Diseño y Programación Orientados a Objetos
1er. Semestre 2023
Tarea 1: Simulando una Alarma Domiciliaria

Lea detenidamente la tarea. **Si algo no lo entiende, consulte. Si es preciso, se incorporarán aclaraciones al final**. Esta interacción se asemeja a la interacción entre desarrolladores y clientes cuando algo no está del todo especificado.

1. Objetivos de la tarea

- Modelar objetos reales como objetos de software.
- Ejercitar la creación y extensión de clases dadas para satisfacer nuevos requerimientos.
- Reconocer clases y relaciones entre ellas en código fuente Java.
- Ejercitar la compilación y ejecución de programas en lenguaje Java desde una consola de comandos.
- Ejercitar la configuración de un ambiente de trabajo para desarrollar aplicaciones en lenguaje Java, se puede trabajar con un editor tipo “Sublime” o con un IDE. IntelliJ es el IDE sugerido y será usado también en la tarea 2.
- Ejercitar la entrada y salida de datos en Java.
- Manejar proyectos vía GIT (voluntario para esta tarea).
- Conocer el formato .csv y su importación hacia una planilla electrónica.
- Ejercitar la preparación y entrega de resultados de software (creación de makefile, readme, documentación).
- Familiarización con desarrollos "iterativos" e "incrementales" (o crecientes).

2. Descripción General

Esta tarea busca practicar la orientación a objeto en un sistema de alarma domiciliaria.
En esta tarea usted modelará y programará los elementos sensores (magnético, infrarrojos y de humo) y de alerta (sirena) presentes en una alarma simple. Para esta tarea, todo sensor se caracteriza por contar con un interruptor, el cual está normalmente cerrado y se abre ante activación. Los elementos de alertas informan cuando el sistema se seguridad fue violada, habiendo más de una forma para entregar esa información (sonido). Además, los sistemas de alarmas cuentan con una central que controla la operación general del sistema. Aquí usted puede revisar las bases de un sistema de seguridad domiciliario. Vea aquí qué es y cómo funciona un sistema de seguridad de casa similar al de la Figura 1.

Figura 1: Elementos de un sistema de seguridad domiciliario[1]

[1] Tomado de https://www.security.org/home-security-systems/what-is-a-home-security-system/

Por simplicidad, en esta tarea participan sólo algunos de esos elementos, todo será visto desde la perspectiva de la alarma; es decir, ésta ve sólo puertas, ventanas, y detectores PIR sin distinguir la ubicación de éstos. Ver Figura 2.

Figura 2: Modelo simplificado de puerta, ventana y sensores PIR

2.1. Sensor magnético

Un sensor magnético es usado normalmente para detectar la apertura de puertas y ventanas. En este uso, su aspecto se ve aquí. Cuando el imán (en verde en Figura 2) se aleja del interruptor (en rojo en Figura 2), éste abre el circuito entre sus terminales. Cuando está suficientemente cercano, el interruptor cierra el circuito. En su instalación en puertas y ventanas, el imán está adosado a la parte movible y el interruptor a la parte fija. Así, al abrir la puerta o ventana, éstas alejan el imán del sensor, el cual cambia de interruptor cerrado a abierto. En esta tarea, este sensor será usado en puertas y ventanas, de manera que ante la apertura de una puerta o ventana por un usuario, ésta cambiará el estado del sensor. Lo haremos así pues es más complejo modelar el cambio del interruptor como consecuencia de la ausencia del campo magnético debido al alejamiento del imán cuando un usuario abre la puerta o ventana.

2.2. Detector de movimiento

Un sensor infrarrojo pasivo (PIR) es usado normalmente para detectar movimiento. Este sensor cuenta con un sistema detector de infrarrojo y un interruptor el cual se abre cuando detecta un cambio no gradual en la señal infrarroja recibida[2]. Como resultado, permite detectar el movimiento de personas dentro de un cierto ángulo y distancia (Ver área de color naranja en Figura 2). Normalmente son instalados en las esquinas de las habitaciones. En esta tarea, cuando una persona ingresa a una habitación, ésta informará a su detector de movimiento la presencia de la persona y su ubicación en la habitación.

2.3. Sirena

Una sirena es utilizada para alertar vía un sonido a las personas y vecinos cuando el sistema se **seguridad** ha sido transgredido. Ésta está ubicada normalmente fuera de la casa. La sirena suena a petición de la central una vez que ésta identifica que el circuito de alguna de sus zonas ha sido abierto cuando la alarma está activada (o “armada”).

2.4. Central y teclado

La central es la unidad de control de la alarma. Normalmente dispone de un teclado y un display alfanumérico. En esta tarea, el teclado de su computador permite armar y desarmar la alarma. El armado permite activar sólo el perímetro o la totalidad de los sensores. Notar que para armar una alarma, todas sus zonas deben estar cerradas. Al armar la alarma, ésta responde enviando un mensaje de éxito o los números de las zonas no cerradas.

Una central típicamente puede manejar tipo 4 a 8 zonas. Así un conjunto de sensores se conecta de manera serial a una zona y la central los distingue separadamente de otra zona. Esto permite que por la noche, se active sólo las puertas y ventanas de manera que no disparar la alarma cuando un detector PIR detecte la circulación de ocupantes de la casa. Se puede definir una zona como el perímetro del 1er piso, otra el del segundo piso, otra los sensores PIR, otra la puerta principal, etc. En esta tarea consideraremos tres zonas (etapa 4 de la tarea) puerta principal, otras puertas y ventanas perimetrales, y detectores PIR.

---

[2] Todo cuerpo emite radiación la cual es mayor si su temperatura es mayor. Esto explica por qué al calentar un objeto se pone de color rojo.

2.5. Funcionamiento de la aplicación

En esta tarea su programa leerá la configuración del sistema (puertas, ventanas, sensores, sirena, central, etc.) desde un archivo y a través del teclado del computador el usuario simulará el ingreso y movimientos en la propiedad. En un archivo de salida el programa registrará los eventos del usuario y la alarma.

El formato para el archivo de entrada es rígido y el nombre del archivo es pasado como un argumento al ejecutar el programa. La primera línea define el número de puertas, ventanas y detectores de movimiento. Todas las puertas y ventanas están cerradas al crearse. La primera puerta creada por su programa es la principal. Todas las puertas y ventanas incluyen un sensor magnético asociado a la zona que corresponda. Luego se lista la ubicación, orientación y rango de cada uno de los detectores PIR, finalmente se define el nombre del archivo con el sonido de la sirena.
Su formato es:
<#_doors> <#_windows> <#_PIRs>
<x> <y> <direction_angle> <sensing_angle> <sensing_range>
...
<siren_sound_file>

La primera línea es única. Luego vienen tantas líneas como detectores PIR, cada una indica la posición y la orientación de su área de detección [º], en ángulo del cono sensor [º] y el rango de detección [m]. La ubicación de los detectores PIR están referidos a un plano R^2 usando metros como unidad de cada eje de ordenadas. La primera puerta creada será la puerta principal y debe ser asociada a la zona 0. Las otras puertas y ventanas a la zona 1 y los sensores PIR a la zona 2. La última línea contiene el nombre del archivo a reproducir por la sirena en señal de alerta.

Un ejemplo para el contenido de este archivo es:

```
2       1       3
0.0     0.0     45      60      5
10.0    4.0     225     90      4
5.0     5.0     135     30      5
siren.wav
```

Una vez creados los objetos a partir del archivo de configuración, el programa acepta los siguientes comandos ingresados por teclado:
<comando> <parameter>
Donde el valor ingresado puede ser:
k <a | p | d> // acción sobre el teclado para a: armar todo (**a**ll), p: armar **p**erímetro d: desarmar.
di <o | c> // con i en 0..(#_doors-1), por ejemplo d0 es la puerta principal. o: open, c: close
wi <o | c> // con i en 0..(#_windows-1), o: open, c: close
c <x> <y> // así se crea una nueva persona en la ubicación indicada.
pi < ↑ | ↓ | → | ←> // con I en 0..#person-1, cada flecha desplaza la persona en 0.5 [m]
// en esa dirección (norte, sur, este, oeste).

Para el archivo de entrada mostrado, un ejemplo de entrada sería:
k a
c 4.0 4.0 // crea una persona (p0 por ser la primera) en coordenadas (4, 4)
p0 ←
p0 ←
p0 ← // en éste o en un paso anterior debería sonar la alarma por PIR0

Llamando a su programa AlarmTest.java, su ejecución sería:
$ java AlarmTest config.txt

Además de los mensajes a pantalla de la central y el sonido de la sirena, su programa debe enviar a un archivo de salida (output.csv) el estado de cada sensor (1 cerrado), central (1 armada), sirena (1 sonando) y posición de personas inicial del programa y luego de cada comando ingresado por el usuario. La primera línea es de encabezado de dada columna (excepto las personas).

Step	d0	..di	w0	.. wi	Pir0	..Piri	Siren	Central	
0	1	..1	1	..1	1	.. 1	0	0	
1	1	..1	1	..1	1	.. 1	0	1	
2	1	..1	1	..1	1	.. 1	0	4.0	4.0
3	….								

3. Desarrollo en Etapas

Para llegar al resultado final de esta tarea usted debe aplicar una metodología de desarrollo "Iterativo y creciente (o incremental)" para desarrollo de software. Usted y su equipo irán desarrollando etapas donde los requerimientos del sistema final son abordados gradualmente. En cada etapa usted y su equipo obtendrá una solución que funciona para un subconjunto de requerimientos finales o bien es un avance hacia ellos. En AULA se dispondrá el recurso para subir la solución correspondiente a cada etapa del desarrollo. **Su equipo deberá entregar una solución para cada una de las etapas** aun cuando la última integre las primeras. **El readme y archivo de documentación deben ser preparados solo para la última etapa. Prepare y entregue un makefile para cada una de las etapas.** Esto tiene por finalidad, educar en la metodología de desarrollo iterativo e incremental.

3.1. Primera Etapa: Propiedad con una puerta y una ventana (sin central ni sirena)

En esta primera etapa se deben crear las clases Sensor, MagneticSensor, Door y Window. Cree el programa Stage1.java para crear los objetos a partir de la definición hecha en el archivo de entrada con valor 0 para el número de detectores PIR. La última línea del archivo de entrada se puede omitir (no hay sirena).
Verifique que la interacción con el usuario genera una salida acorde.
Entregue las clases de esta etapa, incluya su makefile, el archivo de entrada usado y el de salida generado.

Use y/o complete el código inicial para esta etapa. Usted no está obligado(a) a usar estos códigos. Están para su conveniencia y en caso de que le resulten útiles. Otras formas de estructurar el resultado hasta llegar a la etapa 4 también son válidos.

3.2. Segunda Etapa: Propiedad con dos puertas, dos ventanas, central y sirena

A las clases de la etapa previa incorporar la clase Central y Siren, las cuales siguen la lógica descrita para ellas previamente. Como archivo de sonido de la sirena, puede usar éste. Cree el programa Stage2.java para esta etapa y verifique que la interacción con el usuario genera una salida acorde.
Entregue todas las clases de esta etapa, incluya su makefile, el archivo de entrada usado y el de salida generado.
Use y/o complete el código inicial para esta etapa.

3.3. Tercera Etapa: Agrega 1 detector PIR y hasta una persona circulando

Cree la clase PIR_Detector y Person. Esta última tiene su posición (x,y) como únicos atributos, responde a los desplazamientos de 0,5 [m] en las cuatro direcciones principales (norte, sur, este, oeste). Como en la etapa previa, la central no permite aún armar sólo los sensores del perímetro.
Cree el programa Stage3.java para esta etapa y verifique que la interacción con el usuario genera una salida acorde.
Entregue todas las clases de esta etapa, incluya su makefile, el archivo de entrada usado y el de salida generado.

3.4. Cuarta Etapa: Como la etapa previa incorporando varios detectores PIR, más de una persona y la posibilidad de armando nocturno.

En esta se da cumplimiento a la totalidad de la especificación de la tarea.

3.5. Extra-crédito: Permitir la salida y entrada con armado y disparo retardado

Modifique la clase Central de manera que una vez armada dé un tiempo (a definir en constructor, suponga 5 [s]) para salir por la puerta principal sin disparar la alarma. Igualmente, al ingresar por la puerta principal, la Central debe dar un tiempo para desarmarla.
Esta parte es voluntaria, su desarrollo otorga 10 puntos adicionales (la nota final se satura en 100). Si desarrolla esta parte, indíquelo en su documentación y, además de la cuarta etapa, agregue un escenario donde se aprecie el efecto de este cambio.

4. Elementos a considerar en su documentación

Entregue todo lo indicado en "Normas de Entrega de Tareas".
Prepare un archivo makefile para compilar y ejecutar su tarea en aragorn con rótulo "run". Además, incluya rótulos "clean" para borrar todos los .class generados. Los comandos a usar en cada caso son:

```
$ make    /* para compilar */
$ make  run  /* para ejecutar la tarea */
$ make clean  /* para borrar archivos .class */
```

En su archivo de documentación (pdf o html) incorpore el diagrama de clases de la aplicación (etapa 4).