TI 1

Specification of requirements and design

Cliente	Discreet Guys
Usuario	Empleados
Descripción	Simular el funcionamiento de los ascensores de los nuevos edificios que van a construir en el reciente lote que adquirieron cerca de la universidad ICESI.
Requerimientos funcionales	**R_01** Operar hacia dónde se dirige cada persona. **R_02** Determinar el ingreso al ascensor por orden de llegada y la salida del ascensor inversamente al orden de llegada. **R_03** Dirigir el ascensor a cada piso con base al orden en el que los usuarios pulsan el botón.
Requerimientos no funcionales	Hacer uso de las estructuras de datos (colas, pilas, colas de prioridad ,tablas hash).

Nombre o Identificador	**R_01** Operar hacia dónde se dirige cada persona.		
Resumen	Cada persona va a una determinada oficina y el sistema debe poder operar hacia dónde se dirige cada persona, teniendo en cuenta el piso y la oficina.		
Entradas	**Nombre**	**Tipo**	**Condición**
	name	String	
	floor	int	
	numOffice	int	No sea un número repetido
Resultado	Cada usuario se encuentra en la oficina a la que se dirigía.		
Salidas	**Nombre**	**Tipo**	**Condición**
	movement	String	Si las entradas son correctas, el sistema muestra los movimientos que realiza cada usuario.

Nombre o Identificador	**R_02** Determinar el ingreso al ascensor por orden de llegada y la salida del ascensor inversamente al orden de llegada.		
Resumen	El sistema debe permitir que el ingreso de las personas a los ascensores sea determinado de acuerdo con el orden de llegada al ascensor. Además, la salida de personas del ascensor debe ser inversa al orden de llegada.		
Entradas	**Nombre**	**Tipo**	**Condición**
	name	String	
	floor	int	
	numOffice	int	No sea un número repetido
Resultado	La salida de personas del ascensor es inverso al orden de llegada.		
Salidas	**Nombre**	**Tipo**	**Condición**
	List	String	Si las entradas son correctas, el sistema muestra un listado con el orden de llegada a cada oficina.

Nombre o Identificador	**R_03** Dirigir el ascensor a cada piso con base al orden en el que los usuarios pulsan el botón.		
Resumen	El sistema debe permitir que cada ascensor se dirija a cada piso con base al orden en el que los usuarios pulsan el botón.Teniendo en cuenta que obviamente dicho orden se va a ver afectado por la dirección en la que vaya el ascensor .		
Entradas	**Nombre**	**Tipo**	**Condición**
	floor	int	
Resultado	El ascensor recorre los pisos con base al orden de los botones oprimidos.		
Salidas	**Nombre**	**Tipo**	**Condición**

Requerimiento	Clase	Método	Return
R_01 Operar hacia dónde se dirige cada persona.	main.Main model.HashMap model.LinkedList	1. elevatorBuilding() 2. searchFloorArrive(int numFloors,int office,int officeB) 3. put(K key, V value) 4. size() 5. keySet() 6. values() 7. add(E e) 8. get(K key) 9. getHashCode(K key)	1. void 2. int 3. V 4. int 5. LinkedList 6. LinkedList 7. void 8. V 9. int
R_02 Determinar el ingreso al ascensor por orden de llegada y la salida del ascensor inversamente al orden de llegada.	main.Main model.HashMap model.Max_PriorityQueue model.Min_PriorityQueue model.Node	1. elevatorBuilding() 2. searchFloorArrive(int numFloors,int office,int officeB) 3. put(K key, V value) 4. size() 5. keySet() 6. add(E e) 7. get(K key) 8. peek() 9. poll() 10. MaxHeapify(int i) 11. less(E e1,E e2) 12. leftChildren(int i) 13. rightChildren(int i) 14. compareTo(E o)	1. void 2. int 3. V 4. int 5. LinkedList 6. void 7. V 8. E 9. E 10. Void 11. Boolean 12. E 13. E 14. int
R_03 Dirigir el ascensor a cada piso con base al orden en el que los usuarios pulsan el botón.	main.Main model.HashMap model.LinkedList model.Max_PriorityQueue model.Min_PriorityQueue model.Node	1. elevatorBuilding() 2. searchFloorArrive(int numFloors,int office,int officeB) 3. put(K key, V value) 4. size() 5. keySet() 6. values() 7. add(E e) 8. get(K key) 9. getHashCode(K key) 10. peek() 11. poll() 12. MaxHeapify(int i) 13. less(E e1,E e2) 14. leftChildren(int i) 15. rightChildren(int i) 16. compareTo(E o	1. void 2. int 3. V 4. int 5. LinkedList 6. LinkedList 7. void 8. V 9. Int 10. E 11. E 12. Void 13. Boolean 14. E 15. E 16. int