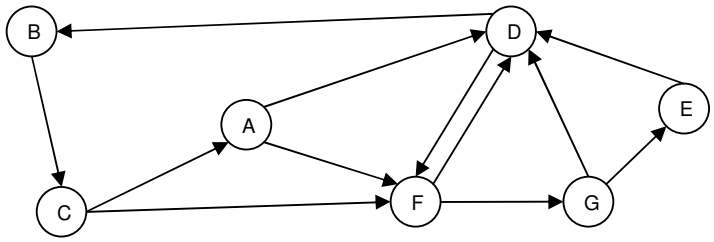
Práctica 8 – Grafos

1. Sin desarrollar un programa, determinar en el siguiente digrafo:



   a) grados de entrada y salida de cada vértice.
   b) todos los ciclos que pasan por el vértice C
   c) todos los caminos del vértice A al E
   d) todos los caminos del vértice B al D
   e) todos los caminos de longitud 4 con origen en el vértice A
   f) la matriz de adyacencia que lo representa.
   g) la lista de adyacencia que lo representa.
   h) la matriz de adyacencia que lo representa, suponiendo que no es un grafo dirigido.

2. Graficar el digrafo representado por la siguiente matriz de adyacencia, asumir que los vértices se numeran de 1 a 4:

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

3. Dibujar un grafo de 5 vértices que tengan respectivamente los siguientes grados: 1, 2, 2, 1, 4.

4. Desarrollar un subprograma que reciba como parámetro una matriz de adyacencia que representa un digrafo sin aristas ponderadas y retorne en la misma matriz el grafo subyacente.

5. Dado un vértice determinado de un digrafo representado en una matriz de adyacencia, implementar funciones que devuelvan:
   a) su grado de entrada
   b) su grado de salida
   c) el grado de dicho vértice

6. Resolver el ejercicio anterior suponiendo que el digrafo está almacenado en una lista de adyacencia.

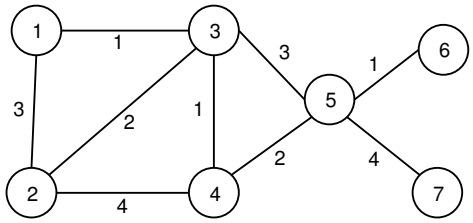
7. Desarrollar una función que obtenga el grado de un vértice (dato) de un grafo almacenado en:
   a) una matriz de adyacencia
   b) una lista de adyacencia
   c) media matriz (triángulo superior)

8. Desarrollar funciones recursivas para:
   a) generar un arreglo con el grado de cada vértice de un grafo almacenado en una matriz de adyacencia.
   b) hallar el vértice con el mayor grado de entrada, en un digrafo representado por una matriz de adyacencia.

c) determinar si todos los vértices de un grafo almacenado en media matriz de adyacencia (triángulo superior) tienen al menos un vértice adyacente con costo mayor a X (dato).
d) generar un vector de registros con grado y vértice para aquellos vértices, de un grafo almacenado en una matriz de adyacencia, cuyo grado sea mayor a 3.

9. Hallar el vértice con el mayor grado de entrada, en un digrafo representado por una lista de adyacencia.

10. Para el siguiente grafo (sin desarrollar programa):



a) mostrar sus vértices utilizando el recorrido en profundidad partiendo del vértice 1
b) mostrar sus vértices utilizando el recorrido en amplitud partiendo del vértice 1
c) mostrar sus vértices utilizando el recorrido en profundidad partiendo del vértice 5
d) mostrar sus vértices utilizando el recorrido en amplitud partiendo del vértice 5

11. Para un grafo almacenado en una matriz de adyacencia, desarrollar funciones para:
a) mostrar todos sus vértices mediante recorrido en profundidad
b) mostrar todos sus vértices mediante recorrido en amplitud
c) devolver la cantidad de componentes conexas

12. Idem ejercicio 11 pero suponiendo el grafo almacenado en una lista de adyacencia

13. Para el grafo del ejercicio 10:
a) encontrar el árbol abarcador de costo mínimo (AAM) mediante el algoritmo de Prim
b) encontrar el árbol abarcador de costo mínimo (AAM) mediante el algoritmo de Kruskal

14. Una empresa ferroviaria planifica el tendido de vías para unir cinco localidades determinadas. Para ello cuenta con la siguiente tabla con las distancias entre las mismas. Se solicita diseñar de la manera más eficiente dicho tendido.

	L1	L2	L3	L4	L5
L1		5	50	80	90
L2			70	60	50
L3				8	20
L4					10

15. Codificar soluciones para los algoritmos de Kruskal y de Prim, suponiendo grafos almacenados en matriz de adyacencia.

16. Calcular el camino más corto entre todos los pares de vértices, para el digrafo representado por la siguiente matriz de adyacencia, aplicando:
a) el algoritmo de Dijkstra
b) el algoritmo de Floyd

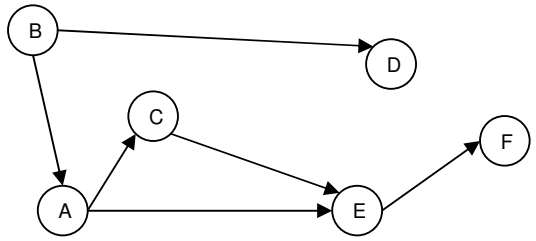
$$\begin{bmatrix} 0 & 30 & 0 & 13 \\ 22 & 0 & 10 & 12 \\ 0 & 25 & 0 & 0 \\ 13 & 6 & 0 & 0 \end{bmatrix}$$

Además, en ambos casos determinar cómo se conforma el camino mínimo

17. Para un digrafo G(V,E) la Matriz de Alcance R de define como:

$$R_{ik} = \begin{cases} 1 \text{ si } v_k \text{ es alcanzable desde } v_i. \\ 0 \text{ si } v_k \text{ no es alcanzable desde } v_i. \end{cases}$$

   a) Determinar la dimensión de R
   b) Construir R para el siguientes digrafo:



18. Para un digrafo dado, desarrollar un programa que genere la matriz de alcance R a partir de la matriz A resultante del algoritmo de Floyd.
19. Desarrollar un subprograma que determine si un vértice (v_j) es alcanzable desde otro (v_j) para un digrafo almacenado en:
   a) una matriz de adyacencia
   b) una lista de adyacencia
20. ¿Cómo modificaría el algoritmo anterior si además quisiera saber por cuántos vértices debe pasar para llegar de v_i a v_j?
21. Dada la matriz de alcance correspondiente a un digrafo, realizar una función recursiva para hallar el vértice alcanzado por la mayor cantidad de vértices. En el caso de que existan más de uno, devolver el primero encontrado.
22. Desarrollar un subprograma para mostrar el camino del vértice origen al vértice v (dato) a partir del vector P generado por el algoritmo de Dijkstra.
23. Desarrollar un subprograma para mostrar el camino de un vértice v a un vértice w a partir de la matriz P generada por el algoritmo de Floyd. (v, w datos).