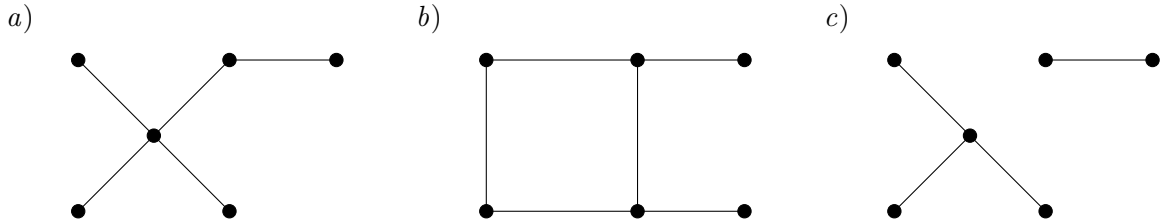
Práctica 6 - Árboles

1. Determinar cuáles de los siguientes grafos son árboles.



2. Determinar todos los valores de $n \in \mathbb{N}$ (y $m \in \mathbb{N}$ si corresponde) para los cuales los siguientes grafos son árboles.

   a) El grafo completo K_n.
   b) El camino P_n.
   c) El ciclo C_n.
   d) El grafo bipartito completo $K_{m,n}$.
   e) El n-cubo Q_n.

3. Probar que todo árbol es bipartito.

4. Describir todos los árboles T que tienen exactamente dos hojas.

5. Sea G un grafo con $|E(G)| < |V(G)| - 1$. Probar que G no es conexo.

6. En cada caso, dar un grafo G que tenga las propiedades indicadas o explicar por qué no existe tal grafo.

   a) G acíclico, $|E(G)| = 4$ y $|V(G)| = 6$.
   b) G árbol 2-regular.
   c) G árbol, $|V(G)| = 6$ y los vértices de G tienen grados 1, 1, 1, 1, 3 y 3.
   d) G árbol con 10 vértices, de los cuales exactamente 6 son hojas.

7. Sean T_1 y T_2 dos árboles tales que $|E(T_1)| = 17$ y $|V(T_2)| = 2|V(T_1)|$. Determinar $|V(T_1)|$, $|V(T_2)|$ y $|E(T_2)|$.

8. a) Probar que cada componente conexa de un bosque es un árbol.
   b) Sea F un bosque con $\kappa(F) = k$ y $|V(F)| = n$. Determinar $|E(F)|$.
   c) Sea F un bosque con $\kappa(F) = 7$ y $|E(F)| = 40$. Determinar $|V(F)|$.
   d) Sea F es un bosque con $|V(F)| = 62$ y $|E(F)| = 51$. Determinar $\kappa(F)$.

9. Sea T un árbol.

   a) Probar que toda arista $e \in E(T)$ es una arista de corte.
   b) Probar que todo vértice $v \in V(T)$ con $d(v) \geqslant 2$ es un vértice de corte.
   c) ¿Es cierto el ítem anterior si $d(v) = 1$?

10. Sea G un grafo simple y conexo, que no es isomorfo a un grafo completo. Probar que G es un árbol si y sólo si cuando se agrega una arista entre dos vértices cualesquiera no adyacentes, se crea exactamente un ciclo.

Árboles recubridores

11. *a*) Probar que todo grafo conexo G tiene un árbol recubridor.

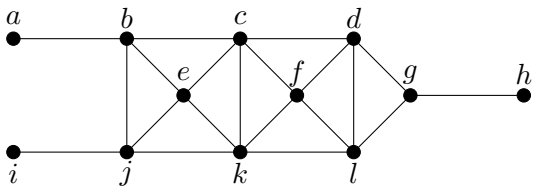
    b) ¿En qué condiciones una arista en un grafo conexo G está contenida en todo árbol recubridor de G?

12. *a*) Sea T un árbol recubridor para un grafo G. Probar que si una arista e está en G pero no en T, entonces al agregar e a T se crea exactamente un ciclo.

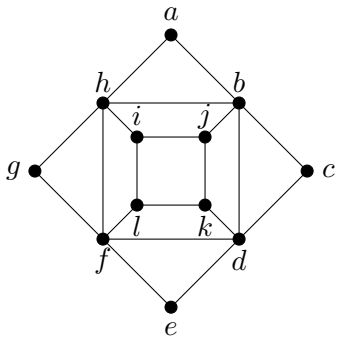
    b) Sean T y T' dos árboles recubridores de un grafo conexo G y $e \in E(T) \backslash E(T')$. Probar que existe $e' \in E(T') \backslash E(T)$ tal que $(T' \cup e) \backslash e'$ es un árbol recubridor de G.

13. Dar un árbol recubridor para cada uno de los siguientes grafos.

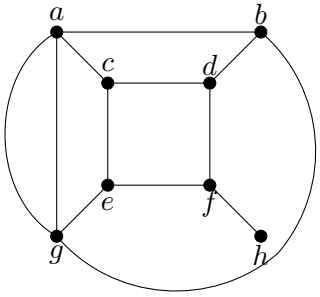
a)



b)



14. Considerar el siguiente grafo G.



a) Usar el algoritmo de búsqueda en anchura (BFS) para dar un árbol recubridor de G con el orden de vértices dado en cada caso.

I. $hgfedcba$ II. $hfdbgeca$ III. $chbgadfe$

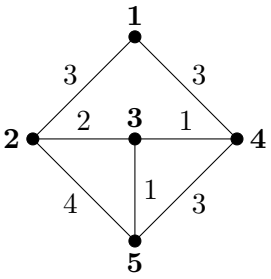
b) Usar el algoritmo de búsqueda en profundidad (DFS) para dar un árbol recubridor de G con el orden de vértices dado en cada caso.

I. $hgfedcba$ II. $hfdbgeca$ III. $dhcbefag$

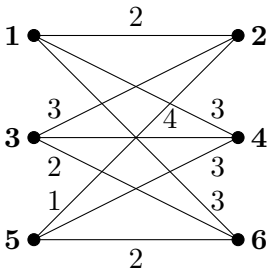
ESCUELA DE CIENCIAS EXACTAS Y NATURALES
DEPARTAMENTO DE MATEMÁTICA
Av. Pellegrini 250. Rosario
+54 0341 - 480 2649 internos 216 - 119

15. Dar un árbol recubridor de peso mínimo para cada uno de los siguientes grafos, mediante el algoritmo de Kruskal.

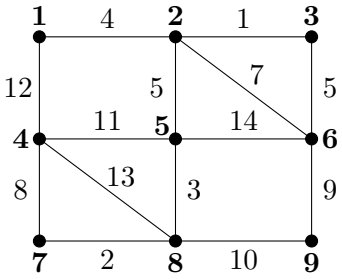
I.



II.



III.



16. *Algoritmo de Prim.*

Entrada. Un grafo conexo ponderado.

Idea. Mantener un subgrafo conexo H e ir agregando nuevos vértices incidentes en aristas de peso mínimo.

Inicio. Elegir $v \in V(G)$. $V(H) = \{v\}$, $E(G) = \varnothing$.

Iteración. Mientras $V(H) \neq V(G)$, elegir la arista e de menor peso entre el conjunto de aristas que tienen un extremo en $V(H)$ y otro en $V(G) \backslash V(H)$. Si la arista elegida es $e = uw$ con $u \in V(H)$ y $w \in V(G) \backslash V(H)$, actualizamos $V(H) \leftarrow V(H) \cup \{w\}$, $E(H) \leftarrow E(H) \cup \{e\}$.

a) Probar que si G es un grafo conexo ponderado, el algoritmo de Prim produce un árbol recubridor de G de peso mínimo.

b) Dar un árbol recubridor de peso mínimo para cada uno de los grafos del ejercicio 15, utilizando el algoritmo de Prim.

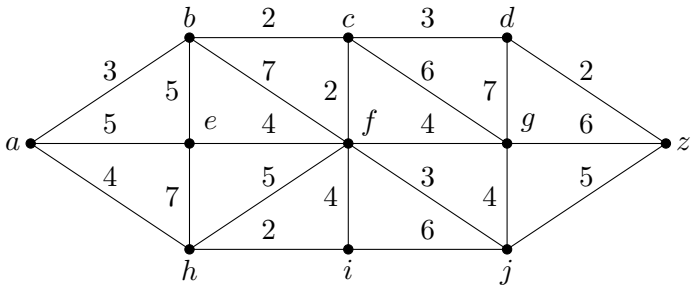
17. Determinar si cada una de las siguiente afirmaciones es verdadera o falsa, justificando adecuadamente.

a) Si todos los pesos en un grafo conexo ponderado G son diferentes, entonces G admite un único árbol recubridor de peso mínimo.

b) Si todos los pesos en un grafo conexo ponderado G son diferentes, entonces los árboles recubridores de G distintos tienen pesos distintos.

Algoritmo de Dijkstra

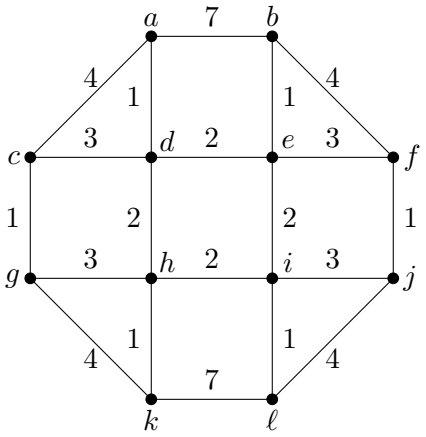
18. Aplicar el algoritmo de Dijkstra al grafo ponderado de la figura y determinar la distancia entre los pares de vértices indicados.



a) a y f. *b*) a y g. *c*) a y z. *d*) b y j. *e*) h y b. *f*) h y d.

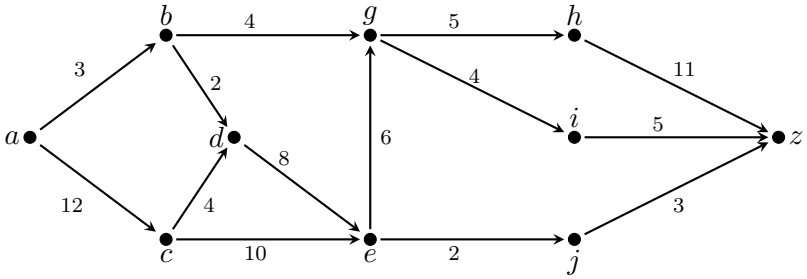
19. Suponer que el algoritmo de Dijkstra tiene como entrada un grafo ponderado G que no es conexo, y un vértice inicial $v \in V(G)$. Si w es un vértice de G que no está en la misma componente conexa que v ¿qué valor tiene $t(v)$ al finalizar la ejecución del algoritmo?

20. *a*) Aplicar el algoritmo de Dijkstra al grafo ponderado de la figura y determinar la distancia del vértice a a cada uno de los otros vértices.



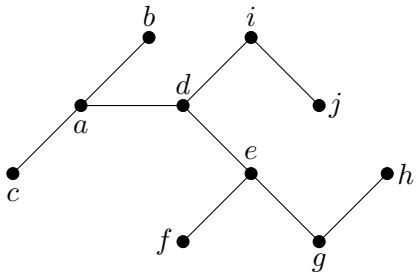
b) Dar un camino de longitud mínima desde el vértice a a los vértices f, g y ℓ.

21. Considerar el siguiente digrafo ponderado G. Determinar la longitud de una ruta más corta desde el vértice a a cada uno de los otros vértices de G.



Árboles binarios

22. Considerar el siguiente árbol, con el vértice a como raíz.



ESCUELA DE CIENCIAS EXACTAS Y NATURALES
DEPARTAMENTO DE MATEMÁTICA
Av. Pellegrini 250. Rosario
+54 0341 - 480 2649 internos 216 - 119

a) Determinar el nivel de cada vértice.

b) Determinar la altura del árbol.

c) ¿Se trata de un árbol binario?

d) Repetir los ítems anteriores pero considerando al vértice g como raíz.

23. En cada caso, dar un grafo que tenga las propiedades indicadas o explicar por qué no existe.

a) Árbol binario completo, 4 vértices internos, 5 hojas.

b) Árbol binario completo, altura 3, 9 hojas.

c) Árbol binario completo, altura 4, 9 hojas.

24. Un árbol enraizado es *m-ario* (con $m \in \mathbb{N}$) si todo vértice tiene a lo sumo m hijos. Si en particular todo vértice tiene 0 o m hijos, se dice *m-ario completo*. Sea T un árbol m-ario completo con i vértices internos.

a) Determinar la cantidad de hojas de T.

b) Determinar la cantidad de vértices de T.

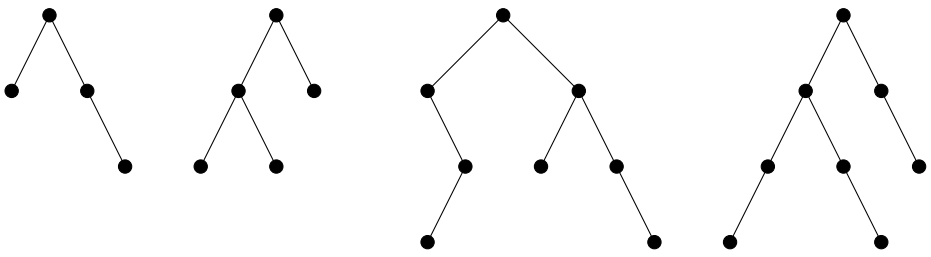
25. *a*) Un árbol 3-ario (o *ternario*) completo $T = (V, E)$ tiene 34 vértices internos. ¿Cuántas aristas tiene T? ¿Cuántas hojas?

b) ¿Cuántos vértices internos tiene un árbol 5-ario completo con 817 hojas?

26. Un árbol binario T está *balanceado* si para cada vértice $v \in V(T)$ las alturas de los subárboles izquierdo y derecho de v difieren en a lo sumo 1 (la altura de un árbol vacío se define como -1).

Establecer si cada uno de los siguientes árboles está o no balanceado.

(a) (b) (c) (d)



Práctica complementaria

Árboles recubridores

1. Determinar si cada una de las siguiente afirmaciones es verdadera o falsa, justificando adecuadamente.

   a) Si G es un grafo conexo y T un árbol recubridor de G, entonces existe un orden de los vértices de G tal que el algoritmo BFS tiene como salida al árbol recubridor T.

   b) Si G es un grafo conexo y T un árbol recubridor de G, entonces existe un orden de los vértices de G tal que el algoritmo DFS tiene como salida al árbol recubridor T.

2. *a*) Escribir un algoritmo basado en el BFS para determinar si un grafo es o no conexo.

   b) Escribir un algoritmo basado en el DFS para determinar si un grafo es o no conexo.

3. *a*) Sea G un grafo conexo ponderado, $v \in V(G)$ y e una arista con peso mínimo incidente en v. Probar que e está contenida en algún árbol recubridor de peso mínimo.

   b) Sea G un grafo conexo ponderado, $v \in V(G)$. Suponer que los pesos de las aristas incidentes en v son distintos. Sea e la arista con peso mínimo incidente en v. ¿Debe estar e contenida en todo árbol recubridor de peso mínimo?

4. *a*) Sea G un grafo ponderado. Probar que si, mientras sea posible, se elimina una arista de G con peso máximo cuya eliminación no desconecta a G, el resultado es un árbol recubridor de peso mínimo para G.

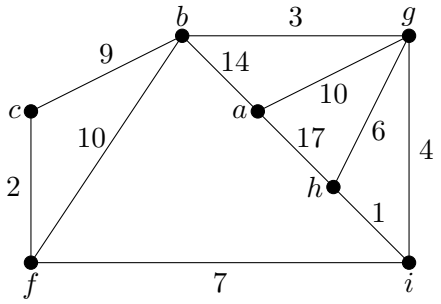
   b) Escribir un algoritmo que encuentre un árbol recubridor de peso mínimo en un grafo conexo ponderado utilizando el resultado del ítem anterior.

Algoritmo de Dijkstra

5. *a*) Modificar el algoritmo de Dijkstra y escribir un algoritmo que dado un grafo conexo ponderado G y un vértice $v \in V(G)$, devuelva $d(v, w)$ y dé un (v, w)-camino de longitud mínima para todo $w \in V(G)$.

   b) Modificar el algoritmo de Dijkstra y escribir un algoritmo que dado un grafo conexo ponderado G y dos vértices $v, w \in V(G)$, devuelva $d(v, w)$.

6. *a*) Aplicar el algoritmo de Dijkstra al grafo ponderado de la figura y determinar la distancia del vértice a a cada uno de los otros vértices.



   b) Dar un camino de longitud mínima desde el vértice a a los vértices c, f e i.

ESCUELA DE CIENCIAS EXACTAS Y NATURALES
DEPARTAMENTO DE MATEMÁTICA
Av. Pellegrini 250. Rosario
+54 0341 - 480 2649 internos 216 - 119

7. Determinar si cada una de las siguiente afirmaciones es verdadera o falsa, justificando adecuadamente.

   a) Sea G un grafo ponderado con $V(G) = \{v_1, v_2, \ldots, v_n\}$ y $w(e^*) < w(e)$ para toda arista $e \in E(G)$, con $e \neq e^*$ (donde $w(e)$ denota el peso asignado a la arista e). Si aplicamos el algoritmo de Dijkstra a G y calculamos la distancia de v_1 a cada uno de los vértices v_i para $2 \leqslant i \leqslant n$, entonces existe un vértice v_j, para algún $2 \leqslant j \leqslant n$, tal que la arista e^* se use en el camino más corto de v_1 a v_j.

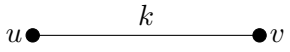
   b) El algoritmo de Dijkstra encuentra la longitud de una ruta más corta en un grafo conexo ponderado incluso si algunos pesos son negativos.

   c) Dados un grafo conexo ponderado G con $w(e) \geqslant 0$ para cada $e \in E(G)$ y $a, z \in V(G)$, el siguiente algoritmo devuelve la distancia entre a y z.

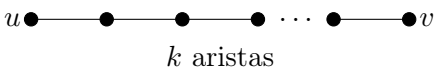
```
algo (w, a, z) {
    longitud = 0
    v = a
    T = conjunto de todos los vertices
    while (v not in z) {
        T = T - {v}
        seleccionar x in T con w(v,x) minimo
        longitud = longitud + w(v,x)
        v = x
    }
    return longitud
}
```

8. *a*) Escribir un algoritmo basado en el BFS que, dado un grafo no ponderado G y $v \in V(G)$, devuelva la distancia desde v a todos los demás vértices del grafo.

   b) Sea G un grafo ponderado en el que el peso de cada arista es un entero positivo. Sea G' el grafo obtenido a partir de G al sustituir cada arista



   en G de peso k por un camino P_k de aristas no ponderadas.



k aristas

   Demostrar que el algoritmo de Dijkstra para encontrar la distancia en el grafo ponderado G desde un vértice fijo v a todos los demás y realizar la búsqueda en anchura en el grafo no ponderado G' comenzando en el vértice v son el mismo proceso.

Árboles binarios

9. *a*) Sea T un árbol binario. Si $|V(T)| = n$, ¿cuál es la máxima altura posible de T?

   b) Sea T un árbol binario completo. Si $|V(T)| = n$, ¿cuál es la máxima altura posible de T en este caso?

   c) Repita los ítems anteriores para un árbol m-ario y m-ario completo respectivamente.

10. *a*) ¿Cuál es el número máximo de vértices internos que puede tener un árbol cuaternario completo de altura 8?

b) ¿Cuál es el número máximo de vértices internos que puede tener un árbol m-ario completo de altura h?

11. Sea N_h el número mínimo de vértices en un árbol binario balanceado de altura h, y sea $f_1, f_2, \ldots$ la sucesión de Fibonacci.

    a) Probar que $N_0 = 1$, $N_1 = 2$ y $N_2 = 4$.

    b) Probar que $N_h = 1 + N_{h-1} + N_{h-2}$, para $h \geqslant 2$.

    c) Probar que $N_h = f_{h+3} - 1$, para $h \geqslant 0$.

12. *a*) Cuatro monedas son idénticas en apariencia, pero una es defectuosa y es más pesada o más liviana que las otras, que pesan lo mismo. Dibujar un árbol de decisiones para proporcionar un algoritmo que identifique la moneda defectuosa (pero no necesariamente que determine si es más o menos pesada que las otras) usando sólo una balanza de cruz.

    b) Demostrar que se requiere pesar al menos dos veces para resolver el problema del ítem anterior.

    c) Repetir el ítem (a) pero ahora determinando si la moneda defectuosa es más pesada o más liviana que las otras.

    d) Determinar la menor cantidad de pesadas requeridas (en el peor caso) para resolver el problema del ítem (c).

13. Ocho monedas son idénticas en apariencia, pero una es defectuosa y es más pesada o más liviana que las otras, que pesan lo mismo. Dibujar un árbol de decisiones para proporcionar un algoritmo que identifique (usando la menor cantidad de pesadas en el peor caso) la moneda defectuosa y determine si es más pesada o más liviana que las otras usando sólo una balanza de cruz.

14. Dibujar un árbol de decisiones para proporcionar un algoritmo que ordene de menor a mayor cuatro números reales, usando la menor cantidad comparaciones en el peor caso.

15. Dibujar un árbol de decisiones para proporcionar un algoritmo que dados cuatro números reales determine cuál es el mayor de ellos, usando la menor cantidad de comparaciones en el peor caso.

ESCUELA DE CIENCIAS EXACTAS Y NATURALES
DEPARTAMENTO DE MATEMÁTICA
Av. Pellegrini 250. Rosario
+54 0341 - 480 2649 internos 216 - 119