Sebastian Morales

Informe de laboratorio

Aquí especifico los requerimientos funcionales y no funcionales del sistema de gestión de turnos, para ser más claro, y preciso con las funcionalidades del proyecto, aquí por tanto están establecidos los requerimientos del primer laboratorio, y de este, especificando claramente donde se usan los ordenamientos, interfaces de comparación, algoritmos de búsqueda y demás.

* + Búsqueda secuencial
  + Esta búsqueda se usó en el método en el método findUserbyID, el cual le entra como parámetro un id y un tipo de identificación, buscándome los usuarios por Id.
  + Búsqueda Binaria
  + Para llevar a cabo esta búsqueda se hizo necesario la implementación de una interfaz comparator para retornarme la posición de un turno, se usó en el método findShifthBinary().
  + Nota: El método sort() ordena los elementos de un array localmente y devuelve el array ordenado. Y el sort
* Selection y collections.sort
* Este método se uso en la clase shift Control en el método de usershiftsReport(), ordenándome los turnos en orden ascendente o descendente (collections. Sort) mediante el id, retornándome el historial de turnos de un usuario, mostrando en pantalla el código del turno, además de que, si el usuario estaba presente, o no ha sido atendido.
* Este reporte se puede imprimir por consola o en archivo de texto.
* BubbleSort y insertion
* Este ordenamiento se implementó en el método UserByShiftReport() el cual le entra como parámetro un turno, una opción ya si es por consola o texto, y un orden, el ordenamiento burbuja me ordena los usuarios mediante el apellido, ejemplo dese el A00 a A01
* Este ordenamiento por inserción se utilizó para organizar los usuarios por el apellido, pero en orden descendiente. También para mostrar el historial de turnos que han utilizado un turno, ejemplo desde el z98 al z99.
* Sort
* El método sort, se implementa mediante una interfaz comparator, que me compara dos turnos, para así a la hora de guardar el archivo o imprimir por consola, los organice.
* Sort reverse y sort normal
* Este ordenamiento se utilizó en el método de suspender a un usuario, ya que este método me ayuda a obtener el historial de turnos por cada usuario, los cuales han usado y demás, luego me ordena teniendo en cuenta el tiempo del turno y toma últimos dos, si no ha estado los últimos dos, se penalizará por 48 horas.

**Requerimientos no funcionales**

En el inicio del informe de laboratorio se especifica donde fueron usados los requerimientos 4, 5 y 6.

1. El sistema debe ser persistente, para la generación de reportes usando serializable, el formato de estos debe ser en txt, imprimiendo los reportes mencionados en los requerimientos funcionales.
2. Generar aleatoriamente los usuarios, este debe estar cargado en un archivo .txt con una cantidad determinada de usuarios, en la carpeta data del proyecto, este archivo se debe cargar para así inicializarlo, en el sistema de turnos.
3. Cuando ocurra una actualización de la fecha y hora del sistema, se deben procesar como atendidos, todos los turnos, y aleatoriamente indicar si dichas personas estaban presentes o no en el momento en que fueron llamadas.
4. En la impresión del reporte se debe llevar a cabo métodos de ordenamiento usando el método collections sort.

Imprimir reporte por id, nombre y apellido, o inverso.

1. El sistema debe llevar a cabo una búsqueda binaria para que esta me permita devolver la posición de un elemento, ya sea de un usuario que está asignado a un turno.
2. El sistema debe tener un algoritmo de ordenamiento ya sea el método burbuja, selección, este puesto en un método para organizar datos.
3. En el modelo, se debe establecer la clase fecha hora y hora, estos con sus respectivos atributos, o la librería que se implementó.
4. El sistema debe permitir que, a la hora de actualizar el sistema, La hora de este de turnos inicia siendo la misma hora del sistema (es decir, la hora del computador). Pero cuando se actualice la fecha y hora en el sistema de turnos, lo que se debería almacenar no es la nueva fecha y hora, sino la diferencia de la nueva fecha y hora respecto de la fecha y hora del computador.

Ejemplo son las 9: 44 y se actualiza a las 10:06, el sistema debe guardar la diferencia que son16 minutos, De esta manera, en cualquier momento que el usuario solicite la fecha y hora actual, dicha fecha y hora actual del sistema de turnos será calculada con base en la hora actual del computador y la diferencia almacenada.

1. El sistema debe permitir mostrar los tipos de turno para que el usuario seleccione no con la duración de este, a la hora de atenderlos automáticamente.
2. los turnos deben ir desde el primero A00; cuando llegue al turno A99, debe cambiar al B00, y así sucesivamente, letra por letra, hasta Z99.

**Requerimientos funcionales la 2**

1. El sistema debe permitir asignarle a un turno, la creación de un tipo de turno con una duración en minutos o segundos, Todo tipo de turno tiene un nombre y su duración es un entero, (un turno tiene una espera obligatoria de 15 segundos).
   1. por ejemplo: 1 minuto y medio (1.5), 3 minutos y cuarto (3.25), etc.
2. El sistema debe desplegar manualmente la hora y fecha actual del sistema, indicando año, mes, día, hora, minutos, horas, mediante una opción que permita visualizarla en el menú de opciones, actualizando sus segundos y minutos respectivamente.
3. El sistema debe permitir actualizar la fecha y hora del sistema, de dos maneras:
4. Manualmente, a través de valores indicados por el usuario, tales como el día, mes, año, hora, minutos y segundos.
5. Utilizando la fecha y hora actual del sistema de cómputo
   1. La hora debe seguir funcionando normalmente de la manera en que se actualice ya sea manual, o automática, para el funcionamiento correctamente de los turnos, la hora debe ser solamente actualizada a una mayor.
6. El sistema debe permitir generar un reporte con todos los turnos que una persona ha solicitado alguna vez, indicando el código del turno, si ya fue atendido y si la persona estaba presente cuando fue llamada para ser atendida.
7. El sistema debe suspender a una persona no ha estado presente en los últimos dos turnos, de tal manera que no pueda solicitar nuevos turnos por dos días.
8. El sistema debe generar un reporte con todas las personas que han llegado a tener un turno indicado por el usuario (por ejemplo: A07, o C58).
9. El sistema debe permitir ingresar la cantidad de personas a generar aleatoriamente, para así registrarlos en el sistema, y empezar a atenderlos.
10. El sistema debe permitir en cada operación, mostrar el tiempo que se demoro en llevar a cabo dicha operación, ejemplo cargar usuarios, atender turnos, generar reporte, atender personas, generar personas aleatoriamente.
11. El sistema debe permitir que el usuario indique cuantos turnos se generaran por cada día, para atenderlos automáticamente.
12. El sistema debe permitir que el usuario escoja de que manera desea visualizar los reportes, ya sea en consola, o en un archivo de texto.
13. El sistema debe estar en la capacidad de agregar un usuario, mediante su tipo id, número identificación, nombre, apellido, teléfono y dirección.
14. El sistema debe posibilitar que los turnos se atiendan automáticamente, teniendo en cuenta la duración y seleccionando el tipo de turno, cada turno con una duración predeterminado de 15 segundos (esta es la espera de turnos) ocurriendo así el avance de turno. Ejemplo, el usuario agrega un tipo de turno almuerzo, de duración 1,8, este pues debe esperar 1,8 segundos, y esperar 15 segundos para volver a preguntar cual tipo de turno es, y así atenderse automáticamente.