Pruebas de clase.

Adjacent List Graph.

Sea:

PathType = PT.

isPointLight = IPL.

|  |  |  |
| --- | --- | --- |
| Nombre | Clase | Escenario |
| setupEscenary1 | AdjacentListGraph | |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | 0 |  |  |  |  | | --- | --- | | 1 |  |  |  |  | | --- | --- | | 2 |  | | |  |  | | --- | | PT = PATH.  Id = “1”  IPL = False.  ady = {p2} |  |  | | --- | | PT = PATH.  Id = “3”  IPL = False.  ady = {p1} |  |  | | --- | | PT = PATH.  Id = “2”  IPL = False.  ady = {p3} | |
| setupEscenary2 | AdjacentListGraph | |  | | --- | | PT = HOLE.  Id = “1”  IPL = False.  ady = {p2:5, p3:4} |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | 0 |  |  |  |  | | --- | --- | | 1 |  |  |  |  | | --- | --- | | 2 |  | | |  |  | | --- | | PT = PATH.  Id = “3”  IPL = False.  ady ={p1:4, p2:3} |  |  | | --- | | PT = LAKE.  Id = “2”  IPL = False.  ady = {p1:5,p3:3} | |
| setupEscenary3 | AdjacentListGraph |  |
| setupEscenary4 | AdjacentListGraph | |  | | --- | | PT = LAKE.  Id = “1”  IPL = False.  ady ={p2:p5} |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | 0 |  |  |  |  | | --- | --- | | 1 |  |  |  |  | | --- | --- | | 2 |  | | |  |  | | --- | | PT = LAKE.  Id = “9”  IPL = True.  ady ={p3:3} |  |  | | --- | | PT = LAKE.  Id = “3”  IPL = True.  ady ={p1:4} | |
| setupEscenary5 | AdjacentListGraph  PT = LAKE  Id = “7”  IPL =true.  ady ={p8:4} | PT = PATH  Id = “1”  IPL = false.  ady = {p2:3, p3:6}   |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | 0 |  |  |  |  | | --- | --- | | 1 |  |  |  |  | | --- | --- | | 7 |  |  |  |  | | --- | --- | | 6 |  |  |  |  | | --- | --- | | 5 |  |  |  |  | | --- | --- | | 4 |  |  |  |  | | --- | --- | | 3 |  |  |  |  | | --- | --- | | 2 |  | | |   Id = “V1”  ady ={V2:5}  PT = QUICKSAND  Id = “2”  IPL = true.  ady ={p3:5}  PT = WALL  Id = “3”  IPL =false.  ady ={p4:3}  PT = LAKE  Id = “8”  IPL =false.  ady = {}  PT = LAKE  Id = “6”  IPL =false.  ady ={p7:2}  PT = QUICKSAND  Id = “5”  IPL =true.  ady ={p7:14, p6:8}  PT = HOLE  Id = “4”  IPL =true.  ady= {p1:7, p2:1, p5:3} |
| setupEscenary6 | AdjacentListGraph | |  | | --- | | PT = LAKE  Id = “1”  IPL =false.  ady ={p2,p3} |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | |  |  | | --- | --- | | 0 |  |  |  |  | | --- | --- | | 1 |  |  |  |  | | --- | --- | | 2 |  | | |  |  | | --- | | PT = LAKE  Id = “3”  IPL =false.  ady ={p1} |  |  | | --- | | PT = LAKE  Id = “2”  IPL =false.  ady ={p1} | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Objetivo: Verifica que el método getIndexVertex devuelve el índice donde el vértice está guardado en la lista de vértices.. | | | | |
| Clase | Método | Escenario | Valores de entrada | Resultado |
| GraphByList | getIndexVertex | setupEscenary1 | valueVertex = p1; | La posición del vértice en la lista de vértices es 0 |
| GraphByList | getIndexVertex | setupEscenary1 | valueVertex = p2 | La posición del vértice en la lista de vértices es 1. |
| GraphByList | getIndexVertex | setupEscenary5 | valueVertex = p7 | La posición del vértice en la lista de vértices es 6 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Objetivo: Verificar que el método addEdge añade correctamente una arista dirigida y ponderada al grafo | | | | |
| Clase | Método | Escenario | Valores de entrada | Resultado |
| GraphByList | addEdge | setupEscenary4 | Vertex1 = p4;  Vertex2 = p5;  Vertex3 = p20; | Se agregó correctamente las aristas dirigidas al grafo, ahora el grafo tiene 5 vértices.  El vértice p5 es adyacente al p4, del mismo modo p20 a p5. |
| GraphByList | addEdge | setupEscenary5 | Vertex1 = p10;  Vertex2 = p11;  Vertex3 = p12; | P11 es adyacente a p10, del mismo modo p11 a p12. Pero p12 no es adyacente a p10.  Por otro lado, el grafo tiene 11 vértices y 13 aristas. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Objetivo: Verificar que el método isAdjacent retorna un booleano si dos vértices son adyacentes en el grafo. | | | | |
| Clase | Método | Escenario | Valores de entrada | Resultado |
| GraphByList | areAdjacent | setupEscenary5 | Vertex1 = p1  Vertex2 = p2 | True.  Los vértices 1 y 2 son adyacentes. |
| GraphByList | areAdjacent | setupEscenary5 | Vertex1 = p1;  Vertex2 = p4; | False.  Los vértices 1 y 4 no son adyacentes. |
| GraphByList | areAdjacent | setupEscenary2 | Vertex1 = p1;  Vertex2 = p2; | True.  El vértice 1 es adyacente al vértice 2 y a su vez el vértice 2 es adyacente al vértice 1. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Objetivo: Verifica que el método getNumberVertex retorna el número de vértices en el grafo. | | | | |
| Clase | Método | Escenario | Valores de entrada | Resultado |
| GraphByList | getNumberVertex | setupEscenary5 |  | Número de vértices son 8. |
| GraphByList | getNumberVertex | setupEscenary4 |  | Número de vértices son 3 |
| GraphByList | getNumberVertex | setupEscenary1 |  | Número de vértices son 3 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Objetivo: Verifica que el método getAdjList retorna la lista de adyacencia. | | | | |
| Clase | Método | Escenario | Valores de entrada | Resultado |
| GraphByList | getAdjList | setupEscenary4 |  | Devuelve correctamente la lista de adyacencia del grafo. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Objetivo: Verifica que el método edgeBetween da el costo de las arista entre dos vértices. | | | | |
| Clase | Método | Escenario | Valores de entrada | Resultado |
| GraphByList | getAdjList | setupEscenary5 | VertexA = p1  VertexB = p2 | Devuelve el costo de la arista cuyo valor es 3.0. |
| GraphByList | getAdjList | setupEscenary5 | VertexA = p1  VertexB = p3 | Devuelve el costo de la arista cuyo valor es 6.0. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Objetivo: Verifica que el método getIndexVertex devuelve el índice donde el vértice está guardado en la lista de vértices.. | | | | |
| Clase | Método | Escenario | Valores de entrada | Resultado |
| GraphByList | getIndexVertex | setupEscenary1 | valueVertex = p1; | La posición del vértice en la lista de vértices es 0 |
| GraphByList | getIndexVertex | setupEscenary1 | valueVertex = p2 | La posición del vértice en la lista de vértices es 1. |
| GraphByList | getIndexVertex | setupEscenary5 | valueVertex = p7 | La posición del vértice en la lista de vértices es 6 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Objetivo: Verificar que el método addEdge añade correctamente una arista dirigida y ponderada al grafo | | | | |
| Clase | Método | Escenario | Valores de entrada | Resultado |
| GraphByList | addEdge | setupEscenary4 | Vertex1 = p4;  Vertex2 = p5;  Vertex3 = p20; | Se agregó correctamente las aristas dirigidas al grafo, ahora el grafo tiene 5 vértices.  El vértice p5 es adyacente al p4, del mismo modo p20 a p5. |
| GraphByList | addEdge | setupEscenary5 | Vertex1 = p10;  Vertex2 = p11;  Vertex3 = p12; | P11 es adyacente a p10, del mismo modo p11 a p12. Pero p12 no es adyacente a p10.  Por otro lado, el grafo tiene 11 vértices y 13 aristas. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Objetivo: Verificar que el método isAdjacent retorna un booleano si dos vértices son adyacentes en el grafo. | | | | |
| Clase | Método | Escenario | Valores de entrada | Resultado |
| GraphByList | areAdjacent | setupEscenary5 | Vertex1 = p1  Vertex2 = p2 | True.  Los vértices 1 y 2 son adyacentes. |
| GraphByList | areAdjacent | setupEscenary5 | Vertex1 = p1;  Vertex2 = p4; | False.  Los vértices 1 y 4 no son adyacentes. |
| GraphByList | areAdjacent | setupEscenary2 | Vertex1 = p1;  Vertex2 = p2; | True.  El vértice 1 es adyacente al vértice 2 y a su vez el vértice 2 es adyacente al vértice 1. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Objetivo: Verifica que el método getNumberVertex retorna el número de vértices en el grafo. | | | | |
| Clase | Método | Escenario | Valores de entrada | Resultado |
| GraphByList | getNumberVertex | setupEscenary5 |  | Número de vértices son 8. |
| GraphByList | getNumberVertex | setupEscenary4 |  | Número de vértices son 3 |
| GraphByList | getNumberVertex | setupEscenary1 |  | Número de vértices son 3 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Objetivo: Verifica que el método getAdjList retorna la lista de adyacencia. | | | | |
| Clase | Método | Escenario | Valores de entrada | Resultado |
| GraphByList | getAdjList | setupEscenary4 |  | Devuelve correctamente la lista de adyacencia del grafo. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Objetivo: Verifica que el método edgeBetween da el costo de las arista entre dos vértices. | | | | |
| Clase | Método | Escenario | Valores de entrada | Resultado |
| GraphByList | getAdjList | setupEscenary5 | VertexA = p1  VertexB = p2 | Devuelve el costo de la arista cuyo valor es 3.0. |
| GraphByList | getAdjList | setupEscenary5 | VertexA = p1  VertexB = p3 | Devuelve el costo de la arista cuyo valor es 6.0. |