



Algoritmos y Estructuras de Datos 2020-2 Tarea Integradora 3

Nombre: Jhon Sebastian Ijaji Ortiz Sebastian Villa Avila	Código: A00362423 A00361589
Profesor: Andrés Aristizabal	

Diseño de casos

SETUPS QUEUE

Nombre	Clase	Escenario
setup1	Queue	(7) (15) (53)

Nombre	Clase	Escenario
setup2	Queue	('A') ('Ñ') ('Z') ('D')

Nombre	Clase	Escenario
setup3	Queue	("perro") ("gato") ("vaca") ("pato") ("pajaro") ("mono")

Nombre	Clase	Escenario
--------	-------	-----------

setup4	Queue	VACÍO
--------	-------	-------

SETUPS MINHEAP

Nombre	Clase	Escenario
setup1	minHeap	<pre> imh = new MinHeap<Integer>(6); imh.insert(4); mh.insert(99); imh.insert(3); imh.insert(33); imh.insert(66); imh.insert(10); </pre>

Nombre	Clase	Escenario
setup2	minHeap	<pre> smh = new MinHeap<String>(6); smh.insert("Orden 1"); smh.insert("Orden 3"); smh.insert("Orden 6"); smh.insert("Orden 9"); smh.insert("Orden 4"); smh.insert("Orden 2"); </pre>

Nombre	Clase	Escenario
setup3	miHeap	<pre> cmh = new MinHeap<Character>(6); cmh.insert('A'); cmh.insert('B'); cmh.insert('C'); cmh.insert('D'); cmh.insert('E'); cmh.insert('F'); </pre>

SETUPS MATRIX GRAPH

Nombre	Clase	Escenario
Setup1	MatrixGraph	Cali, bogota, 4 Medellin, bogota, 5 Cali, Medellin, 6 España, cali, 3 Medellin, España, 2

Nombre	Clase	Escenario
Setup2	MatrixGraph	Africa, Australia, 4 Usa, africa, 3 Australia, usa, 2

Nombre	Clase	Escenario
Setup3	MatrixGraph	Canada, Texas, 4 Miami, new York 7 New, york, Canada 5 Londres, Miami 3 Australia, texas 5 Texas, Canada 7 Londres, texas 8 Londres, new York 2 Dember, londres 6 New York, dember 4

SETUPS MATRIX GRAPH

Nombre	Clase	Escenario
Setup1	ListGraph	1,3,2 3,7,4 2, 8, 6

Nombre	Clase	Escenario
Setup2	ListGraph	Cristian, maria, 4 Carlos, juan, 7 Maria, juan, 5 Carlos, cristian, 8 Maria, carlos, 2

Nombre	Clase	Escenario
Setup3	ListGraph	Y, z, 2 A, d, 4

Objetivo de la Prueba: El objetivo de esta prueba es verificar que la estructura de datos tabla Hash, realice la correcta búsqueda de un elemento, dada su clave y retornando el valor

Clase	Método	Escenario	Valores de Entrada	Resultado
Hash	get	setup1()	Ninguno	El resultado de este método es la correcta búsqueda de un elemento dada su clave, por tanto retorna la información de este elemento.
Hash	get	setup2()	Ninguno	
Hash	get	setup3()	Ninguno	

Objetivo de la Prueba: El objetivo de esta prueba es verificar que la estructura de datos tabla Hash, realice la correcta eliminación de un elementos dado su clave

Clase	Método	Escenario	Valores de Entrada	Resultado
Hash	remove	setup1()	Ninguno	El resultado de este método es la correcta eliminación de un nodo de la estructura de datos a la vez se retorna la información de este.
Hash	remove	setup2()	Ninguno	
Hash	remove	setup3()	Ninguno	

Objetivo de la Prueba: El objetivo de esta prueba es verificar que la estructura de datos tabla Hash, realice el correcto cálculo del tamaño de la tabla Hash y lo retorna				
Clase	Método	Escenario	Valores de Entrada	Resultado
Hash	size	setup1()	Ninguno	El resultado de este método es el tamaño de la estructura de datos en ese momento
Hash	size	setup2()	Ninguno	
Hash	size	setup3()	Ninguno	

Objetivo de la Prueba: El objetivo de esta prueba es verificar que la estructura de datos Queue genérica, realice la correcta comprobación si la fila está vacía o no				
Clase	Método	Escenario	Valores de Entrada	Resultado
Queue	isEmpty	setUp1()		El resultado de este método un booleano que confirme si la fila está vacía o no
Queue	isEmpty	setUp2()		
Queue	isEmpty	setUp4()		

Objetivo de la Prueba: El objetivo de esta prueba es verificar que la estructura de datos Queue genérica, realice la correcta eliminación de un nodo dada su clave				
Clase	Método	Escenario	Valores de Entrada	Resultado
Queue	dequeue	setup1()	Ninguno	El resultado de este método es la correcta eliminación de un nodo de la fila genérica y a su vez se retorna el nodo eliminado
Queue	dequeue	setup2()	Ninguno	
Queue	dequeue	setup3()	Ninguno	

Objetivo de la Prueba: El objetivo de esta prueba es verificar que la estructura de datos minHeap (cola de prioridad) intercambie dos nodos dada su posición				
Clase	Método	Escenario	Valores de Entrada	Resultado
MinHeap	swap	setup1()	Ninguno	El resultado de este método es el correcto intercambio de los nodos dado su posición en la estructura contenedora
MinHeap	swap	setup2()	Ninguno	
MinHeap	swap	setup3()	Ninguno	

Objetivo de la Prueba: El objetivo de esta prueba es verificar que la estructura de datos minHeap (cola de prioridad) reordena la estructura a partir de un índice dado.				
Clase	Método	Escenario	Valores de Entrada	Resultado
MinHeap	minHeapify	setup1()	Ninguno	El resultado de este método es la correcta reordenación de la estructura de datos a partir del índice ingresado
MinHeap	minHeapify	setup2()	Ninguno	
MinHeap	minHeapify	setup3()	Ninguno	

Objetivo de la Prueba: El objetivo de esta prueba es verificar que la estructura de datos minHeap (cola de prioridad), agrega un elemento a la cola, teniendo en cuenta su valor de prioridad				
Clase	Método	Escenario	Valores de Entrada	Resultado
MinHeap	insert	setup1()	Ninguno	El resultado de este método es la correcta inserción del elemento teniendo en cuenta su valor de prioridad
MinHeap	insert	setup2()	Ninguno	
MinHeap	insert	setup3()	Ninguno	

Objetivo de la Prueba: El objetivo de esta prueba es verificar que la estructura de datos minHeap (cola de prioridad), reordena la estructura, organizando los nodos de acuerdo a su valor de prioridad				
Clase	Método	Escenario	Valores de Entrada	Resultado
MinHeap	minHeap	setup1()	Ninguno	El resultado de este método es una estructura de datos ordenada de acuerdo al valor de prioridad de cada elemento
MinHeap	minHeap	setup2()	Ninguno	
MinHeap	minHeap	setup3()	Ninguno	

Objetivo de la Prueba: El objetivo de esta prueba es verificar que la estructura de datos minHeap (cola de prioridad), extrae el nodo con el mínimo valor de prioridad				
Clase	Método	Escenario	Valores de Entrada	Resultado
MinHeap	extractMin	setup1()	Ninguno	El resultado de este método es la eliminación de el nodo con el menor valor de prioridad
MinHeap	extractMin	setup2()	Ninguno	
MinHeap	extractMin	setup3()	Ninguno	

Objetivo de la Prueba: El objetivo de esta prueba es verificar que se haga la correcta eliminación de un vértice "lugar", en la matriz				
Clase	Método	Escenario	Valores de Entrada	Resultado
MatrixGraph	removeVetex	setup1()	Ninguno	El resultado de este método es la correcta eliminación de un vértice
MatrixGraph	removeVetex	setup2()	Ninguno	
MatrixGraph	removeVetex	setup3()	Ninguno	

Objetivo de la Prueba: El objetivo de esta prueba es verificar que se elimine correctamente una arista de la matriz				
--	--	--	--	--

Clase	Método	Escenario	Valores de Entrada	Resultado
MatrixGraph	removeEdge	setup1()	Ninguno	El resultado de este método es la correcta eliminación de un camino de la matriz
MatrixGraph	removeEdge	setup2()	Ninguno	
MatrixGraph	removeEdge	setup3()	Ninguno	

Objetivo de la Prueba: el objetivo de esta prueba es verificar la correcta eliminación de un vértice de la lista				
Clase	Método	Escenario	Valores de Entrada	Resultado
LisGraph	deleteVertex	setup1()	Ninguno	El resultado de esta prueba es la eliminación adecuada de un vertice
LisGraph	deleteVertex	setup2()	Ninguno	
LisGraph	deleteVertex	setup3()	Ninguno	

Objetivo de la Prueba: el objetivo de esta prueba es recorrer o buscar elementos en un grafo. Intuitivamente, se comienza en la raíz (eligiendo algún nodo como elemento raíz en el caso de un grafo) y se exploran todos los vecinos de este nodo.				
Clase	Método	Escenario	Valores de Entrada	Resultado
LisGraph	bfs	setup1()	Ninguno	El resultado de esta prueba la ruta mas corta entre en par de vertices
LisGraph	bfs	setup2()	Ninguno	
LisGraph	bfs	setup3()	Ninguno	