Algoritmos y Estructuras de Datos
2020-2 Tarea Integradora 1- Corrección

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

Diseño de casos

SETUPS HASH

Nombre	Clase	Escenario
setup1	Hash	("Spanish","Hola") ("English", "Hello") ("Korean", "Annyeonghaseyo")

Nombre	Clase	Escenario
setup2	Hash	(4,"Sublime") (126,"SL") (2015, "SKT T1")

Nombre	Clase	Escenario
setup3	Hash	('F',106) ('A',101) ('K',113) ('E',105) ('R',122)

SETUPS QUEUE

Nombre	Clase	Escenario

setup1	Queue	(7, "Niño") (15, "Joven") (53, "Adulto")

Nombre	Clase	Escenario
setup2	Queue	('A', 1) ('Ñ', 15) ('Z', 27) ('D', 4)

Nombre	Clase	Escenario
setup3	Queue	("perro", "firulais") ("gato", "misifus") ("vaca", "lola") ("pato", "donald") ("pajaro", "lucas") ("mono", "jorge")

Nombre	Clase	Escenario
setup4	Queue	VACÍO

SETUPS MINHEAP

Nombre	Clase	Escenario

setup1	minHeap	imh = new MinHeap<Integer>(6);
		imh.insert(4);
		imh.insert(99);
		imh.insert(3);
		imh.insert(33);
		imh.insert(66);
		imh.insert(10);

Nombre	Clase	Escenario

setup2	minHeap	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");

Nombre	Clase	Escenario

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

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 lla 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 eliminacion de el nodo con el menor valor de prioridad
MinHeap	extractMin	setup2()	Ninguno	
MinHeap	extractMin	setup3()	Ninguno	