El sistema debe estar en la capacidad de:

Req1. ***Registrar*** un nuevo cliente con tipo de documento de identidad, núm. De documento, nombre, apellidos, teléfono y dirección. No permitirá que se agreguen usuarios con el mismo documento de identidad. La dirección y el teléfono son información opcional.

Req2. ***Asignar*** un turno a un usuario. El turno está formado por una letra y un número entre 00 y 99. Cuando se llegue al último número de una letra (el 99), se cambiará al primer número de la letra siguiente y así continuará. Por ejemplo, del A99 sigue el B00. No se permitirá asignar un turno a un usuario que ya tenga un turno y no haya sido atendido.

Req3. ***Llamar*** a la persona con el turno actual. El usuario tendrá la opción de informar si la persona fue atendida o no estaba presente. Después de la llamada, el turno será desasignado.

Diseño casos de prueba

Configuración de los Escenarios

Nombre	**Clase**	**Escenario**
setup1	CompanyTest	Una lista de objetos de tipo User vacía, users<User> = null
setup2	CompanyTest	Una lista de objetos de tipo User, users<User> = { (“cedula”, “2003560499”, ”Michael”, ”Trump”, “3105862525”, “ “), (“cedula”, “100629121”, “Laura”, “Perez”, “ ”, “ ”), (“cedula”, “111019930”, ”Nicole”, “Watson”, “ ”, “ ”)}
setup3	UserTest	Un objeto de la clase User con idType = “cedula”, id = “1110199300”, name = “Nicole”, lastName = ”Watson”, phone = “ ” y address = “ ”
setup4	UserTest	Un objeto de la clase User con idType = “cedula”, id = “5600102904”, name = “Mathias”, lastName = “Larsson”, phone = “3123456789” y address “ ”
setup5	CompanyTest	Una lista de objetos de tipo Turn, turns<Turn> = { (‘A’, 0, true, false), (‘A’, 1, false, true), (‘A’, 2, false, false)}

Diseño de Casos de Prueba

Objetivo de la Prueba: Verificar que el método addUser de la clase Company funcione correctamente si el usuario que se desea añadir no existe, asignando apropiadamente los valores de los parámetros a sus respectivos atributos.				
Clase	**Método**	**Escenario**	**Valores de Entrada**	**Resultado**
User	addUser	setup2	*setup4*	Se ha creado correctamente un objeto de la clase User con los valores pasados por parámetro al método.

Objetivo de la Prueba: Verificar que el método addUser de la clase Company funcione correctamente si el usuario que se desea añadir ya existe.				
Clase	**Método**	**Escenario**	**Valores de Entrada**	**Resultado**
User	addUser	setup2	*setup3*	No se ha creado correctamente un objeto de la clase User con los valores pasados por parámetro al método y se ha lanzado la excepción UserExistsException

Objetivo de la Prueba: Verificar que el método addUser de la clase Company funcione correctamente si no hay usuarios en el programa, asignando apropiadamente los valores de los parámetros a sus respectivos atributos.				
Clase	**Método**	**Escenario**	**Valores de Entrada**	**Resultado**
User	addUser	setup1	*setup4*	Se ha creado correctamente un objeto de la clase User con los valores pasados por parámetro al método.