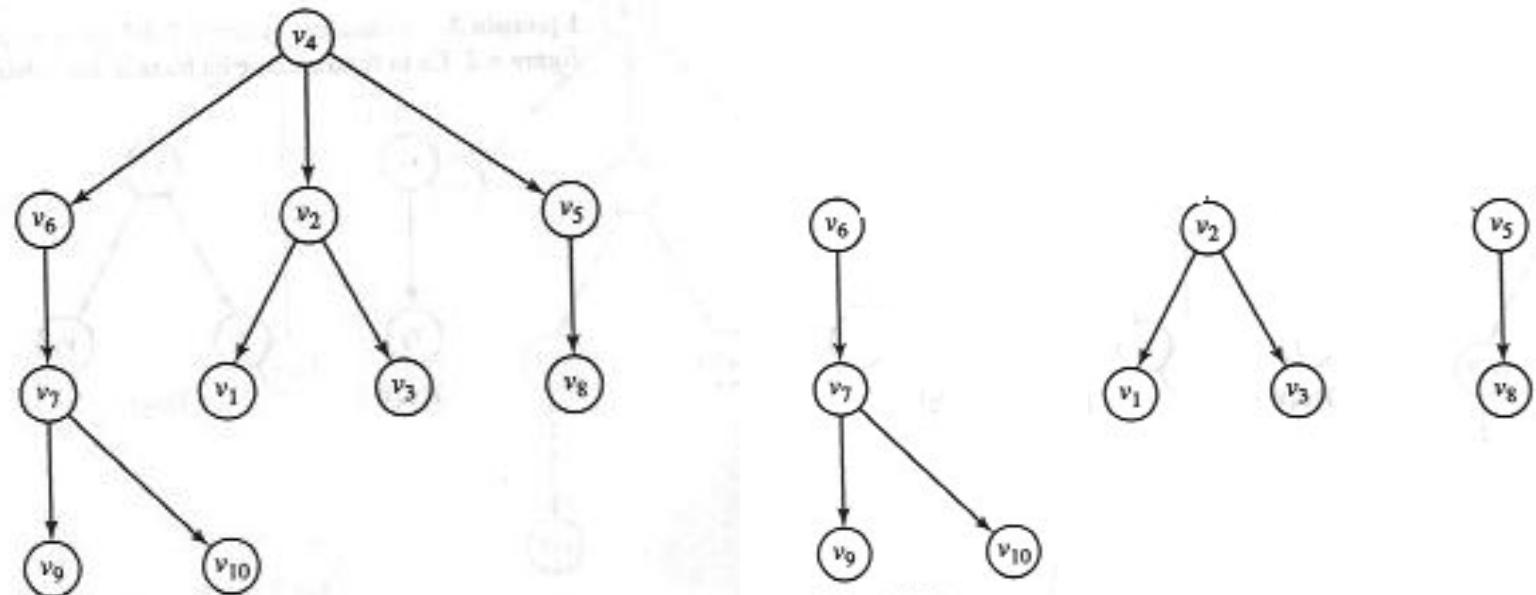


TEOREMA 3: Si (T, v_0) es un árbol con raíz y v pertenece a T , entonces:

$T(v)$ también es un árbol con raíz en v .

$T(v)$ es el subárbol que comienza en v

Ejemplo: El siguiente árbol tiene raíz en v_4 ,
con subárboles $T(v_6)$; $T(v_2)$ y $T(v_5)$



DEMOSTRACION: Si (T, v_0) es un árbol con raíz y v pertenece a T , entonces:

$T(v)$ también es un árbol con raíz en v .

$T(v)$ es el subárbol que comienza en v

Existe una trayectoria de v a cualquier otro vértice en $T(v)$

(por definición)

Si existe un vértice w en $T(v)$ tal que:

existen dos trayectorias distintas q y q' de v a w y

p es la trayectoria en T de v_0 a v

Entonces:

$$q \circ p \quad q' \circ p$$

serían dos trayectorias distintas en T de v_0 a w .

Dos trayectorias distintas en T de v_0 a w es IMPOSIBLE

T es un árbol con raíz en v_0

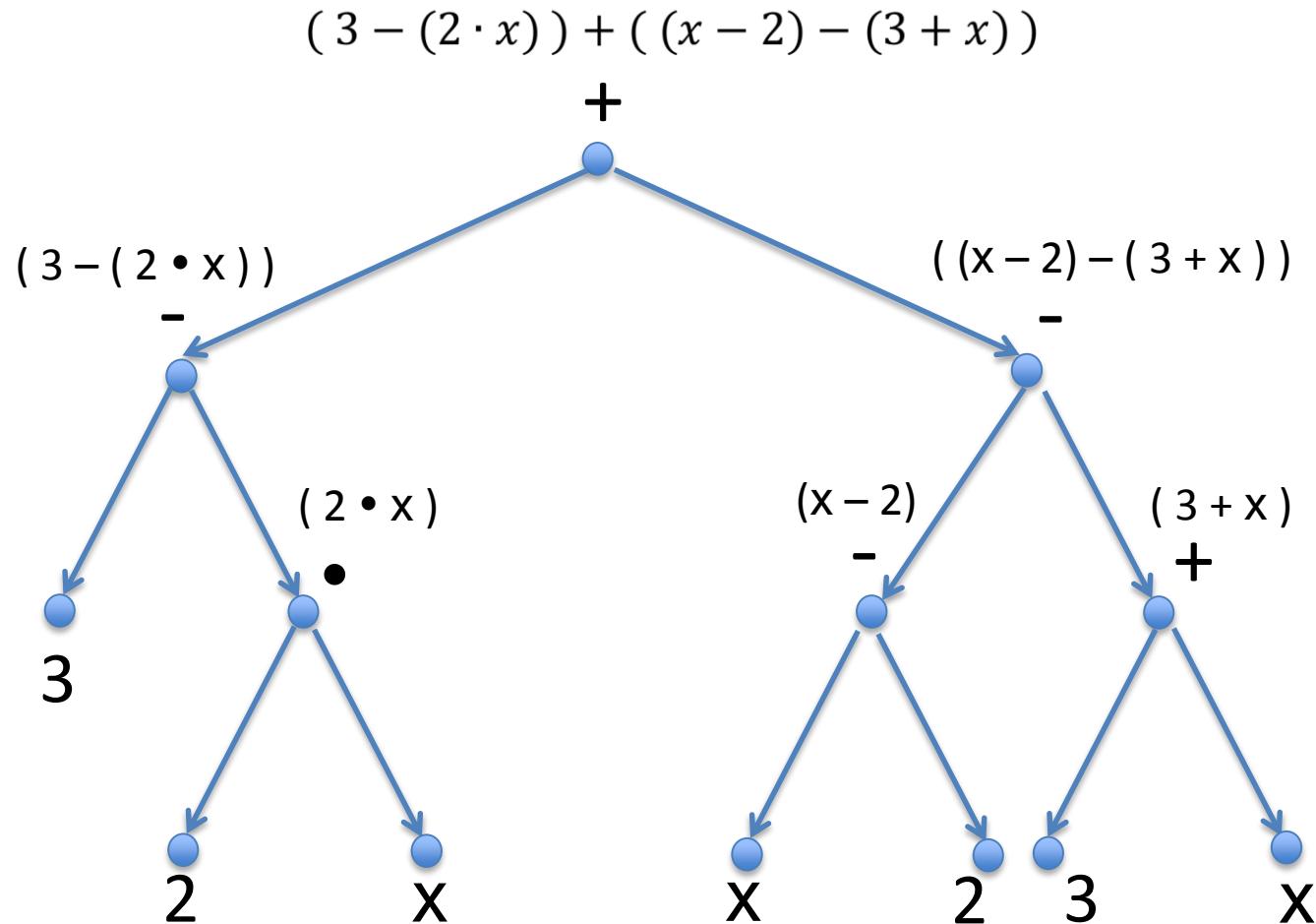
Cada trayectoria desde v a w en $T(v)$ debe ser única

Si q es un ciclo en v en $T(v)$; q es un ciclo también en T

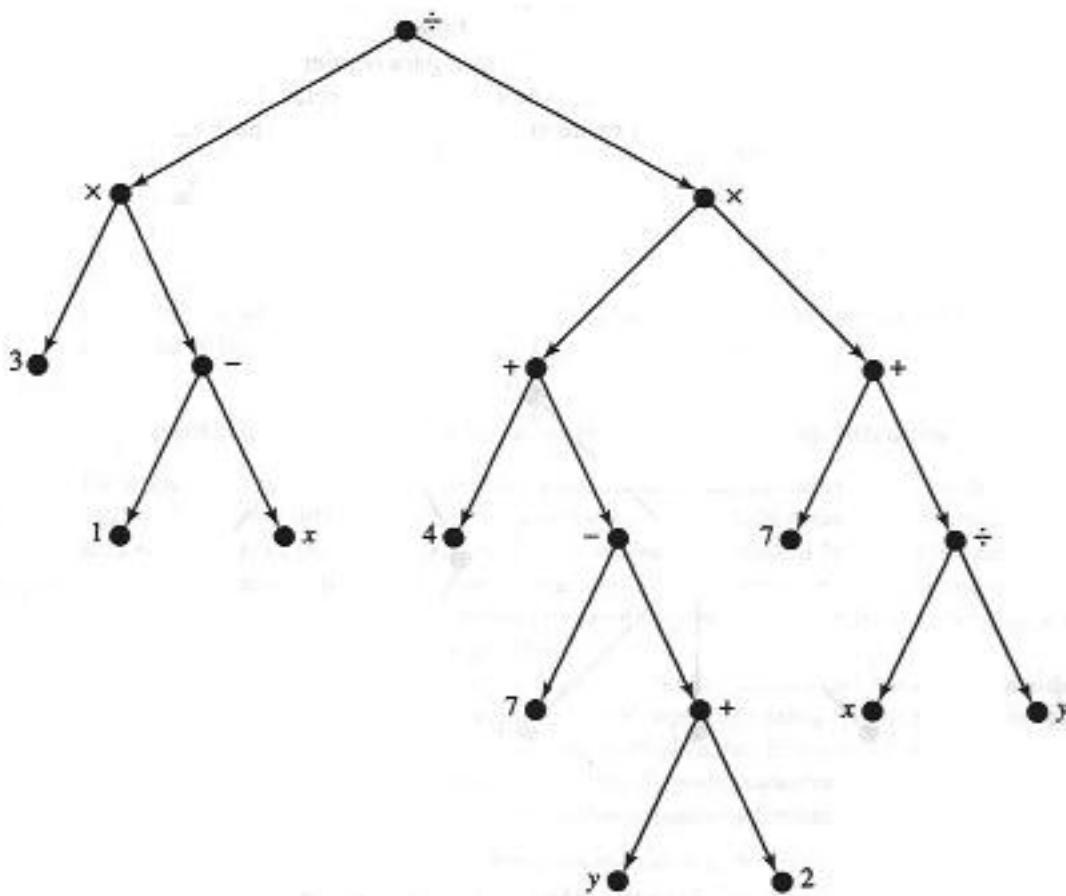
Esto contradice el Teorema 1 (a); por tanto q NO EXISTE

Esto implica que $T(v)$ es un árbol con raíz en v

Arboles etiquetados

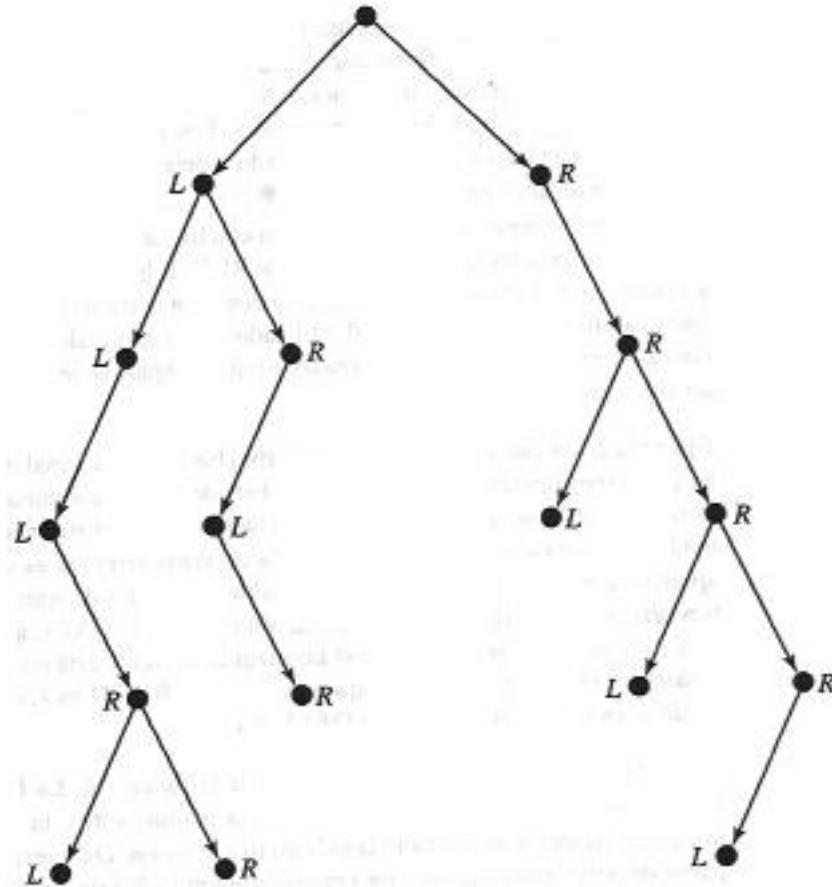


Componga la expresión que se corresponde con el árbol:



$$(3 \times (1 - x)) / ((4 + (7 - (y + 2))) \times (7 + (x / y)))$$

ARBOL BINARIO POSICIONAL



CODIGO ASCII II

Lista completa de caracteres, letras, signos y símbolos del código ASCII :

Caracteres de control ASCII no imprimibles :

codigo ascii 00 = NULL (Carácter nulo)
 codigo ascii 01 = SOH (Inicio de encabezado)
 codigo ascii 02 = STX (Inicio de texto)
 codigo ascii 03 = ETX (Fin de texto, palo corazon barajas inglesas de poker)
 codigo ascii 04 = EOT (Fin de transmisión, palo diamantes barajas de poker)
 codigo ascii 05 = ENQ (Consulta, palo treboles barajas inglesas de poker)
 codigo ascii 06 = ACK (Reconocimiento, palo picas cartas de poker)
 codigo ascii 07 = BEL (Timbre)
 codigo ascii 08 = BS (Retroceso)
 codigo ascii 09 = HT (Tabulador horizontal)
 codigo ascii 10 = LF (Nueva linea - salto de linea)
 codigo ascii 11 = VT (Tabulador vertical)
 codigo ascii 12 = FF (Nueva página - salto de página)
 codigo ascii 13 = CR (ENTER - retorno de carro)
 codigo ascii 14 = SO (Desplazamiento hacia afuera)
 codigo ascii 15 = SI (Desplazamiento hacia adentro)
 codigo ascii 16 = DLE (Escape de vínculo de datos)
 codigo ascii 17 = DC1 (Control dispositivo 1)
 codigo ascii 18 = DC2 (Control dispositivo 2)
 codigo ascii 19 = DC3 (Control dispositivo 3)
 codigo ascii 20 = DC4 (Control dispositivo 4)
 codigo ascii 21 = NAK (Confirmación negativa)
 codigo ascii 22 = SYN (Inactividad síncronica)
 codigo ascii 23 = ETB (Fin del bloque de transmisión)
 codigo ascii 24 = CAN (Cancelar)
 codigo ascii 25 = EM (Fin del medio)
 codigo ascii 26 = SUB (Sustitución)
 codigo ascii 27 = ESC (Esc - escape)
 codigo ascii 28 = FS (Separador de archivos)
 codigo ascii 29 = GS (Separador de grupos)
 codigo ascii 30 = RS (Separador de registros)
 codigo ascii 31 = US (Separador de unidades)
 codigo ascii 127 = DEL (DEL - Suprimir, borrar, eliminar)

Caracteres ASCII alfanumericos imprimibles :

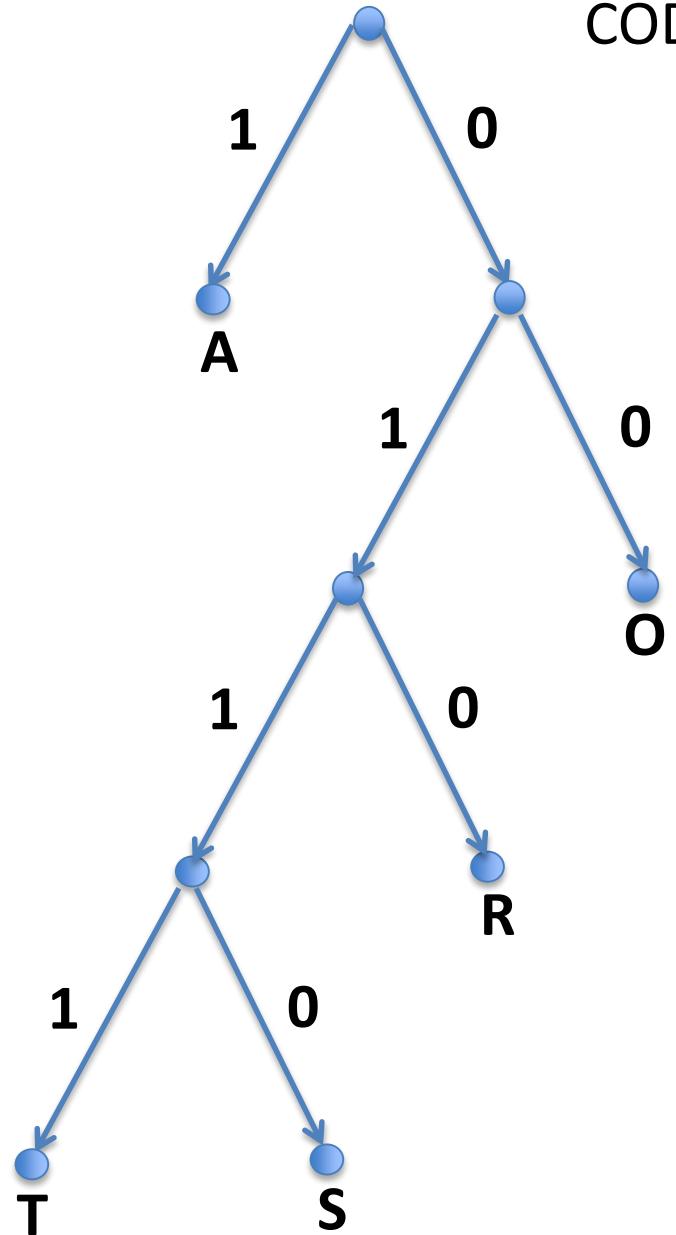
codigo ascii 32 = espacio (Espacio en blanco)
 codigo ascii 1/2 = M (Letra M mayúscula)
 codigo ascii 73 = I (Letra I mayúscula)
 codigo ascii 74 = J (Letra J mayúscula)
 codigo ascii 75 = K (Letra K mayúscula)
 codigo ascii 76 = L (Letra L mayúscula)
 codigo ascii 77 = M (Letra M mayúscula)
 codigo ascii 78 = N (Letra N mayúscula)
 codigo ascii 79 = O (Letra O mayúscula)
 codigo ascii 80 = P (Letra P mayúscula)
 codigo ascii 81 = Q (Letra Q mayúscula)
 codigo ascii 82 = R (Letra R mayúscula)
 codigo ascii 83 = S (Letra S mayúscula)
 codigo ascii 84 = T (Letra T mayúscula)
 codigo ascii 85 = U (Letra U mayúscula)
 codigo ascii 86 = V (Letra V mayúscula)
 codigo ascii 87 = W (Letra W mayúscula)
 codigo ascii 88 = X (Letra X mayúscula)
 codigo ascii 89 = Y (Letra Y mayúscula)
 codigo ascii 90 = Z (Letra Z mayúscula)
 codigo ascii 91 = [(Abre corchetes)
 codigo ascii 92 = \ (Barra invertida , contrabarra , barra inversa)
 codigo ascii 93 =] (Cierra corchetes)
 codigo ascii 94 = ^ (Intercalación - acento circunflejo)
 codigo ascii 95 = _ (Guion bajo , subrayado , subguion)
 codigo ascii 96 = ` (Acento grave)
 codigo ascii 97 = a (Letra a minúscula)
 codigo ascii 98 = b (Letra b minúscula)
 codigo ascii 99 = c (Letra c minúscula)
 codigo ascii 100 = d (Letra d minúscula)
 codigo ascii 101 = e (Letra e minúscula)
 codigo ascii 102 = f (Letra f minúscula)
 codigo ascii 103 = g (Letra g minúscula)

codigo ascii 128 = ç (Letra C cedilla mayúscula)
 codigo ascii 129 = ù (Letra e minúscula con diéresis)
 codigo ascii 130 = é (Letra e minúscula con acento agudo)
 codigo ascii 131 = à (Letra e minúscula con acento circunflejo)
 codigo ascii 132 = á (Letra e minúscula con diéresis)
 codigo ascii 133 = à (Letra e minúscula con acento grave)
 codigo ascii 134 = â (Letra e minúscula con anillo)
 codigo ascii 135 = ç (Letra c cedilla minúscula)
 codigo ascii 136 = ê (Letra e minúscula con acento circunflejo)
 codigo ascii 137 = ê (Letra e minúscula con diéresis)
 codigo ascii 138 = ô (Letra e minúscula con acento grave)
 codigo ascii 139 = ï (Letra i minúscula con diéresis)
 codigo ascii 140 = î (Letra i minúscula con acento circunflejo)
 codigo ascii 141 = ï (Letra i minúscula con acento grave)
 codigo ascii 142 = Å (Letra A mayúscula con diéresis)
 codigo ascii 143 = Ä (Letra A mayúscula con anillo)
 codigo ascii 144 = È (Letra E mayúscula con acento agudo)
 codigo ascii 145 = æ (Diptongo latino ae minúscula)
 codigo ascii 146 = Æ (Diptongo latino AE mayúscula)
 codigo ascii 147 = ö (Letra o minúscula con acento circunflejo)
 codigo ascii 148 = ö (Letra o minúscula con diéresis)
 codigo ascii 149 = ò (Letra o minúscula con acento grave)
 codigo ascii 150 = û (Letra u minúscula con acento circunflejo)
 codigo ascii 151 = ù (Letra u minúscula con acento grave)
 codigo ascii 152 = ý (Letra y minúscula con diéresis)
 codigo ascii 153 = Ö (Letra O mayúscula con diéresis)
 codigo ascii 154 = Ü (Letra U mayúscula con diéresis)
 codigo ascii 155 = ø (Letra o minúscula con barra inclinada)
 codigo ascii 156 = £ (Signo Libra Esterlina)
 codigo ascii 157 = Ø (Letra O mayúscula con barra inclinada)
 codigo ascii 158 = ✖ (Signo de multiplicación)
 codigo ascii 159 = f (Simbolo de función, florín neerlandés)
 codigo ascii 160 = á (Letra a minúscula con acento agudo)
 codigo ascii 161 = í (Letra i minúscula con acento agudo)
 codigo ascii 162 = ó (Letra o minúscula con acento agudo)
 codigo ascii 163 = ú (Letra u minúscula con acento agudo)
 codigo ascii 164 = ñ (Letra n con acento circunflejo , recuadro)
 codigo ascii 165 = ñ (Letra n con acento circunflejo , empalme arriba , recuadro)
 codigo ascii 166 = ñ (Letra n con acento circunflejo , empalme abajo , recuadro)
 codigo ascii 167 = ñ (Letra n con acento circunflejo , empalme izquierdo , recuadro)
 codigo ascii 168 = ñ (Letra n con acento circunflejo , empalme media , recuadro)
 codigo ascii 169 = ñ (Letra n con acento circunflejo , empalme alta , recuadro)
 codigo ascii 170 = ~ (Signo de negación)
 codigo ascii 171 = ½ (Un medio, mitad, fracción)
 codigo ascii 172 = ¼ (Un cuarto, cuarta parte, fracción)
 codigo ascii 173 = ⅓ (Abre signo de exclamación, signo de admiración)
 codigo ascii 174 = ⅔ (Cierra comillas bajas, angulares, latinas o españolas)
 codigo ascii 175 = ⅕ (Cierra comillas bajas, angulares, latinas o españolas)
 codigo ascii 176 = ☒ (Bloque color tramado densidad baja, carácter gráfico)
 codigo ascii 177 = ☓ (Bloque color tramado densidad media, gráfico)
 codigo ascii 178 = ☔ (Bloque color tramado densidad alta, carácter gráfico)
 codigo ascii 179 = ☑ (Línea simple vertical de recuadro gráfico)
 codigo ascii 180 = ☒ (Línea vertical con empalme de recuadro gráfico)
 codigo ascii 181 = ☐ (Letra A mayúscula con acento agudo)
 codigo ascii 182 = ☑ (Letra A mayúscula con acento circunflejo)
 codigo ascii 183 = ☒ (Letra A mayúscula con acento grave)
 codigo ascii 184 = © (Simbolo Copyright, bajo derecho de autor)
 codigo ascii 185 = ☓ (Doble línea vertical empalme izquierdo, gráfico)
 codigo ascii 186 = ☔ (Líneas doble vertical de recuadro gráfico, verticales)
 codigo ascii 187 = ☑ (Línea doble esquina superior derecha de recuadro)
 codigo ascii 188 = ☒ (Línea doble esquina inferior derecha de recuadro)
 codigo ascii 189 = ☒ (Signo centavo, céntimo o centésimo)
 codigo ascii 190 = ¥ (Signo monetario YEN japonés, YUAN chino)
 codigo ascii 191 = ☑ (Línea simple esquina de recuadro gráfico)
 codigo ascii 192 = ☒ (Línea simple esquina de recuadro gráfico)
 codigo ascii 193 = ☑ (Línea horizontal con empalme de recuadro gráfico)
 codigo ascii 194 = ☒ (Línea horizontal con empalme de recuadro gráfico)
 codigo ascii 195 = ☑ (Línea vertical con empalme de recuadro gráfico)
 codigo ascii 196 = ☑ (Línea simple horizontal de recuadro gráfico)
 codigo ascii 197 = ☒ (Líneas simples empalmes de recuadro gráfico)
 codigo ascii 198 = ☐ (Letra A minúscula con tilde)
 codigo ascii 199 = ☑ (Letra A mayúscula con tilde)
 codigo ascii 200 = ☒ (Línea doble esquina inferior izquierda de recuadro)
 codigo ascii 201 = ☑ (Línea doble esquina superior izquierda de recuadro)
 codigo ascii 202 = ☒ (Doble línea horizontal empalme arriba, recuadro)
 codigo ascii 203 = ☑ (Doble línea horizontal empalme abajo, recuadro)

codigo ascii 103 = g (Letra g minúscula)
 codigo ascii 104 = h (Letra h minúscula)
 codigo ascii 105 = i (Letra i minúscula)
 codigo ascii 106 = j (Letra j minúscula)
 codigo ascii 107 = k (Letra k minúscula)
 codigo ascii 108 = l (Letra l minúscula)
 codigo ascii 109 = m (Letra m minúscula)
 codigo ascii 110 = n (Letra n minúscula)
 codigo ascii 111 = o (Letra o minúscula)
 codigo ascii 112 = p (Letra p minúscula)
 codigo ascii 113 = q (Letra q minúscula)
 codigo ascii 114 = r (Letra r minúscula)
 codigo ascii 115 = s (Letra s minúscula)
 codigo ascii 116 = t (Letra t minúscula)
 codigo ascii 117 = u (Letra u minúscula)
 codigo ascii 118 = v (Letra v minúscula)
 codigo ascii 119 = w (Letra w minúscula)
 codigo ascii 120 = x (Letra x minúscula)
 codigo ascii 121 = y (Letra y minúscula)
 codigo ascii 122 = z (Letra z minúscula)
 codigo ascii 123 = { (Abre llave curva - llaves curvas)
 codigo ascii 124 = } (Cierra llave - llaves curvas)
 codigo ascii 125 = ~ (Signo de equivalencia , tilde o virgullilla de la ñ)

2018

CODIGO DE HUFFMAN

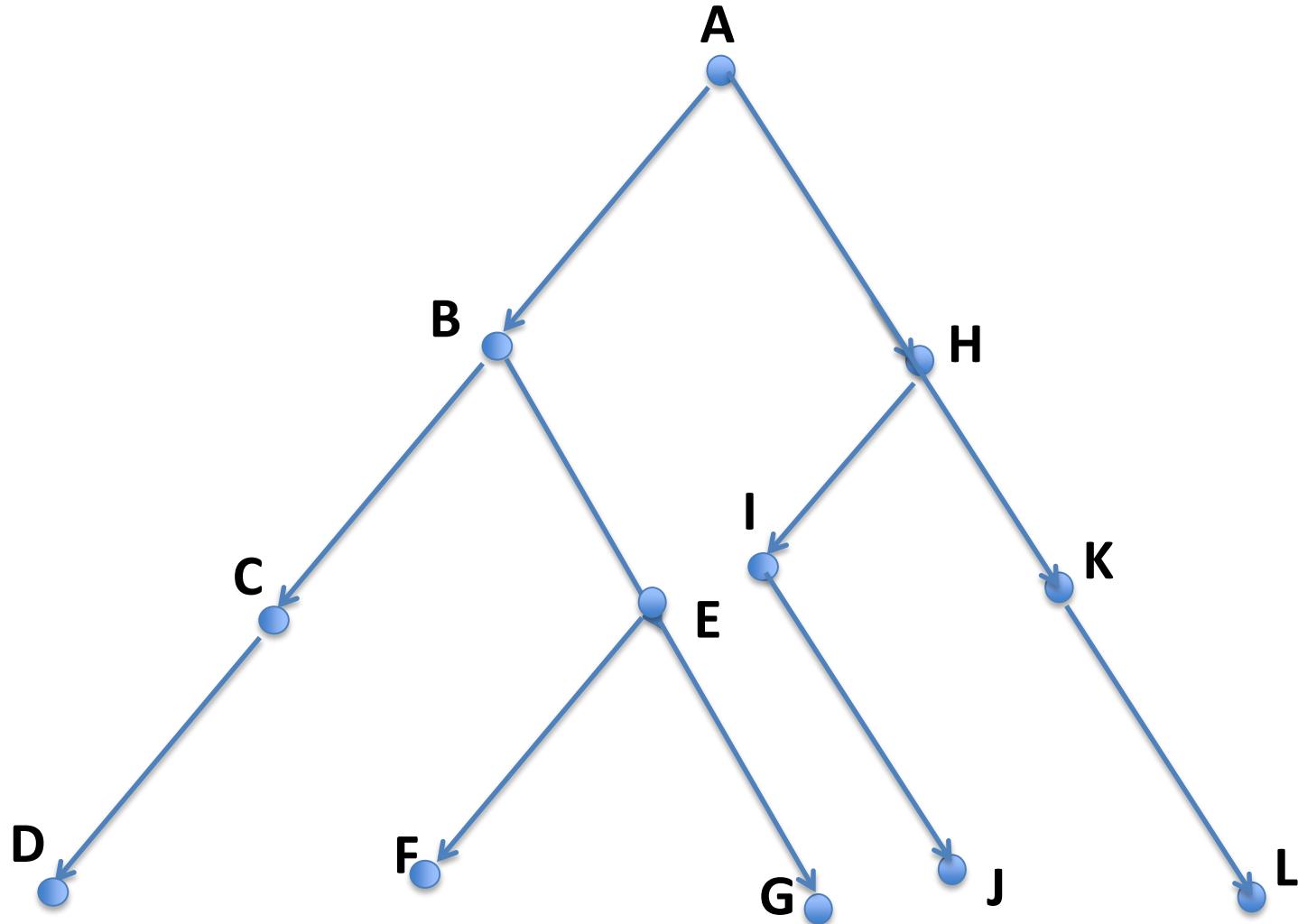


Carácter	Código	ASCII
A	100	0001
B	100	0010
C	100	0011
1	011	0001
2	011	0010
!	010	0001
*	010	1010

RAT 0 1 0 1 0 1 1 1

RATO 0 1 0 1 0 1 1 1 0 0

BUSQUEDA EN ARBOLES



Algoritmo PREORDEN

PASO 1: Visite v

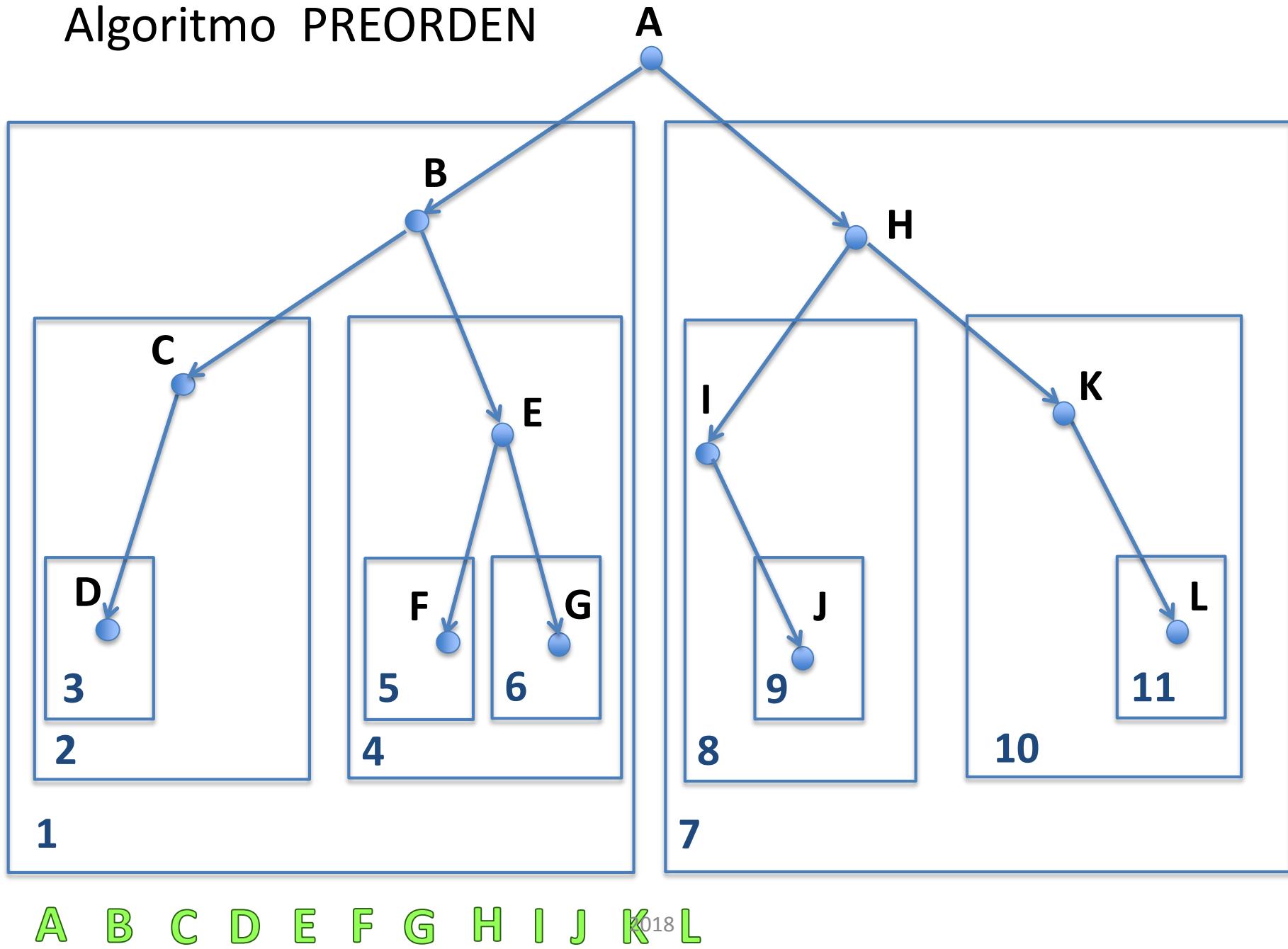
PASO 2: Si existe v_L , entonces aplique este algoritmo
a $(T(v_L), v_L)$

PASO 3: Si existe v_R , entonces aplique este algoritmo
a $(T(v_R), v_R)$

Fin del algoritmo

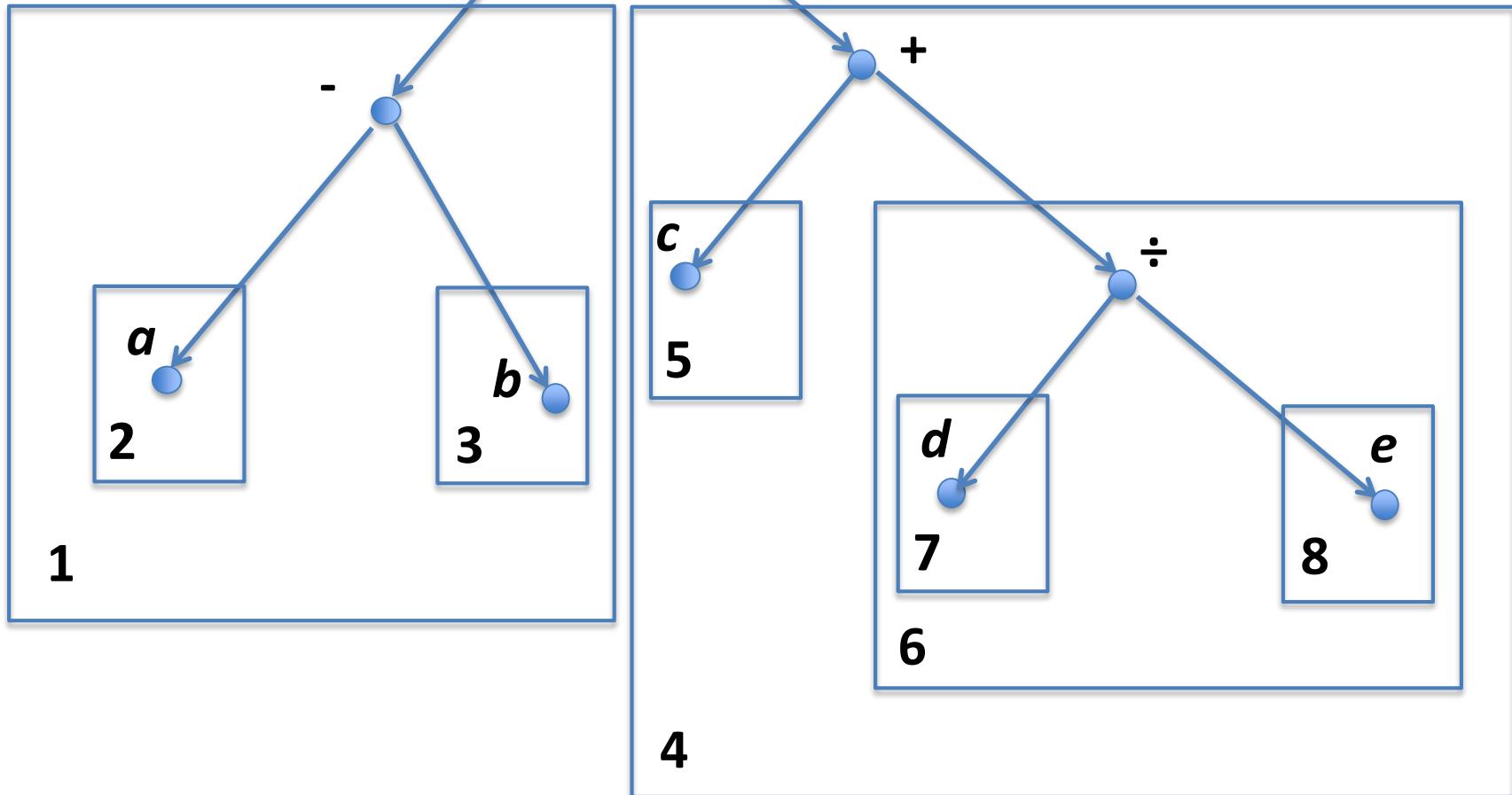
1. Visite la raíz.
2. Busque en el sub árbol izquierdo, si existe
3. Busque en el sub árbol derecho, si existe

Algoritmo PREORDEN



ALGORITMO PREORDEN

$$(a - b) \times (c + (d / e))$$



X - **a** **b** + **c** \div **d** **e**

Escritura polaca
2018

ALGORITMO PREORDEN

$$(a - b) \times (c + (d / e))$$

$$a = 6$$

$$b = 4$$

x - a b + c / d e

$$c = 5$$

$$d = 2$$

x - 6 4 + 5 / 2 2

$$e = 2$$

x - 6 4 + 5 1

x 2 6

12

Algoritmo ENTREORDEN

PASO 1: Si existe v_L , entonces aplique este algoritmo
a $(T(v_L), v_L)$

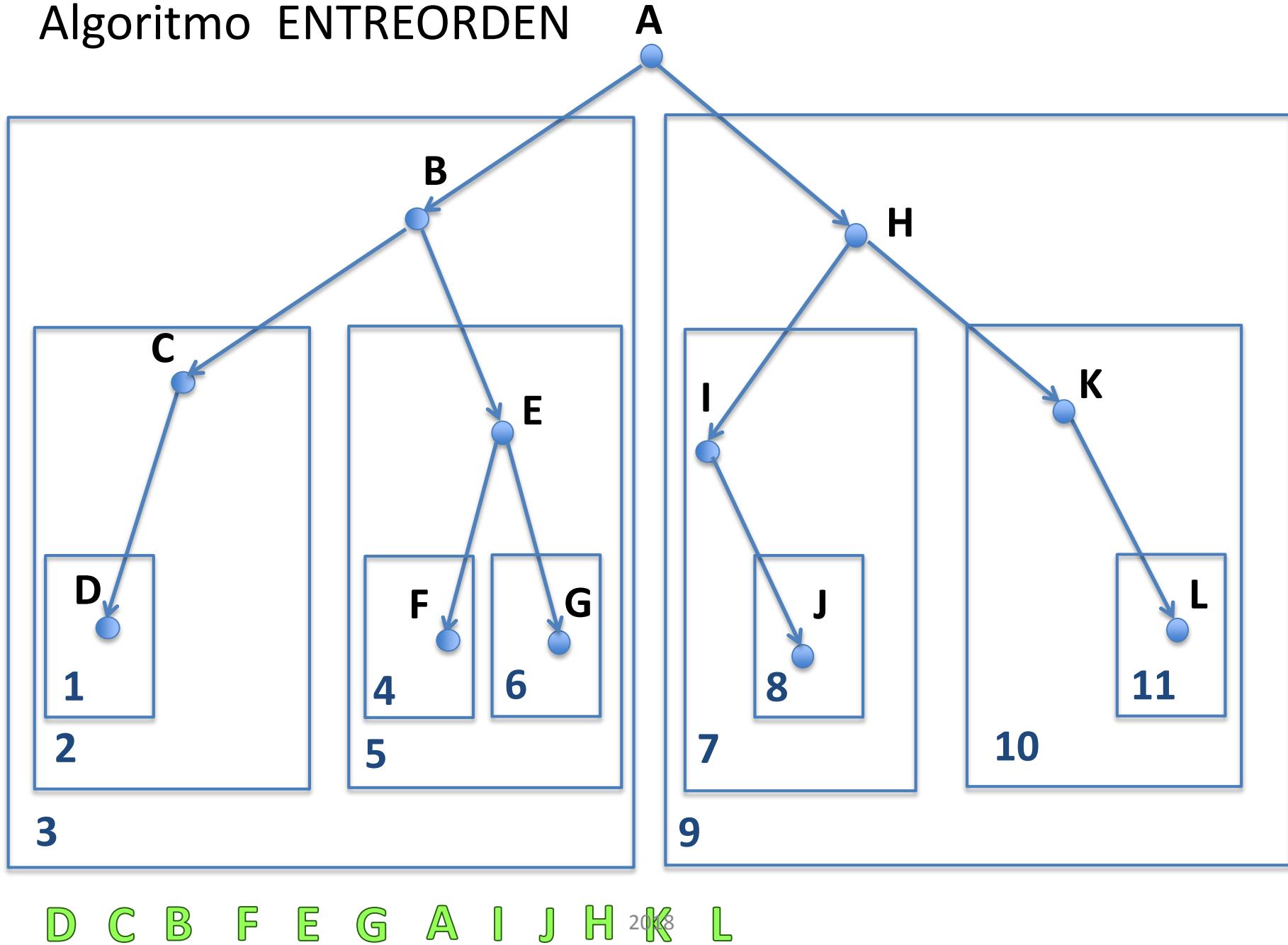
PASO 2: Visite v

PASO 3: Si existe v_R , entonces aplique este algoritmo
a $(T(v_R), v_R)$

Fin del algoritmo

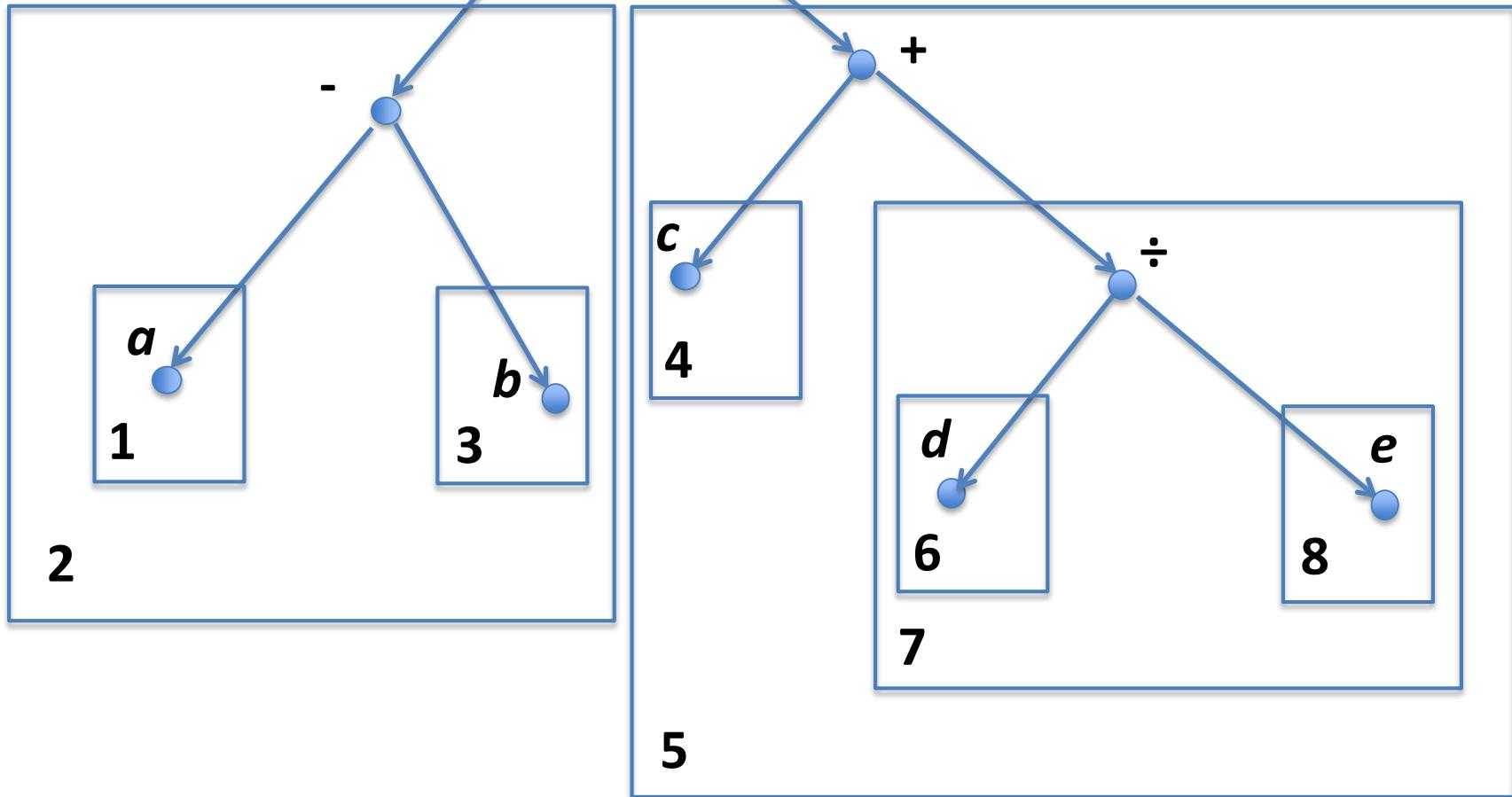
1. Busque en el sub árbol izquierdo, si existe
2. Visite la raíz.
3. Busque en el sub árbol derecho, si existe

Algoritmo ENTREORDEN



ALGORITMO ENTREORDEN

$$(a - b) \times (c + (d / e))$$



a - b x c + d ÷ e

ALGORITMO ENTREORDEN

$$(a - b) \times (c + (d / e))$$

$$a = 6$$

$$b = 4$$

$$a - b \times c + d / e$$

$$c = 5$$

$$d = 2$$

$$e = 2$$

$$6 - 4 \times 5 + 2 / 2$$

$$6 - 4 \times 5 + 2 / 2$$

$$2 \times 5 + 1$$

$$6 - 20 + 1$$

$$2 \times 6$$

$$- 14 + 1 \\ - 13$$

12

ambiguo

Algoritmo POSORDEN

PASO 1: Si existe v_L , entonces aplique este algoritmo
a $(T(v_L), v_L)$

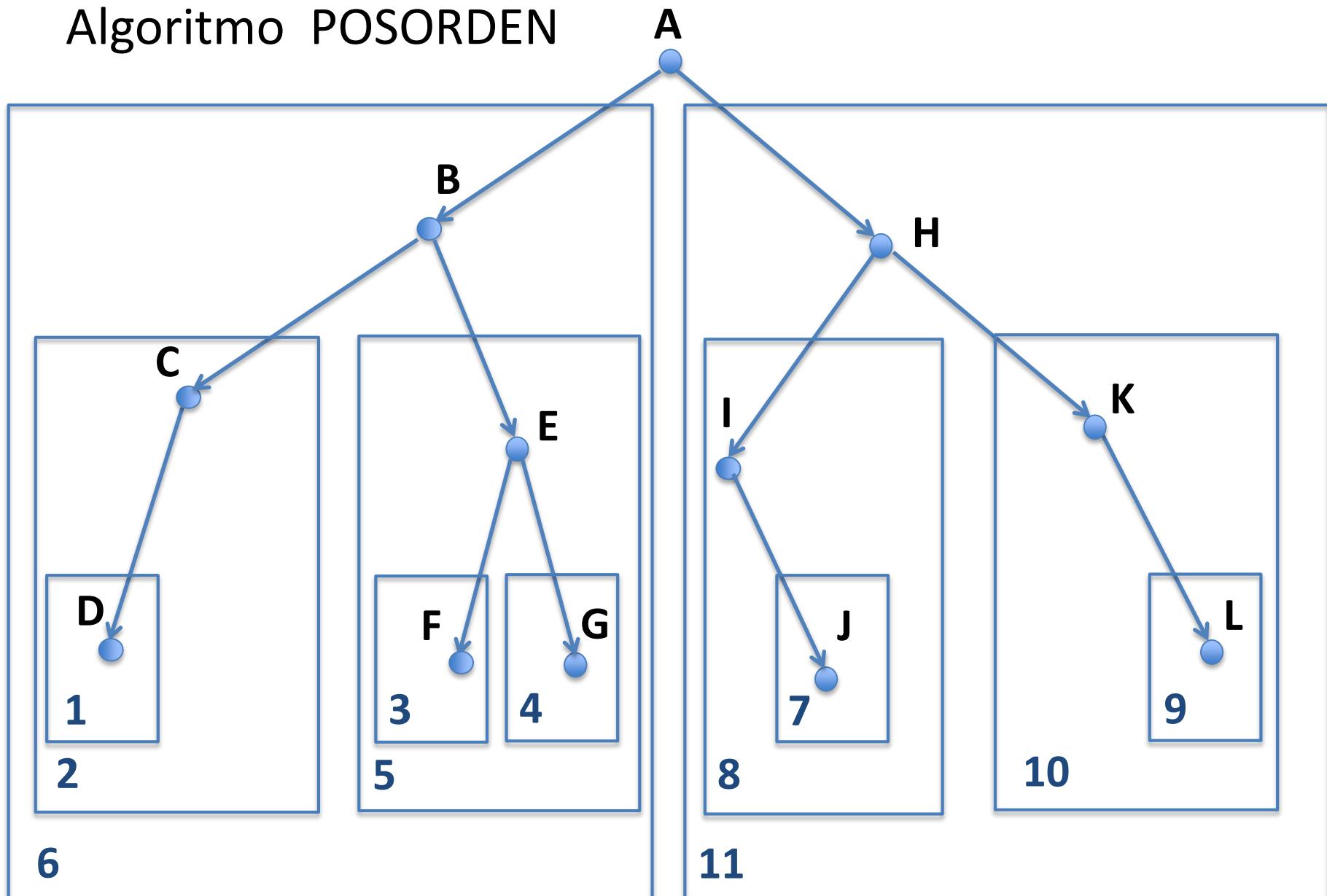
PASO 2: Si existe v_R , entonces aplique este algoritmo
a $(T(v_R), v_R)$

PASO 3: Visite v

Fin del algoritmo

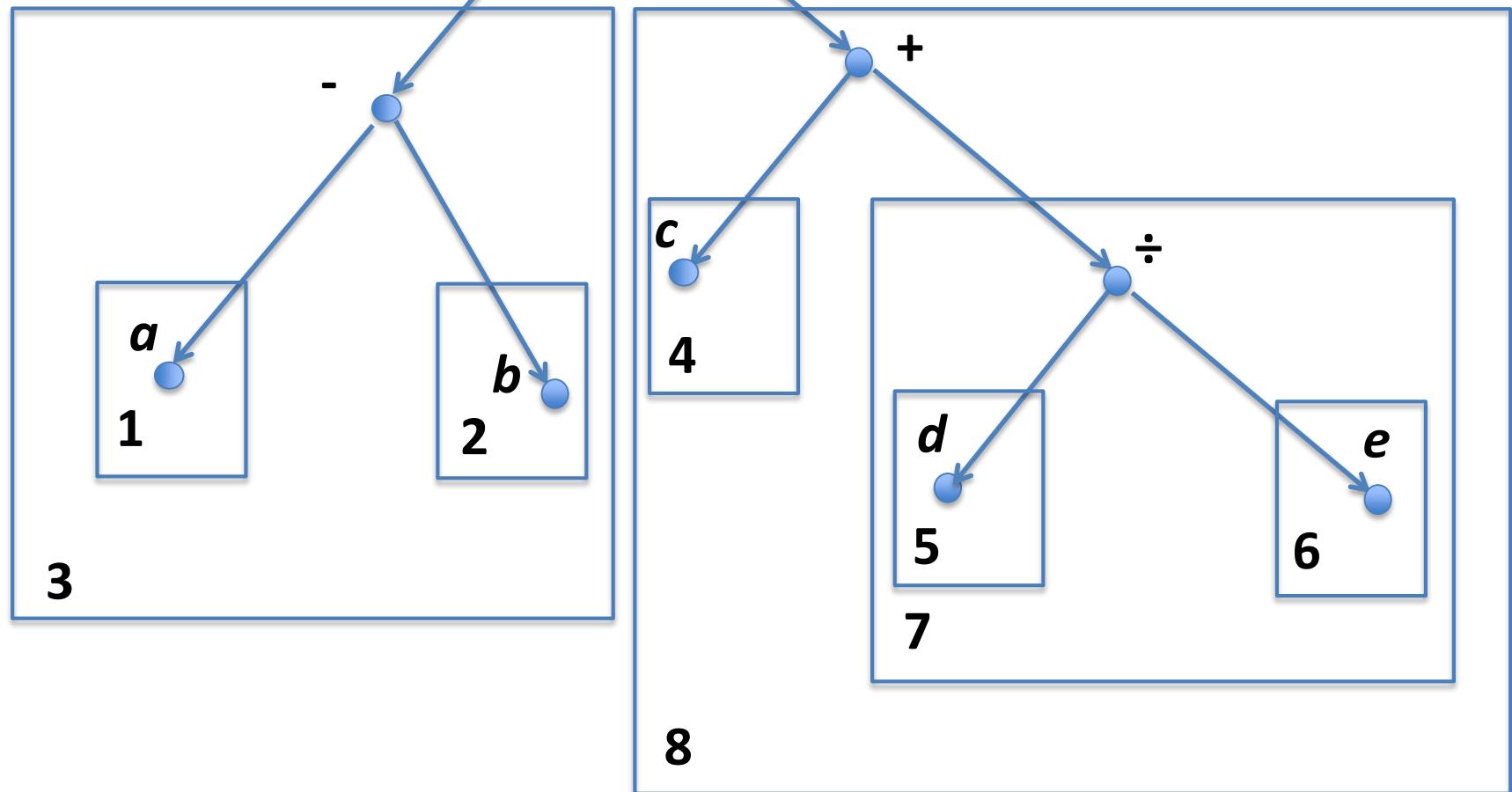
1. Busque en el sub árbol izquierdo, si existe
2. Busque en el sub árbol derecho, si existe
3. Visite la raíz.

Algoritmo POSORDEN



D C F G E B J I L K H A

$$(a - b) \times (c + (d / e))$$



a b - c d e ÷ + x

Escritura polaca inversa
2018

Algoritmo POSORDEN

$$(a - b) \times (c + (d / e))$$

$$a = 6$$

$$b = 4$$

a b - c d e / + x

$$c = 5$$

$$d = 2$$

6 4 - 5 2 2 / + x

$$e = 2$$

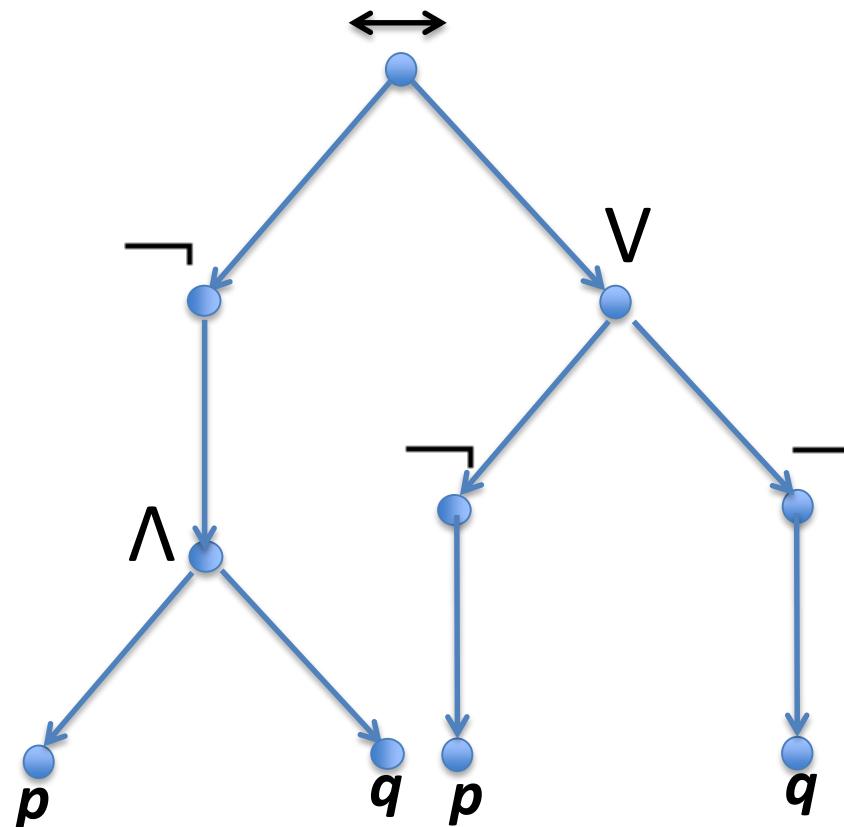
2 5 1 + x

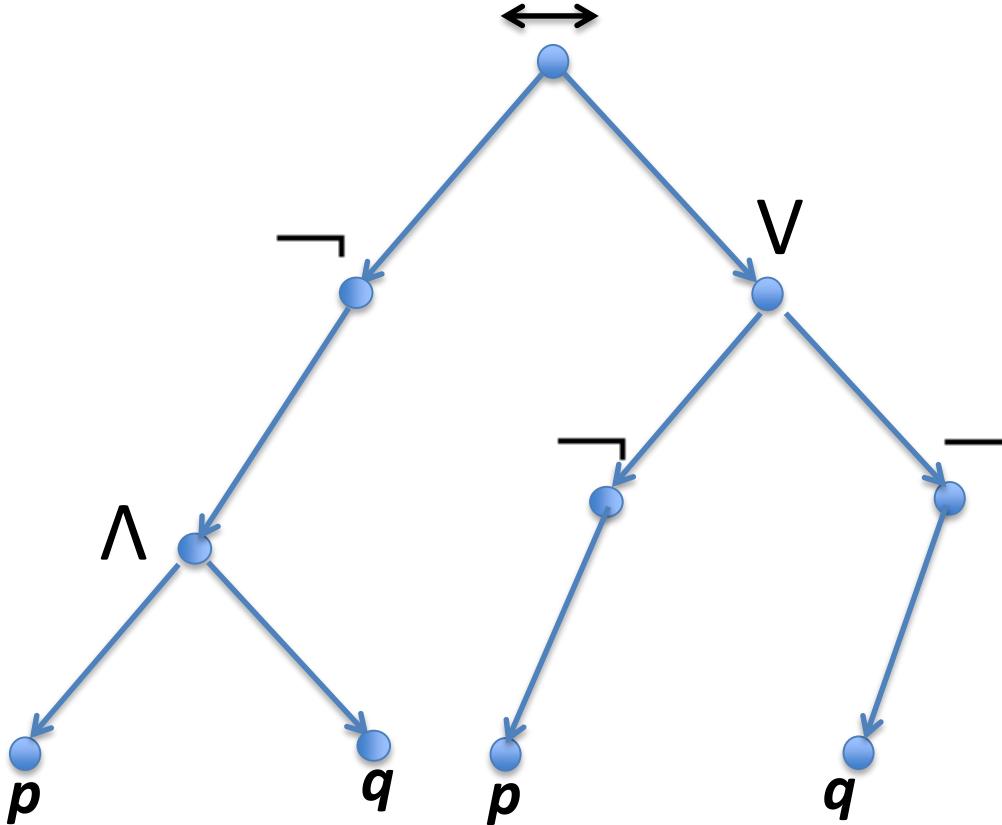
2 6 x

12

Ejercicios: Construya un árbol para la siguiente operación lógica e implemente su recorrido en los tres sentidos

$$(\neg(p \wedge q)) \leftrightarrow (\neg p \vee \neg q)$$





$\leftrightarrow \perp \wedge p\,q \vee \perp p \perp q$

PREORDEN

$p\,q \wedge \perp p \perp q \vee \leftrightarrow$

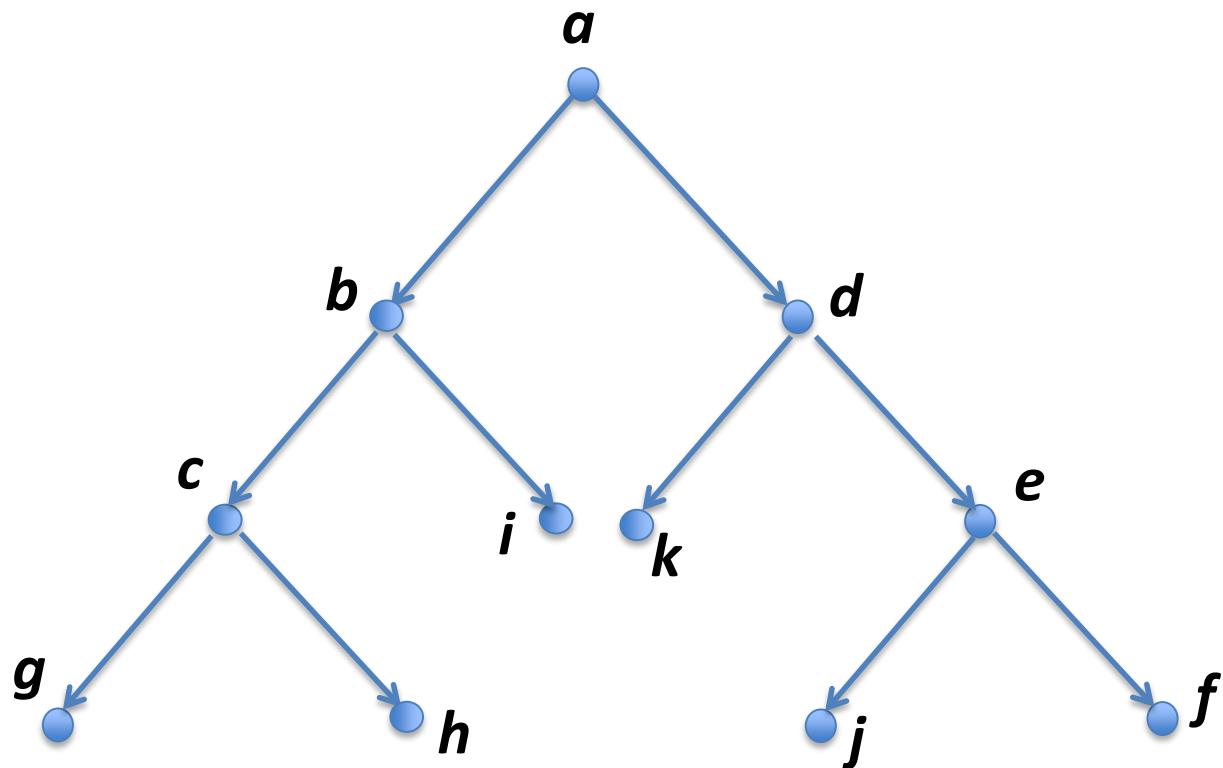
POSORDEN

$\perp p \wedge q \leftrightarrow \perp p \vee \perp q$

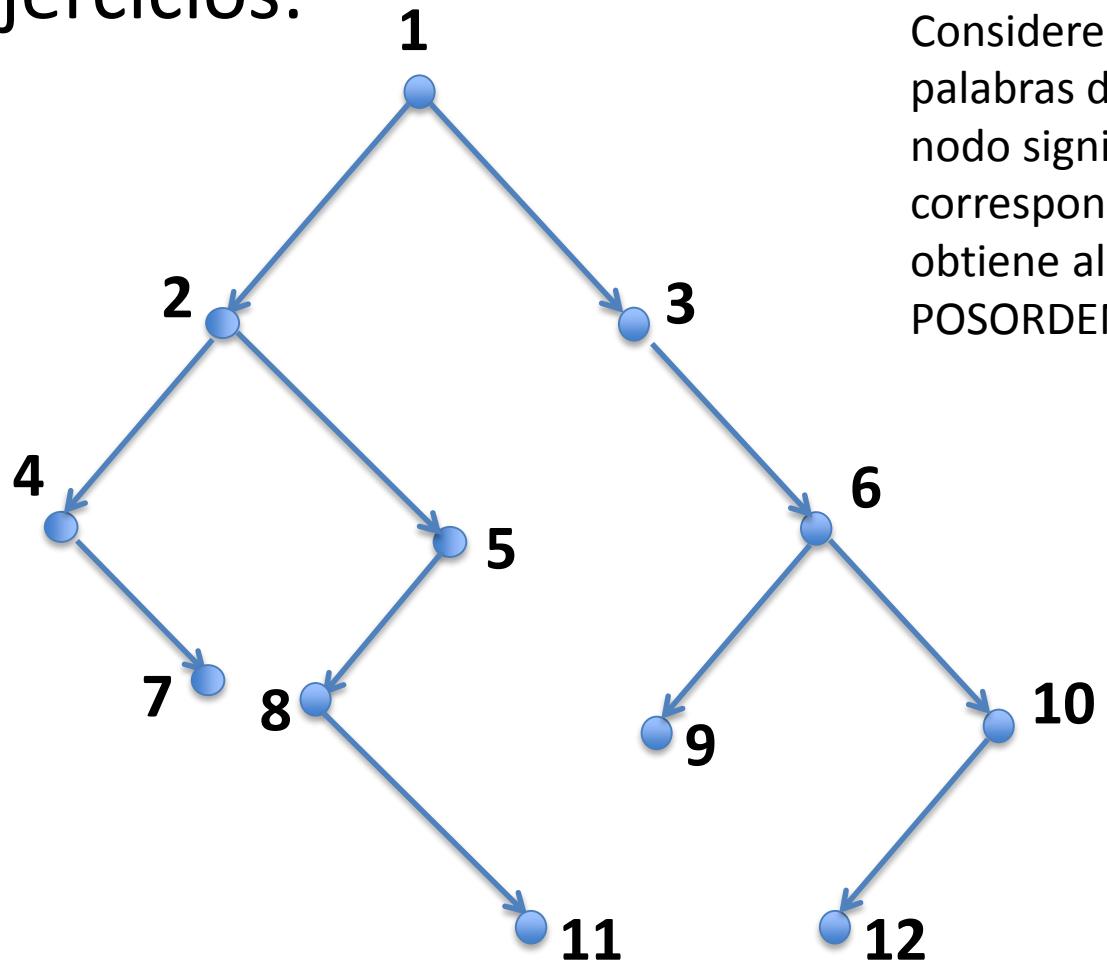
ENTREORDEN (ambigua)

Ejercicios:

Efectúe búsqueda PREORDEN, ENTREORDEN Y POSORDEN en el árbol presentado



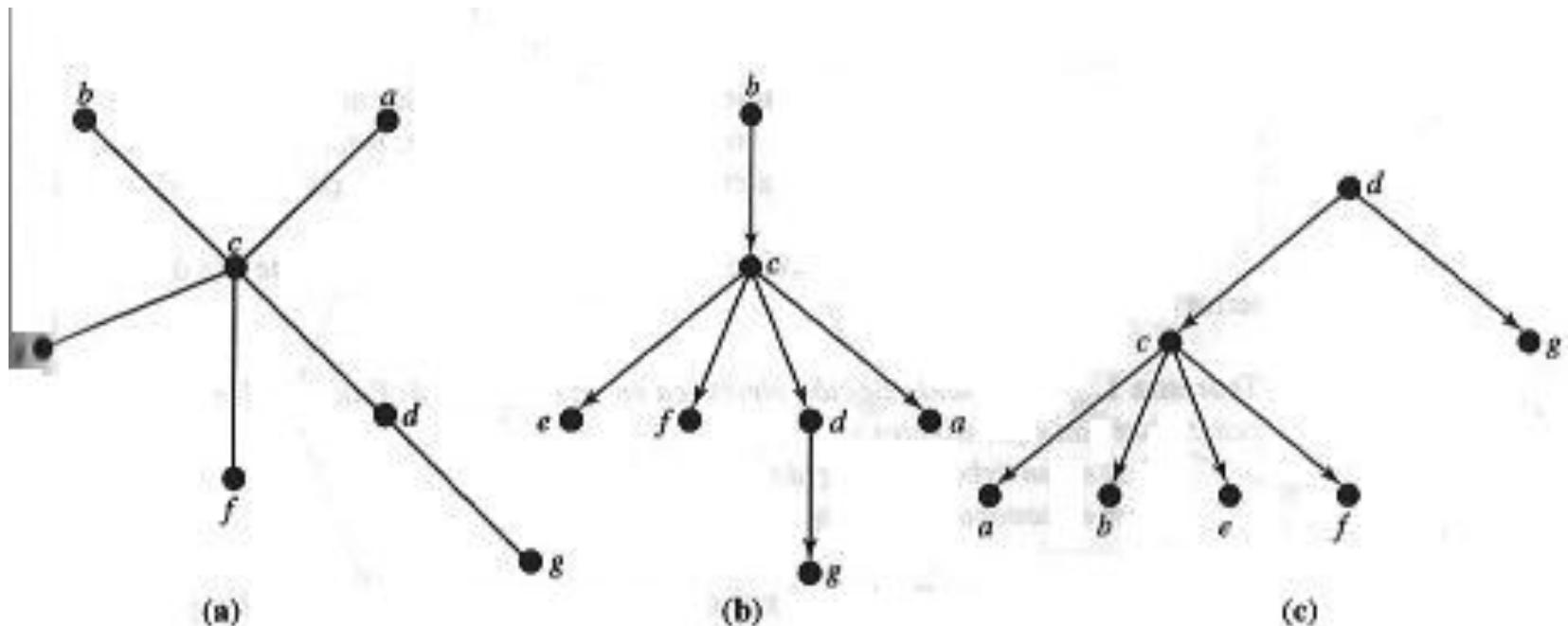
Ejercicios:

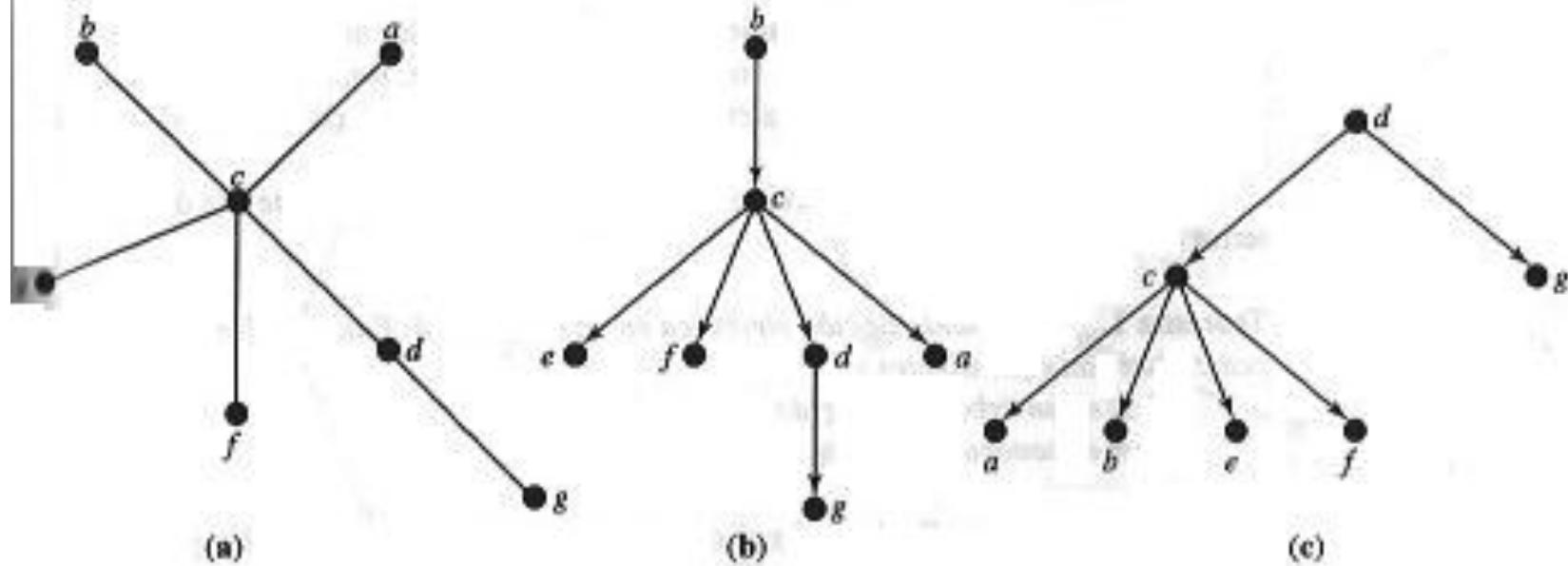


Considere el árbol de la figura y la lista de palabras dadas. Suponga que la visita a un nodo significa imprimir la palabra correspondiente. Imprima la frase que se obtiene al realizar una búsqueda en POSORDEN en el árbol

- | | |
|------------|-----------|
| 1. UNA | 7. YO |
| 2. PURPURA | 8. UNA |
| 3. VEA | 9. ESPERO |
| 4. NUNCA | 10. YO |
| 5. VACA | 11. VI |
| 6. NUNCA | 12. QUE |

ARBOLES NO DIRIGIDOS



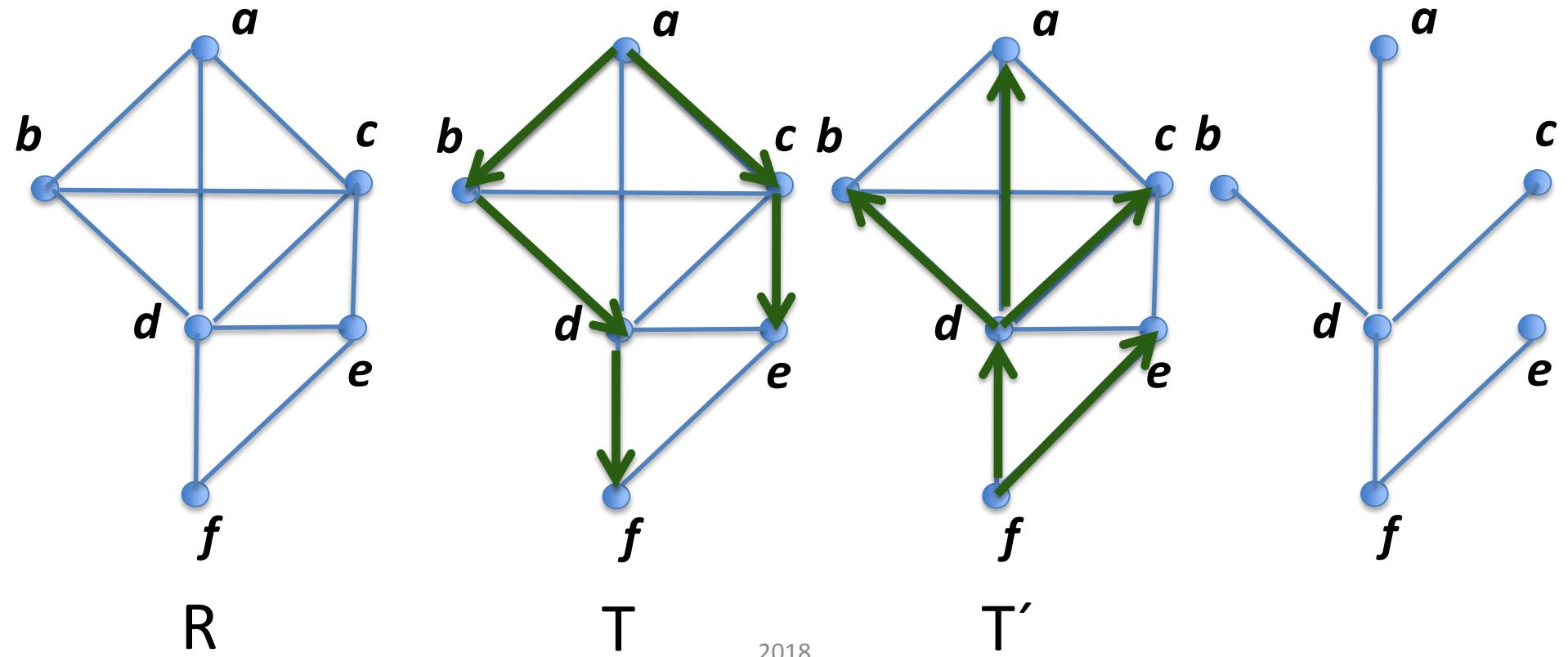


TEOREMA 1: Sea R una relación simétrica en un conjunto A. Entonces las siguientes proposiciones son equivalentes:

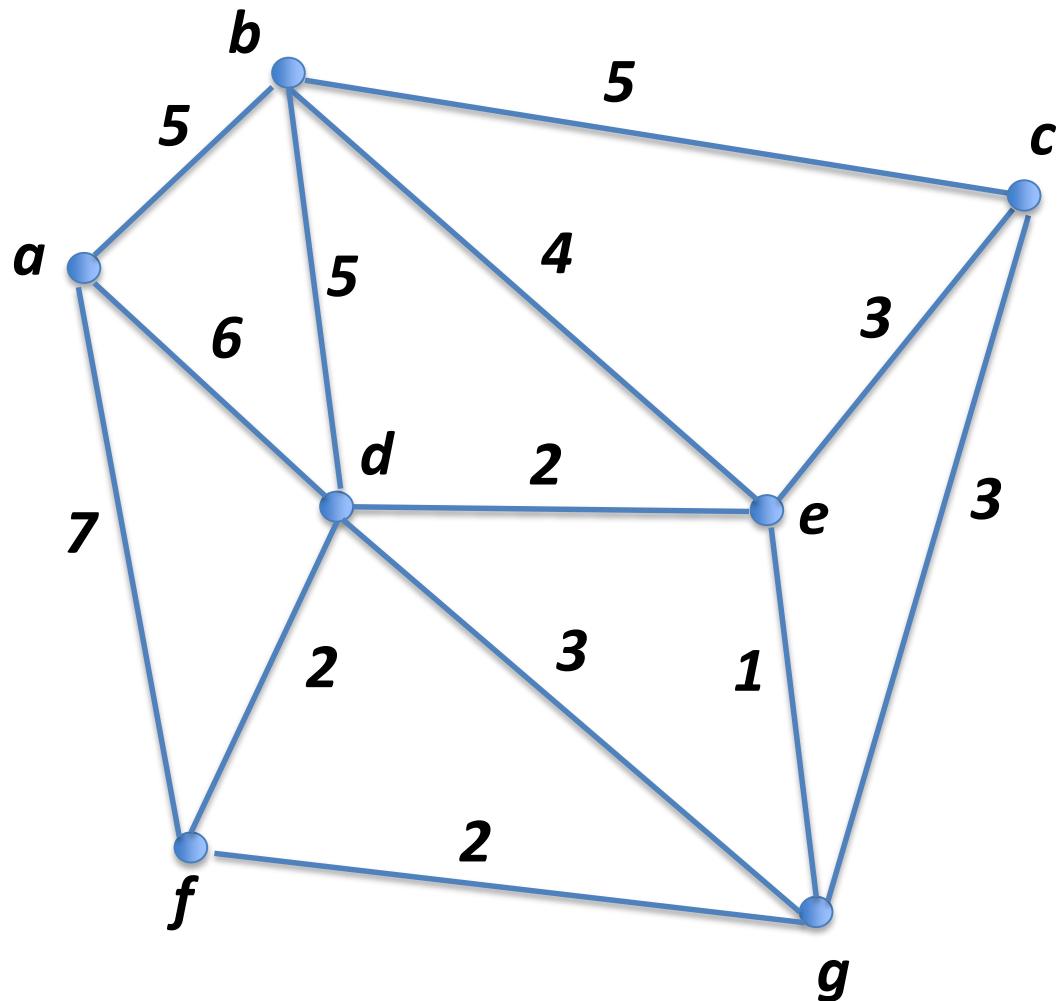
- (a) R es un árbol no dirigido
- (b) R es conexo y acíclico

ARBOLES DE EXPANSION DE RELACIONES CONEXAS

Si R es una relación simétrica conexa sobre un conjunto A , un árbol T en A es un árbol de expansión para R si T es un árbol con exactamente los mismos vértices que R y se puede obtener T eliminando algunas aristas de R



ARBOLES DE EXPANSION MINIMA



Sea el grafo $G=(V,E)$

- Ponderado
- No dirigido
- Conexo
- Sin lazos

ALGORITMO DE PRIM

PASO 1

Hacemos el contador $i = 1$

Colocamos un vértice arbitrario $v_1 \in V$ en el conjunto P .

Definimos $N = V - \{v_1\}$ y $T = \emptyset$

PASO 2

Para $1 \leq i \leq n - 1$, donde $|V| = n$, sean $P = \{v_1, v_2, \dots, v_i\}$
 $T = \{e_1, e_2, \dots, e_{i-1}\}$ y $N = V - P$.

Añadimos a T la arista mas corta (la de peso mínimo) de G que conecta un vértice x en P con un vértice $y (=v_{i+1})$ en N .

Colocamos y en P y lo eliminamos de N

PASO 3

Hacemos $i = i + 1$

Si $i = n$, el subgrafo de G dado por las aristas e_1, e_2, \dots, e_{n-1} es conexo, con n vértices, $n - 1$ aristas y es un árbol óptimo para G .

Si $i < n$, regresamos al paso 2.²⁰¹⁸

ALGORITMO DE PRIM

Inicialización: $i = 1; P = \{a\}; N = \{b, c, d, e, f, g\}; T = \emptyset$

Primera iteración: $T = \{\{a, b\}\}; P = \{a, b\}; N = \{c, d, e, f, g\}; i = 2$

Segunda iteración: $T = \{\{a, b\}, \{b, e\}\}; P = \{a, b, e\}; N = \{c, d, f, g\}$
 $i = 3$

Tercera iteración: $T = \{\{a, b\}, \{b, e\}, \{e, g\}\}; P = \{a, b, e, g\}$
 $N = \{c, d, f\}; i = 4$

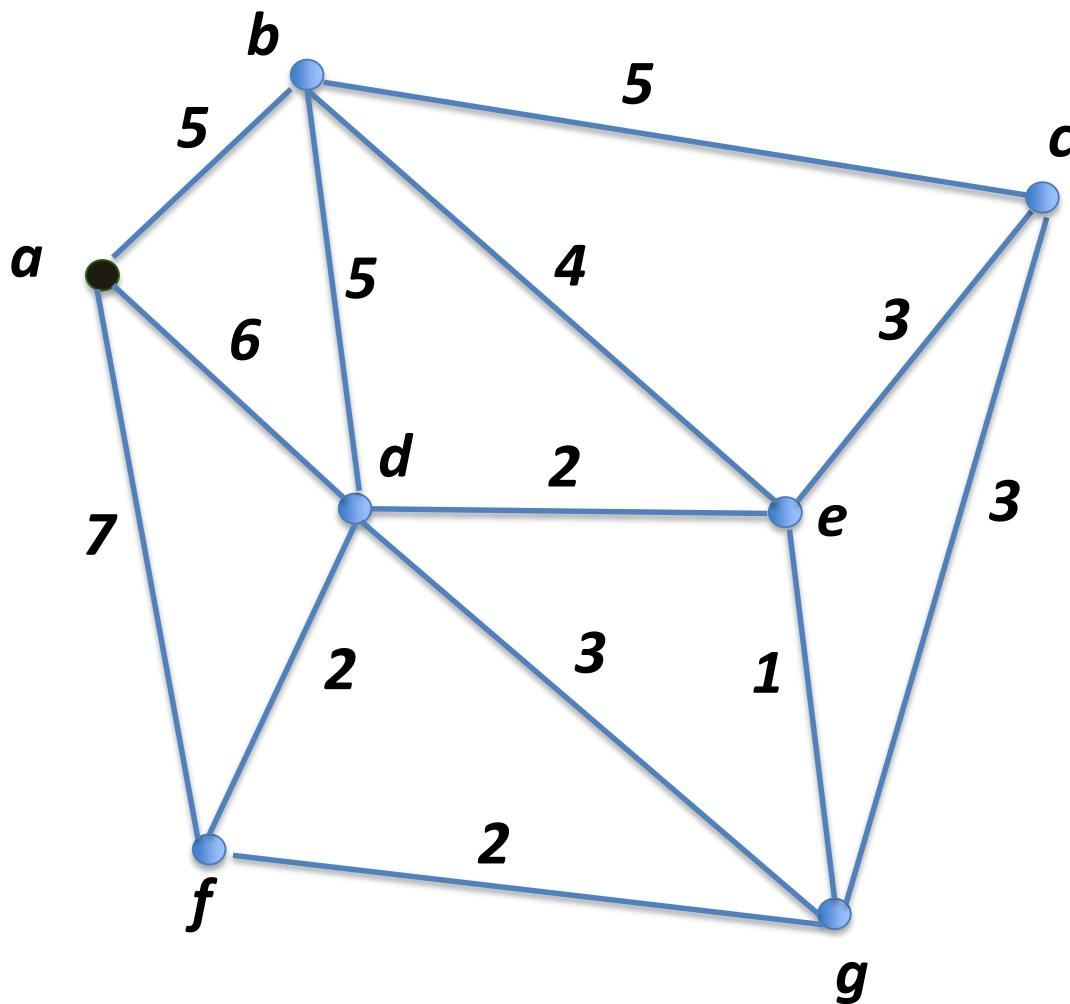
Cuarta iteración: $T = \{\{a, b\}, \{b, e\}, \{e, g\}, \{d, e\}\}$
 $P = \{a, b, e, g, d\}; N = \{c, f\}; i = 5$

Quinta iteración: $T = \{\{a, b\}, \{b, e\}, \{e, g\}, \{d, e\}, \{f, g\}\}$
 $P = \{a, b, e, g, d, f\}; N = \{c\}; i = 6$

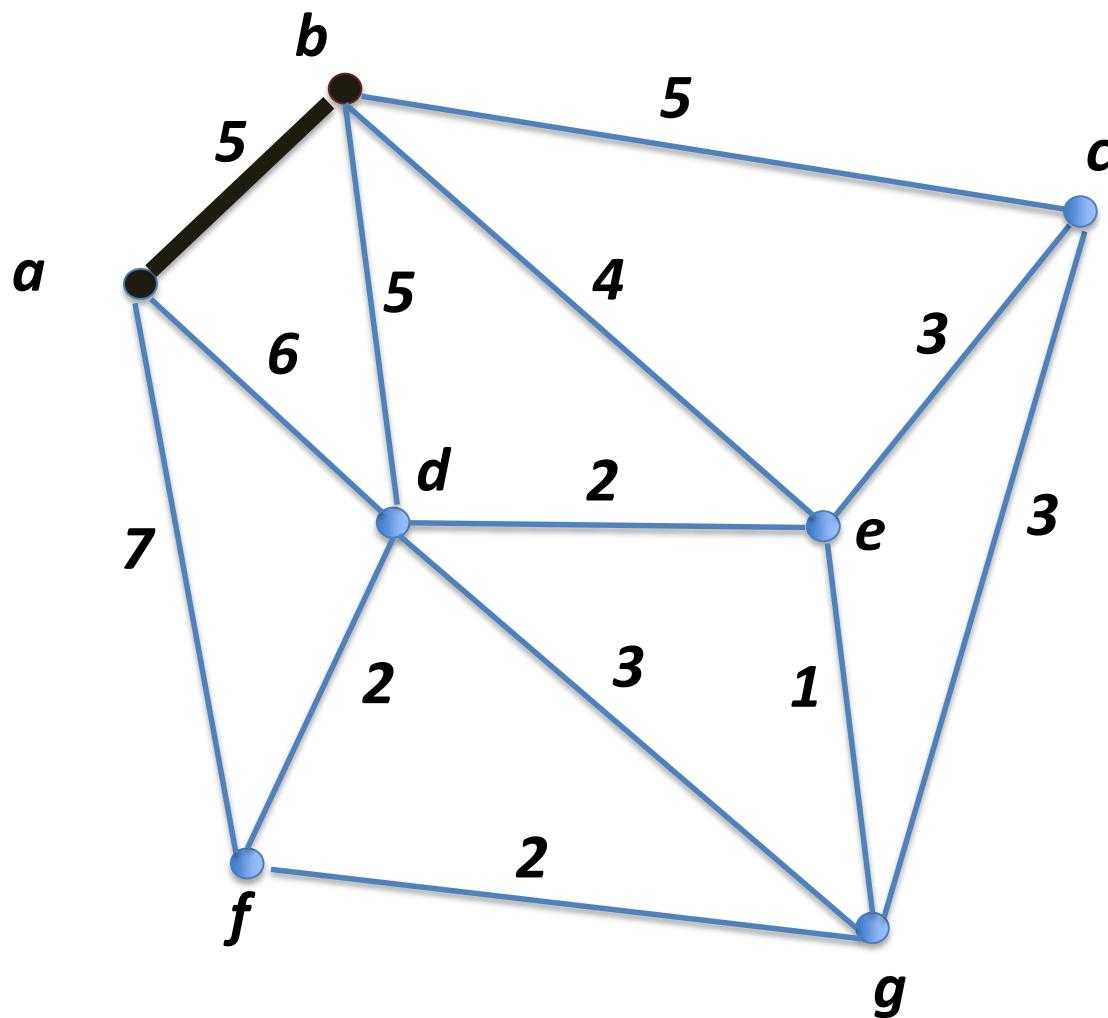
Sexta iteración: $T = \{\{a, b\}, \{b, e\}, \{e, g\}, \{d, e\}, \{f, g\}, \{c, g\}\}$
 $P = \{a, b, e, g, d, f, c\} = V; N = \emptyset; i = 7 = |V|$

Inicialización:

$$i = 1; \quad P = \{a\}; \quad N = \{b, c, d, e, f, g\}; \quad T = \emptyset$$

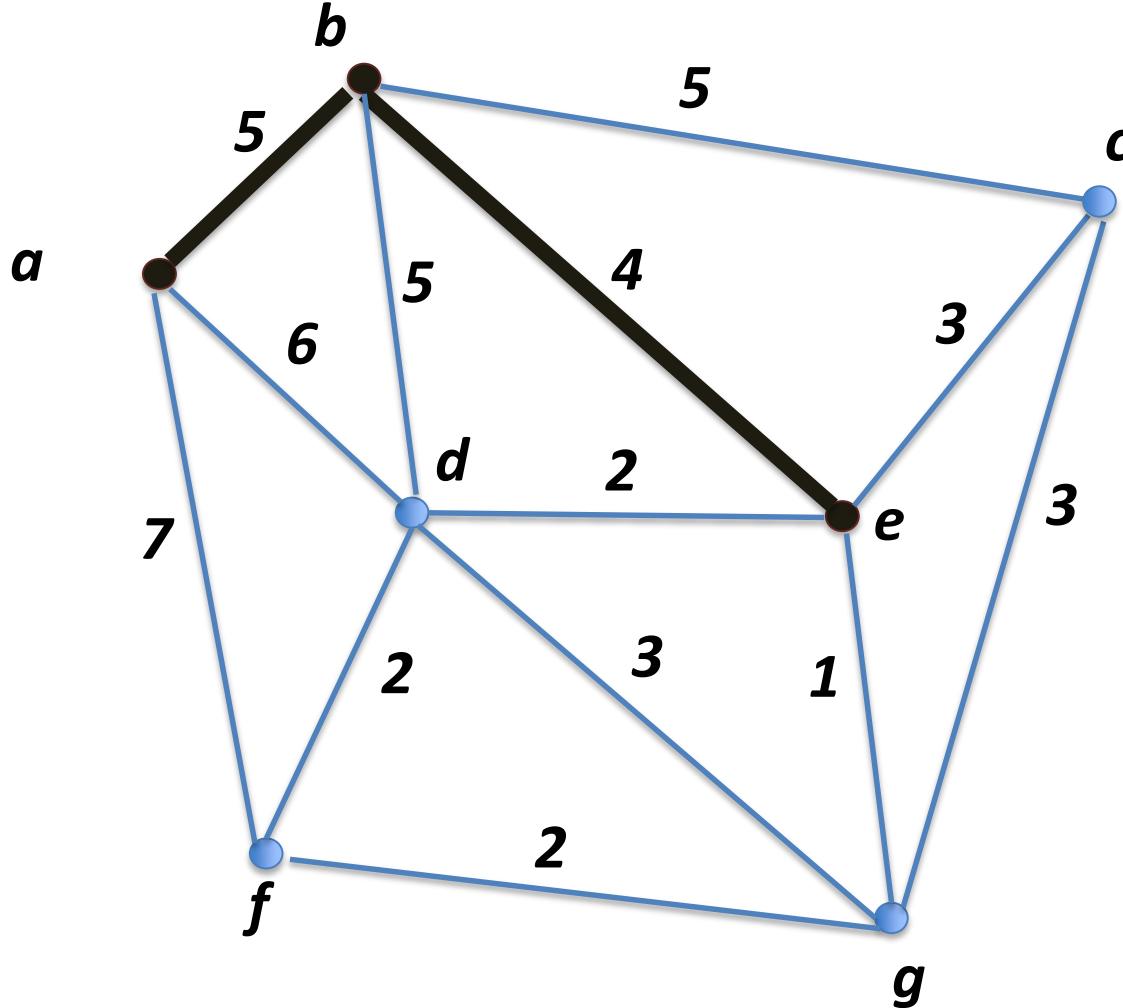


Primera iteración: $T = \{\{a, b\}\}; P = \{a, b\}; N = \{c, d, e, f, g\}; i = 2$

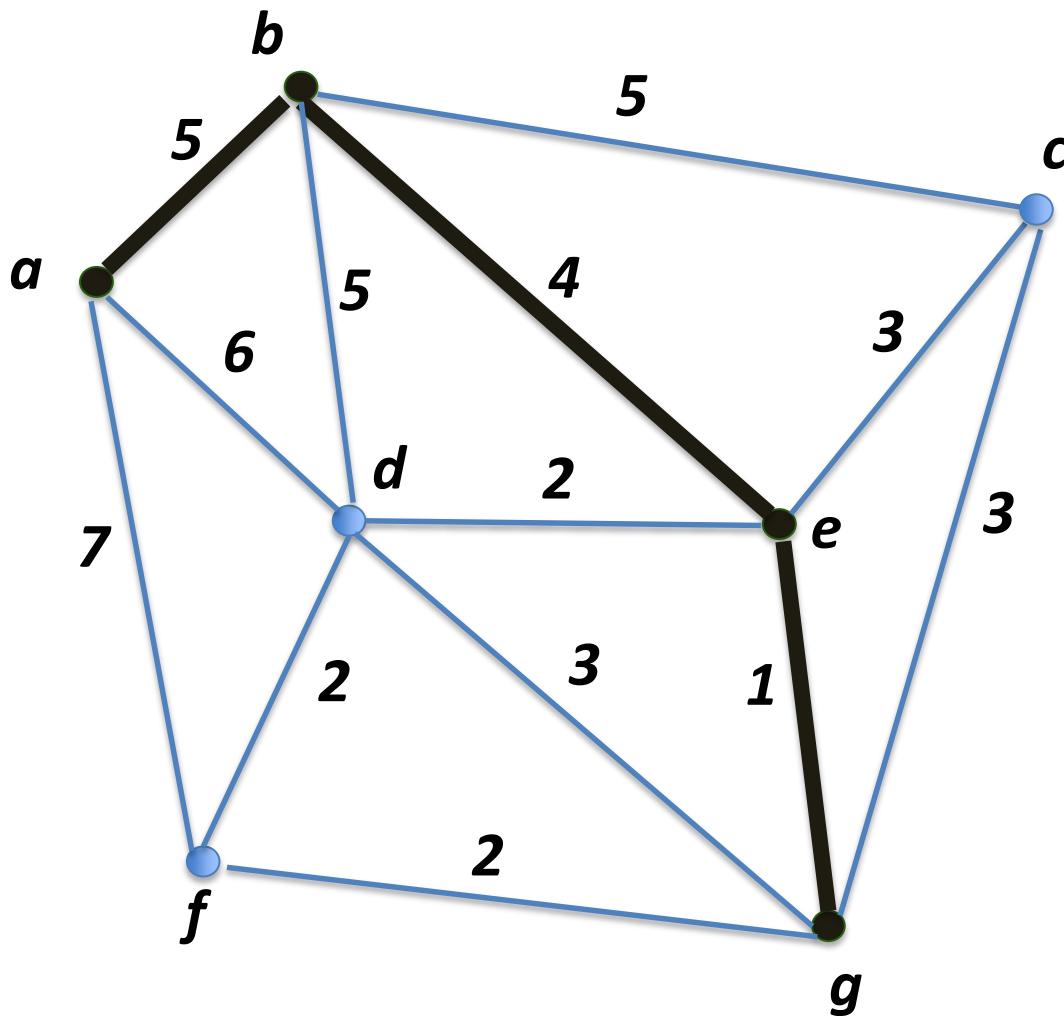


Segunda iteración: $T = \{\{a, b\}, \{b, e\}\}; P = \{a, b, e\}; N = \{c, d, f, g\}$

$i = 3$



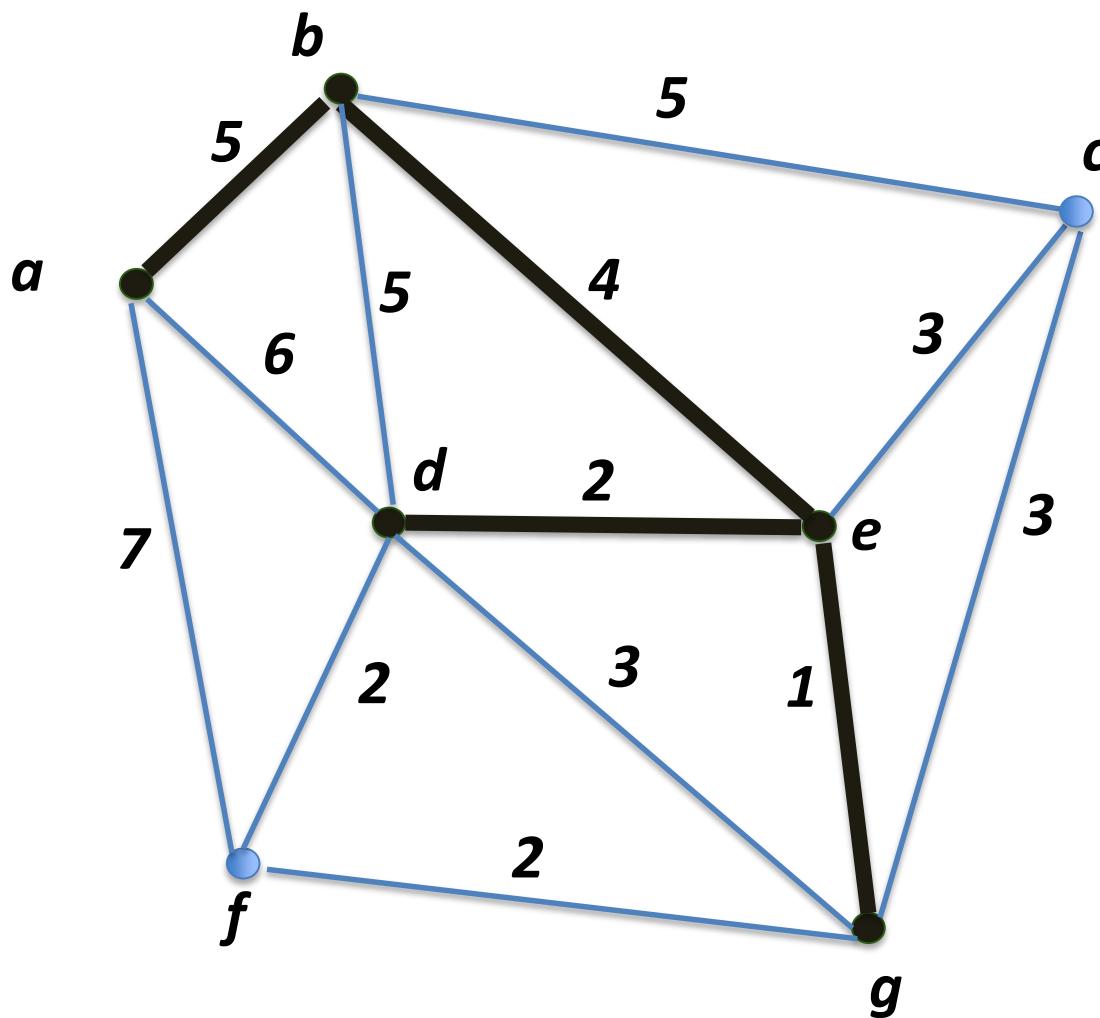
Tercera iteración: $T = \{\{a, b\}, \{b, e\}, \{e, g\}\}; P = \{a, b, e, g\}$
 $N = \{c, d, f\}; i = 4$



Cuarta iteración:

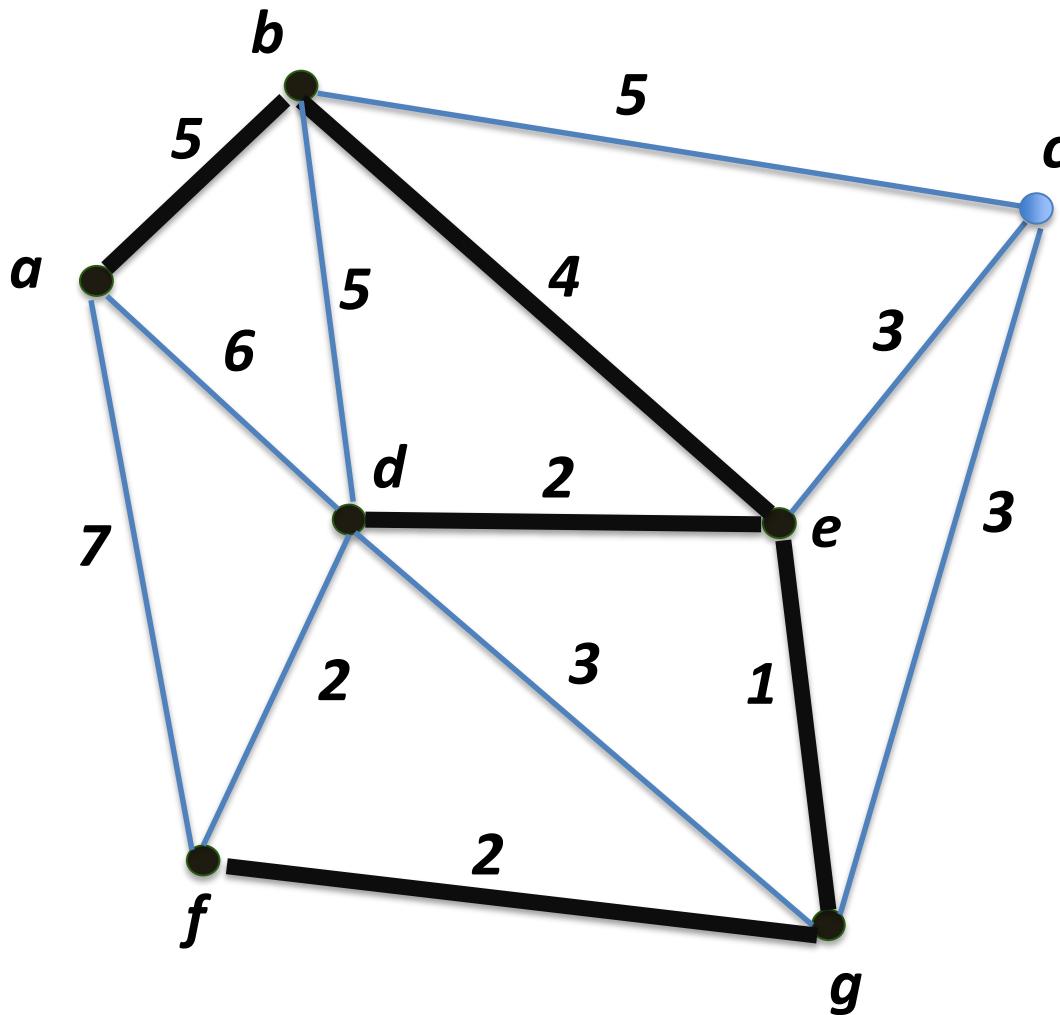
$$T = \{\{a, b\}, \{b, e\}, \{e, g\}, \{d, e\}\}$$

$$P = \{a, b, e, g, d\}; \quad N = \{c, f\}; \quad i = 5$$



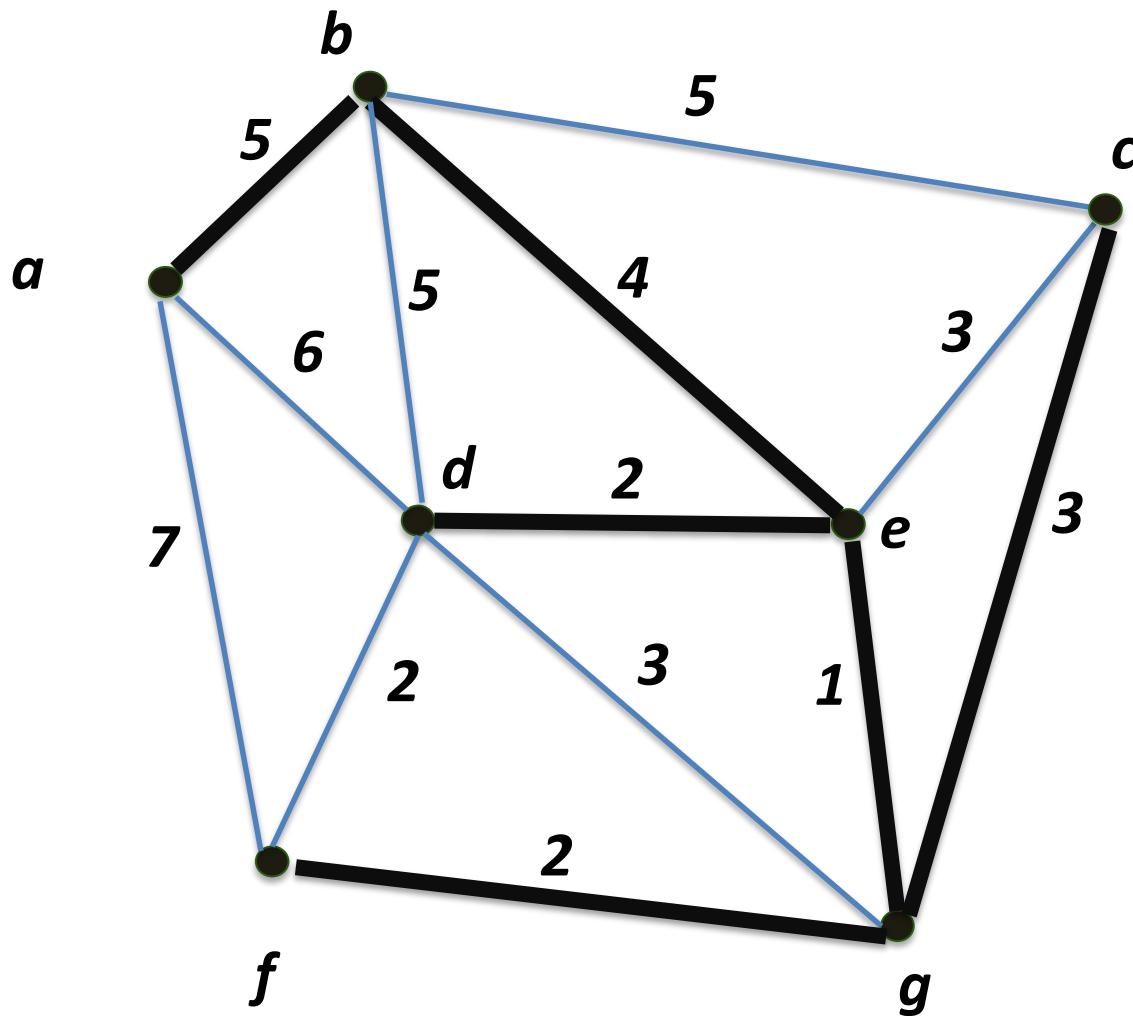
Quinta iteración:

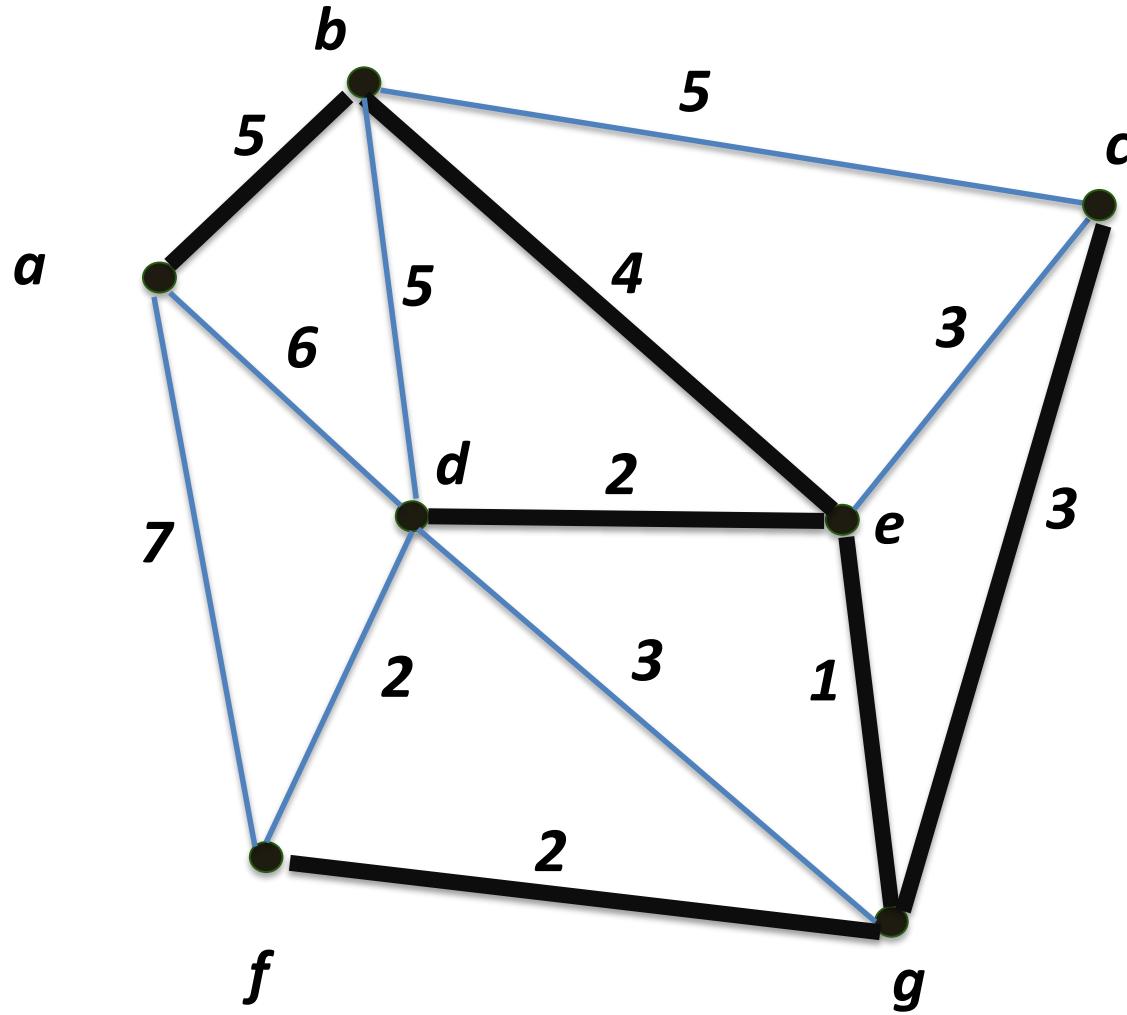
$$T = \{\{a, b\}, \{b, e\}, \{e, g\}, \{d, e\}, \{f, g\}\}$$
$$P = \{a, b, e, g, d, f\}; \quad N = \{c\}; \quad i = 6$$



Sexta iteración: $T = \{\{a, b\}, \{b, e\}, \{e, g\}, \{d, e\}, \{f, g\}, \{c, g\}\}$

$P = \{a, b, e, g, d, f, c\} = V; N = \emptyset; i = 7 = |V|$





T es un árbol de expansión mínimo de peso 17 para G

ALGORITMO DE KRUSKAL

PASO 1. Hacemos el contador $i = 1$

Seleccionamos una arista e_1 en G , tal que $p(e_1)$ séalo mas pequeño posible.

PASO 2. Para $1 \leq i \leq n - 2$, si hemos seleccionado las aristas e_1, e_2, \dots, e_i , entonces seleccionamos la arista e_{i+1} de las aristas restantes de G , de modo que:

- a) $p(e_{i+1})$ sea lo mas pequeño posible.
- b) El subgrafo de G determinado por las aristas $e_1, e_2, \dots, e_i, e_{i+1}$ (y los vértices incidentes) no contenga ciclos.

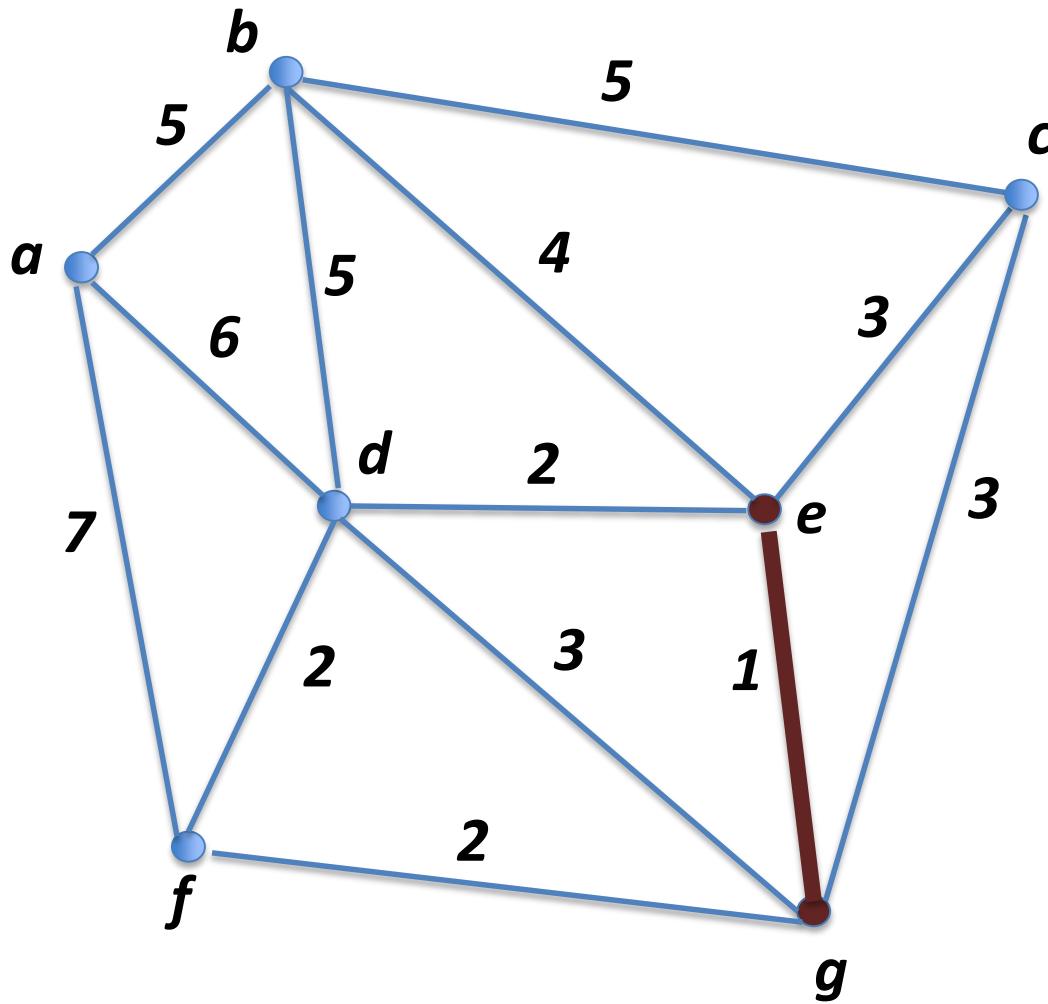
PASO 3. Hacemos $i = i + 1$.

Si $i = n - 1$, el subgrafo de G determinado por las aristas e_1, e_2, \dots, e_{n-1} es conexo, con n vértices y $n - 1$ aristas, yes un árbol de expansión mínimo para G .

Si $i < n - 1$, regresamos al paso 2.

INICIALIZACION: $i = 1$

Seleccionamos la arista de menor peso $\{e, g\}$

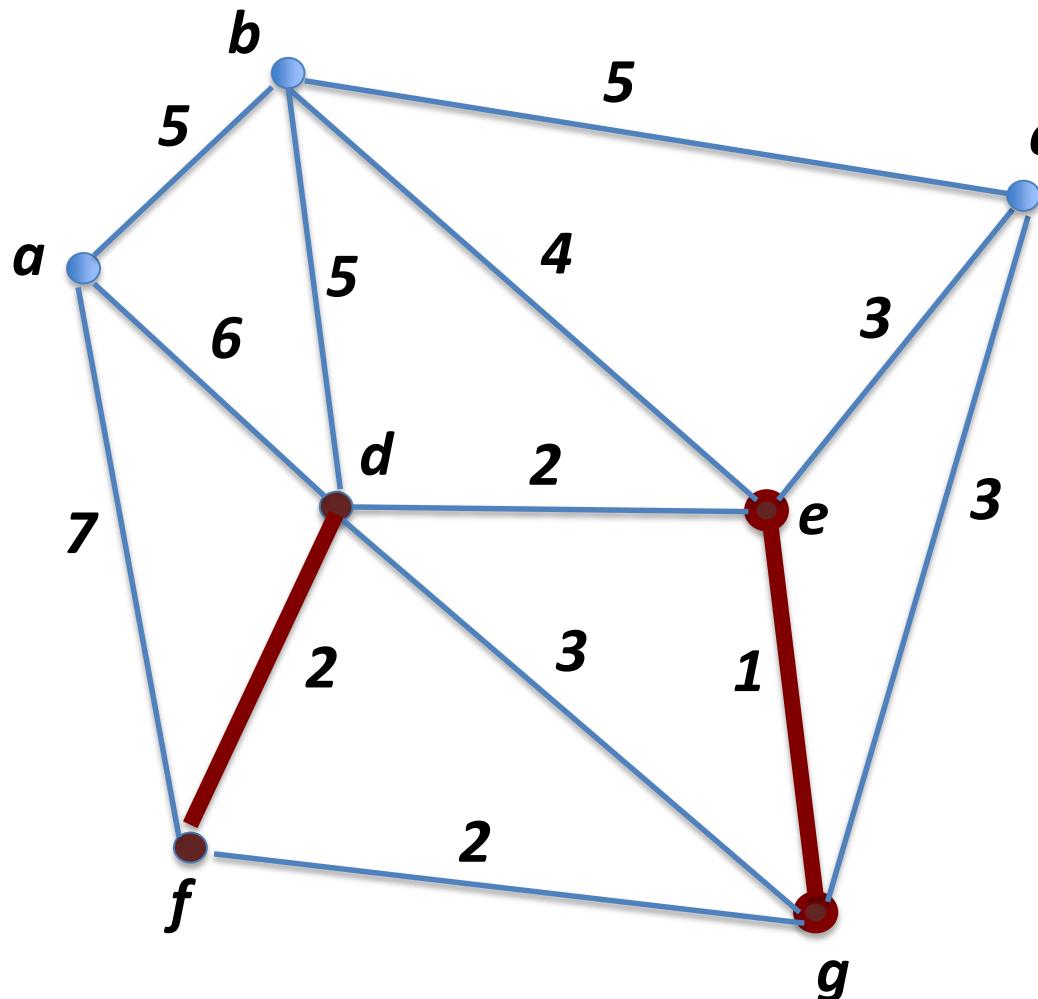


PRIMERA ITERACION:

Entre las aristas restantes tres tienen peso 2, seleccionamos $\{d, f\}$

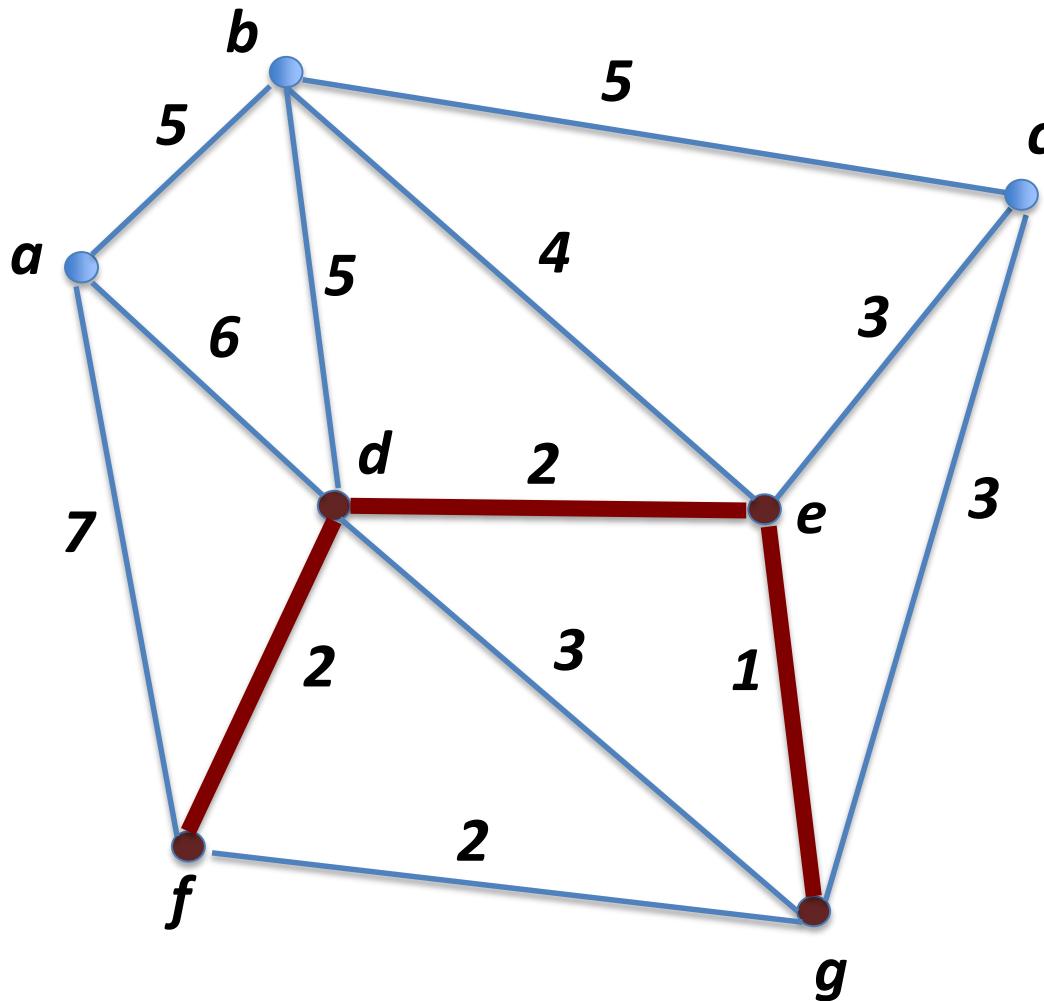
$T = \{\{e, g\}, \{d, f\}\}$ Hacemos $i = i + 1 = 2$

$i = 2 < 6$ volvemos al paso 2.



SEGUNDA ITERACION:

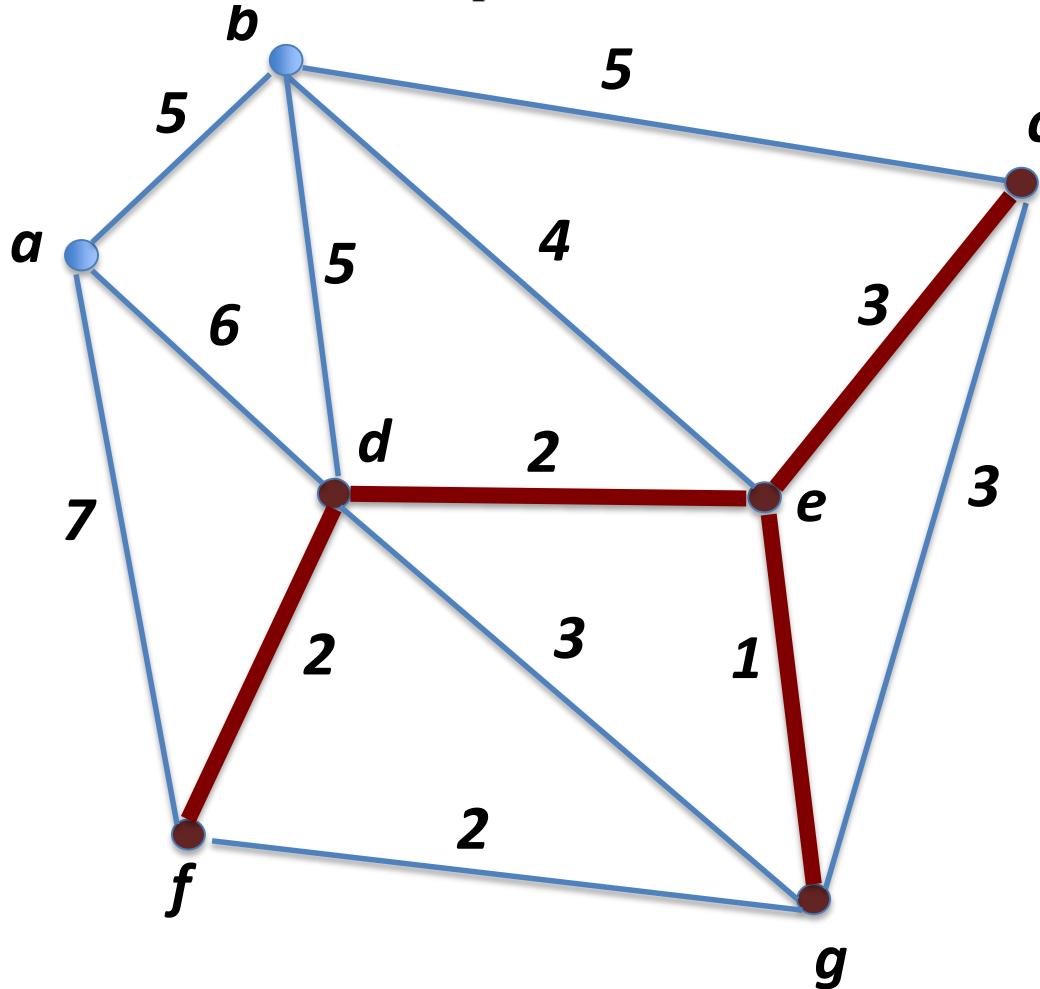
Dos de las aristas restantes tienen peso 2. Seleccionamos $\{d, e\}$
 $T = \{\{e, g\}, \{d, f\}, \{d, e\}\}$ Hacemos $i = i + 1 = 3$
 $i = 3 < 6$ volvemos al paso 2.



TERCERA ITERACION:

De las aristas restantes $\{f, g\}$ tiene el peso mínimo, pero produce un ciclo, en consecuencia es descartada y se elige $\{c, e\}$.

$T = \{\{e, g\}, \{d, f\}, \{d, e\}, \{c, e\}\}$ Hacemos $i = i + 1 = 4$
 $i = 4 < 6$ volvemos al paso 2.

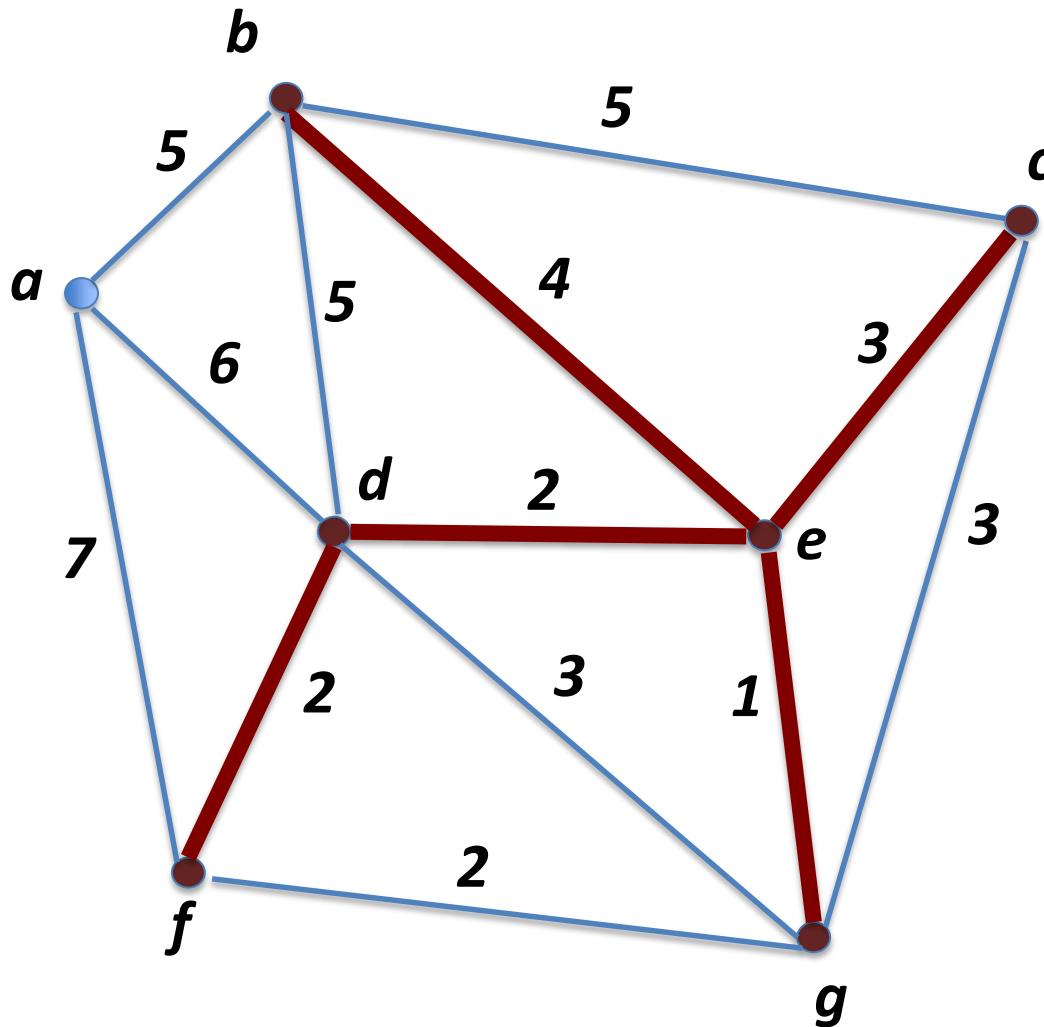


CUARTA ITERACION:

Se elige $\{b, e\}$.

$T = \{\{e, g\}, \{d, f\}, \{d, e\}, \{c, e\}, \{b, e\}\}$ Hacemos $i = i + 1 = 5$

$i = 5 < 6$ volvemos al paso 2.



QUINTA ITERACION:

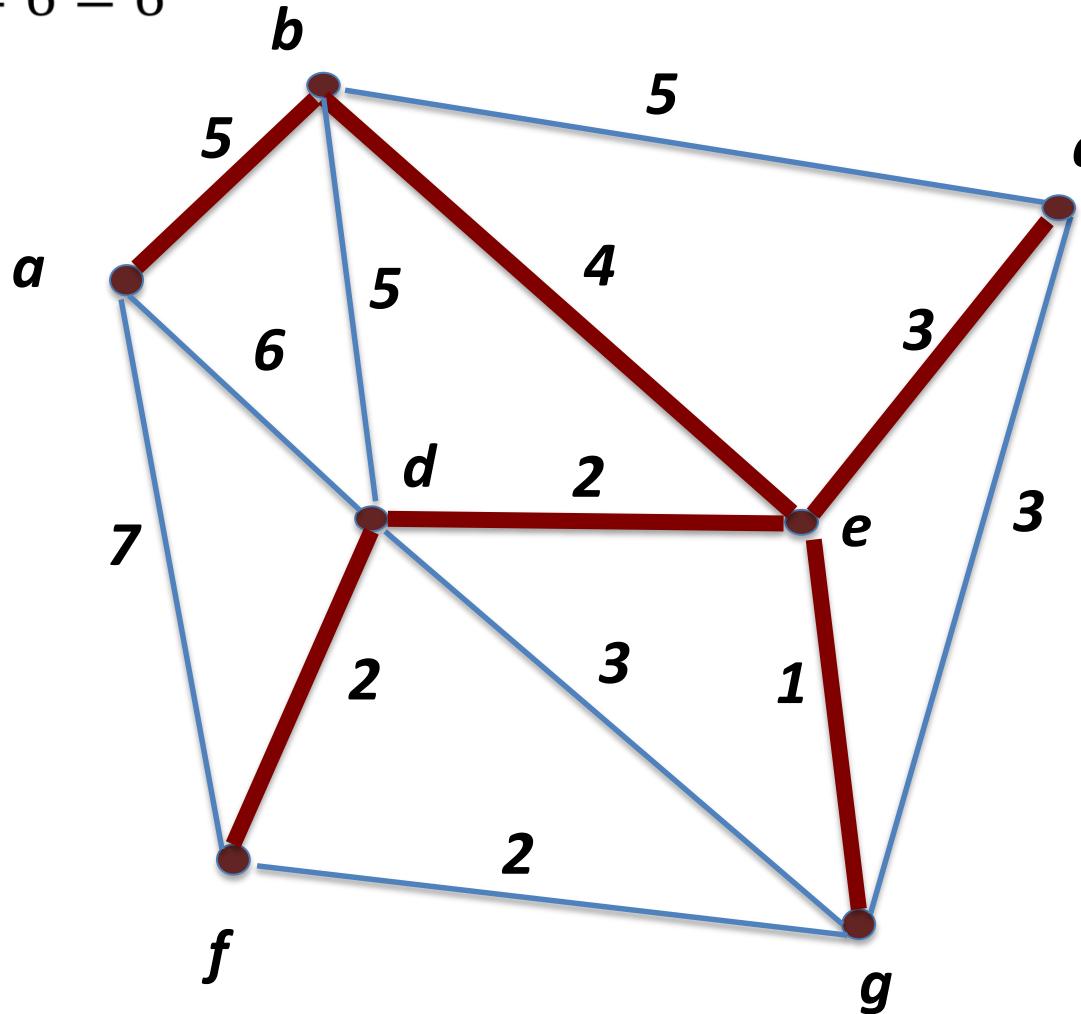
Se elige $\{a, b\}$.

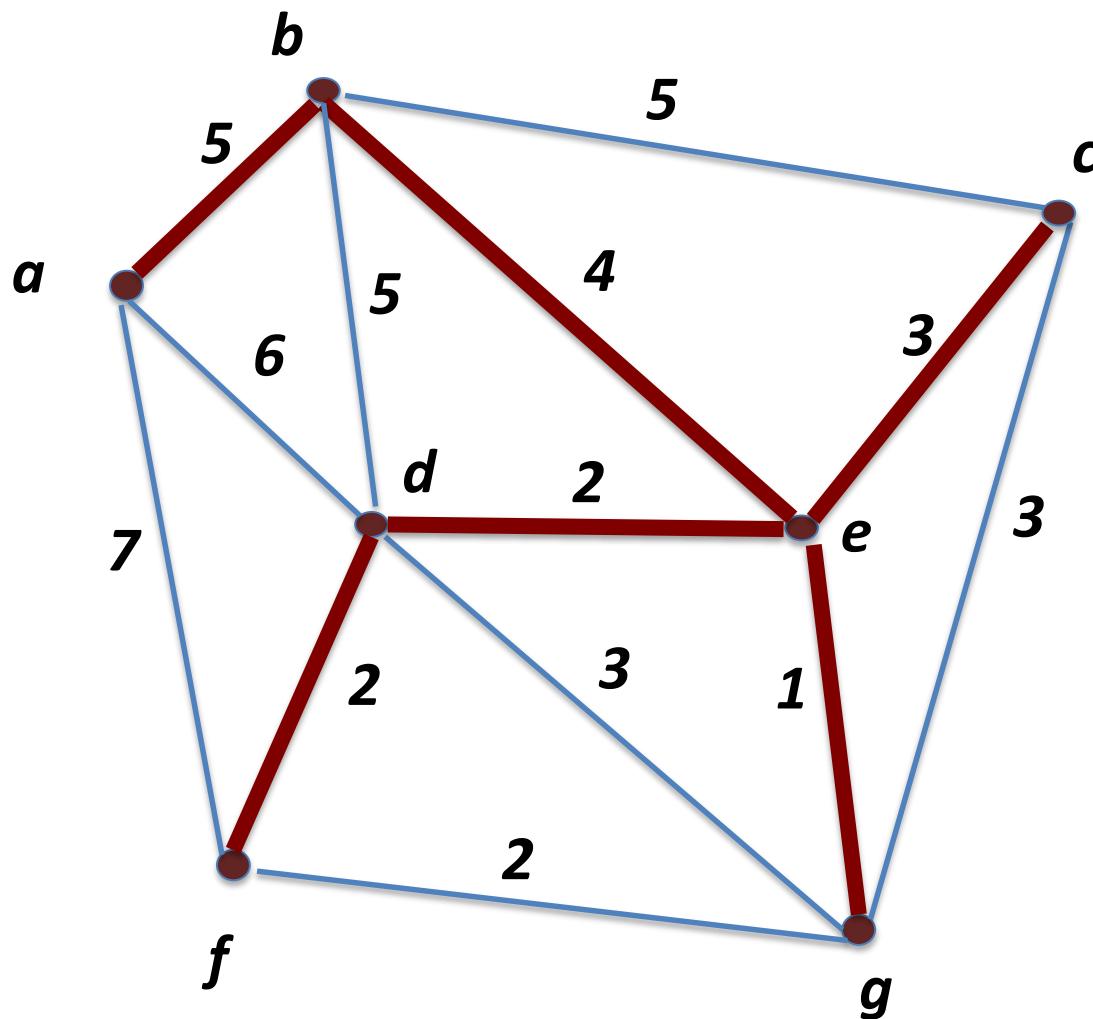
$$T = \{\{e, g\}, \{d, f\}, \{d, e\}, \{c, e\}, \{b, e\}, \{a, b\}\}$$

Hacemos $i = i + 1 = 6$

$$i = 6 = 6$$

FIN





T es un árbol de expansión mínimo de peso 17 para G
2018

