

PRUEBAS

Pruebas AdjListGraphTest

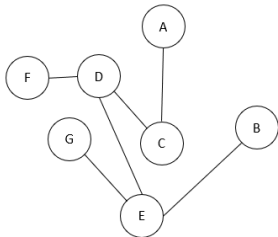
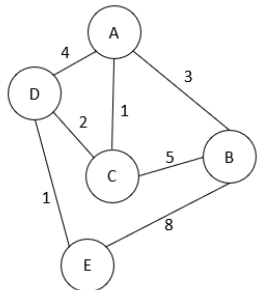
Configuración de los Escenarios

Nombre	Clase	Escenario
Setup1	AdjListGraphTest	AdjListGraph<>(false) GraphAlgorithms<>()
Setup2	AdjListGraphTest	AdjListGraph<String,String>(false) GraphAlgorithms<>()
Setup3	AdjListGraphTest	AdjListGraph<String,String>(false) GraphAlgorithms<>()
Setup4	AdjListGraphTest	AdjListGraph<>(false) GraphAlgorithms<>()
Setup5	AdjListGraphTest	AdjListGraph<>(false) GraphAlgorithms<>()

Diseño de Casos de Prueba

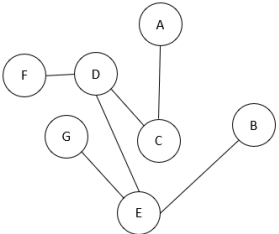
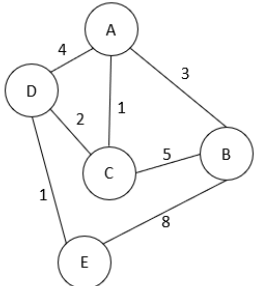
- getWeigthMatrix

Objetivo de la Prueba: Verificar que el método getWeigthMatrix de la clase AdjListGraph funcione correctamente, retornando la matriz de pesos.

Clase	Método	Escenario	Valores de entrada	Resultado
AdjListGraph	getWeigthMatrix	Setup1	graph: 	$\begin{pmatrix} 0 & \infty & 1 & \infty & \infty & \infty & \infty \\ \infty & 0 & \infty & \infty & 1 & \infty & \infty \\ 1 & \infty & 0 & 1 & \infty & \infty & \infty \\ \infty & \infty & 1 & 0 & 1 & 1 & \infty \\ \infty & 1 & \infty & 1 & 0 & \infty & 1 \\ \infty & \infty & \infty & 1 & \infty & 0 & \infty \\ \infty & \infty & \infty & \infty & 1 & \infty & 0 \end{pmatrix}$
AdjListGraph	getWeigthMatrix	Setup2	graph: 	$\begin{pmatrix} 0 & 3 & 1 & 4 & \infty \\ 3 & 0 & 5 & \infty & 8 \\ 1 & 5 & 0 & 2 & \infty \\ 4 & \infty & 2 & 0 & 1 \\ \infty & 8 & \infty & 1 & 0 \end{pmatrix}$

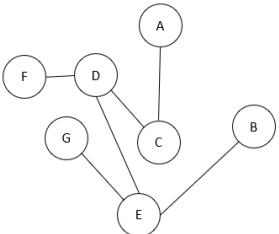
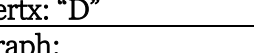
- BFS

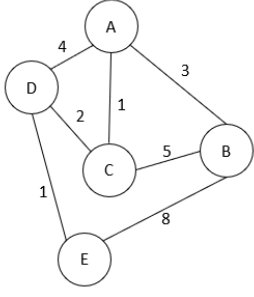
Objetivo de la Prueba: Verificar que el método BFS de la clase AdjListGraph funcione correctamente, realizando la búsqueda en anchura.

Clase	Método	Escenario	Valores de entrada	Resultado
AdjListGraph	BFS	Setup1	graph: 	"A C D F E B G"
AdjListGraph	BFS	Setup2	graph: 	"A C B D E"

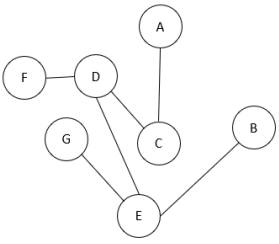
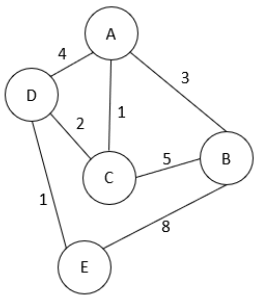
- DFS

Objetivo de la Prueba: Verificar que el método DFS de la clase AdjListGraph funcione correctamente, realizando la búsqueda en profundidad.

Clase	Método	Escenario	Valores de entrada	Resultado
AdjListGraph	DFS	Setup1	graph: 	"D C A F E B G"
AdjListGraph	DFS	Setup2	vertex: "D" graph: 	"A C B E D"

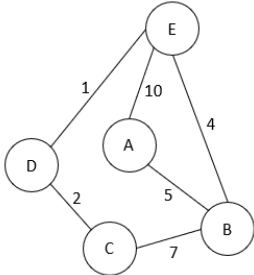
			 <p>vertex: "A"</p>	
--	--	--	---	--

- dijkstra

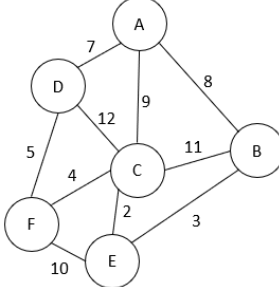
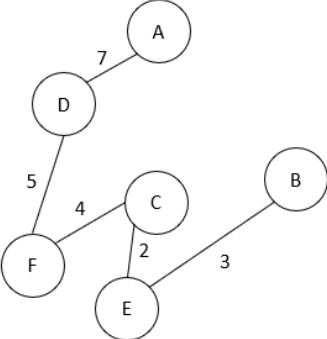
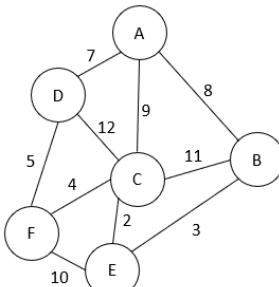
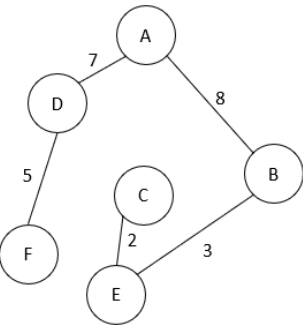
Objetivo de la Prueba: Verificar que el método dijkstra de la clase AdjListGraph funcione correctamente, determinando el camino más corto dado un vértice de origen al resto de todos los vértices.				
Clase	Método	Escenario	Valores de entrada	Resultado
AdjListGraph	dijkstra	Setup1	graph:  <p>vertex: "D"</p>	<p>"[D-C-A] [D-E-B] [D-C] [D] [D-E] [D-F] [D-E-G]"</p> <p>"[2] [2] [1] [0] [1] [1] [2]"</p>
AdjListGraph	dijkstra	Setup2	graph:  <p>vertex: "B"</p>	<p>"[B-A] [B] [B-A-C] [B-A-C-D] [B-A-C-D-E]"</p> <p>"[3] [0] [4] [6] [7]"</p>

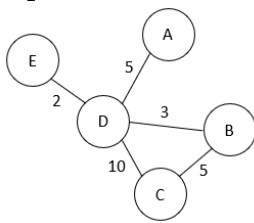
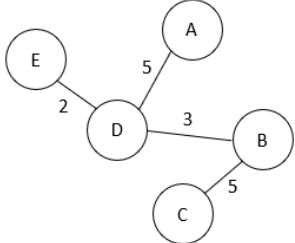
- floydWarshall

Objetivo de la Prueba: Verificar que el método floydWarshall de la clase AdjListGraph funcione correctamente, determinando el camino más corto entre todos los pares de vértices en una ejecución.				
Clase	Método	Escenario	Valores de entrada	Resultado
AdjListGraph	floydWarshall	Setup3	graph:	$\begin{pmatrix} 0 & 5 & 12 & 10 & 9 \\ 5 & 0 & 7 & 5 & 4 \\ 12 & 7 & 0 & 2 & 3 \\ 10 & 5 & 2 & 0 & 1 \\ 9 & 4 & 3 & 1 & 0 \end{pmatrix}$

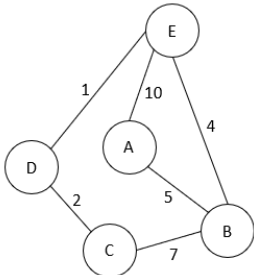
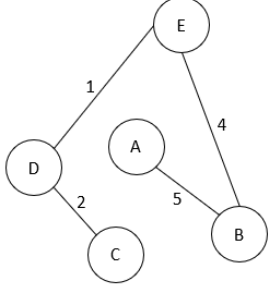
				
--	--	--	--	--

- prim

Objetivo de la Prueba: Verificar que el método prim de la clase AdjListGraph funcione correctamente, determinando el árbol recubridor mínimo dado un vértice de origen.				
Clase	Método	Escenario	Valores de entrada	Resultado
AdjListGraph	prim	Setup4	graph:  vertex: "A"	
AdjListGraph	prim	Setup4	graph:  vertex: "C"	

AdjListGraph	prim	Setup5	graph:  vertex: "A"	
--------------	------	--------	--	---

- kruskal

Objetivo de la Prueba: Verificar que el método kruskal de la clase AdjListGraph funcione correctamente, encontrando el árbol de cobertura mínimo o en su defecto el bosque de distribución mínima.				
Clase	Método	Escenario	Valores de entrada	Resultado
AdjListGraph	kruskal	Setup3	graph: 	
AdjListGraph	kruskal	Setup2	graph: 