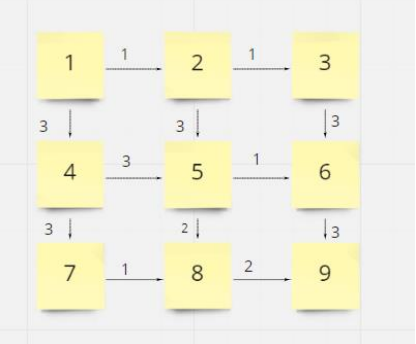
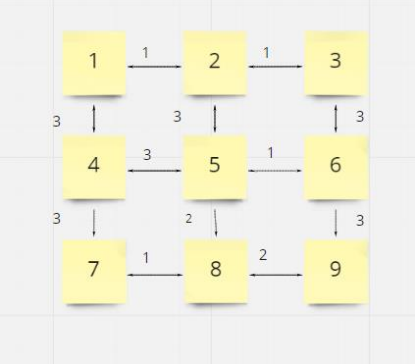


TEST CASE DESIGN

Scenarios:


Name	Class	Scenary
setUpEscenary1	Graph	
setUpscenary2	Graph	
SetUpScenary3	Graph	Empty
SetUpScenary1	Maze	Empty

Test Objective: Search the vertex in one directed graph				
Class	Method	Scenary	Input	Output
Graph	searchVertex()	setupScenary 1	Room two = new Room(2);	Vertex<V, E>(two)

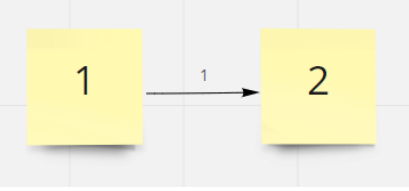
Test Objective: Use the Dijkstra algorithm to search the amount tokens to give the player				
Class	Method	Scenary	Input	Output
Graph	Dijkstra ()	setupScenary 1	Room 1	8

Test Objective: return the best way to win the game				
Class	Method	Scenary	Input	Output
Graph	getPath()	setupScenary 1	Room 1	[1,2,5,8]

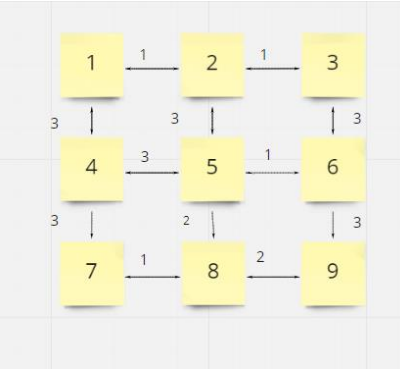
Test Objective: Use the Floyd algorithm to search the best way in any pair of vertex				
Class	Method	Scenary	Input	Output
Graph	floydWarshall()	setupScenary 1	Room 5	6

Test Objective: add the vertex in one directed graph				
Class	Method	Scenary	Input	Output
Graph	addVertex()	setupScenary 1	 <pre> Room two = new Room(2) Vertex v = new Vertex(two, listEdges); </pre>	The vertex 2 was added successfully

Test Objective: Fill matrix				
Class	Method	Scenary	Input	Output
Graph	fillMatrix()	setupScenary 1	Door d = new Door(2); Edge e = new Edge(d,1,2);	i = 1 j = 2 height =d.getData()

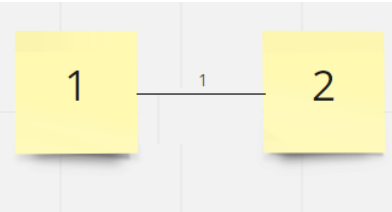
Test Objective: add the edge to one directed graph				
Class	Method	Scenary	Input	Output
Graph	addEdge()	setupScenary 1	 <pre>Door silverDoor = new SilverDoor(); Edge one = new Edge(silverDoor, 1, 2);</pre>	The edge with the height 1 was created successfully

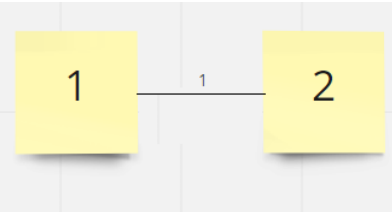
Test Objective: Search the vertex in one undirected graph				
Class	Method	Scenary	Input	Output
Graph	searchVertex()	setupScenary 2	Room two = new Room(2);	Vertex<V, E>(two)

Test Objective: Use the Dijkstra algorithm to search the amount tokens to give the player				
Class	Method	Scenary	Input	Output
Graph	Dijkstra ()	setupScenary 2		8

Test Objective: return the best way to win the game				
Class	Method	Scenary	Input	Output
Graph	getPath()	setupScenary 2	Room 1	[1,2,5,8]

Test Objective: Use the Floyd algorithm to search the best way in any pair of vertex				
Class	Method	Scenary	Input	Output
Graph	floydWarshall()	setupScenary 2	Room 2	3

Test Objective: add the vertex in one undirected graph				
Class	Method	Scenary	Input	Output
Graph	addVertex()	setupScenary 2	 <pre> Room two = new Room(2) Vertex v = new Vertex(two, listEdges); </pre>	The vertex 2 was created successfully

Test Objective: add the edge to one undirected graph				
Class	Method	Scenary	Input	Output
Graph	addEdge()	setupScenary 2	 <pre> Door silverDoor = new SilverDoor(); Edge one = new Edge(silverDoor, 1, 2); </pre>	The edge 1 with the height was created successfully

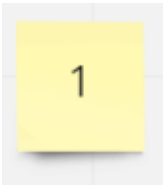
Test Objective: Fill matrix				
Class	Method	Scenary	Input	Output
Graph	fillMatrix()	setupScenary 2	Door d = new Door(2); Edge e = new Edge(d,1,2);	i = 1 j = 2 height =d.getData()


Test Objective: Search the vertex in one directed graph				
Class	Method	Scenary	Input	Output
Graph	searchVertex()	setupScenary 3	Room two = new Room(2);	Doesn't exist anything in the graph

Test Objective: Use the Dijkstra algorithm to search the amount tokens to give the player				
Class	Method	Scenary	Input	Output
Graph	Dijkstra ()	setupScenary 3	Room 1	The graph is empty

Test Objective: return the best way to win the game				
Class	Method	Scenary	Input	Output
Graph	getPath()	setupScenary 3	Room 1	The graph is empty

Test Objective: Use the Floyd algorithm to search the best way in any pair of vertex				
Class	Method	Scenary	Input	Output
Graph	floydWarshall()	setupScenary 3	Room 2	The graph is empty

Test Objective: add the vertex in one directed graph				
Class	Method	Scenary	Input	Output
Graph	addVertex()	setupScenary 3	 Room one = new Room(1) Vertex v = new Vertex(one, listEdges);	The vertex 1 was created successfully

Test Objective: add the edge to one directed graph				
Class	Method	Scenary	Input	Output
Graph	addEdge()	setupScenary 3	 <pre> Door silverDoor = new SilverDoor(); Edge one = new Edge(silverDoor, 1, 2); </pre>	The edge with the height 1 was created successfully

Test Objective: Search the vertex in one directed graph				
Class	Method	Scenary	Input	Output
Maze	createGraph()	setupScenary 1	Graph g = new Graph(15, true);	The graph was created successfully

Test Objective: Use the Dijkstra algorithm to search the amount tokens to give the player				
Class	Method	Scenary	Input	Output
Maze	addDoor()	setupScenary 1	Door door1 = new Door(1,2,1)	The door was created successfully

Test Objective: return the best way to win the game				
Class	Method	Scenary	Input	Output
Maze	addRoom()	setupScenary 1	Room room1 = new Room(1, 2)	The room was created successfully

Test Objective: Use the Floyd algorithm to search the best way in any pair of vertex				
Class	Method	Scenary	Input	Output
Maze	setTreasure()	setupScenary 1	setTreasure(3)	The treasure was put in the Room 3