




PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACIÓN

IIC2233 - Programación Avanzada (I/2016)
Tarea 4

1. Objetivos

- Aplicar conocimientos de simulación síncrona.
- Aplicar conocimientos de testing para verificar funcionalidades en un programa.
- Crear un programa con un buen manejo de excepciones.

2. Introducción

Un día te encontrabas escribiendo muchas consultas en LCJ para saber cuál equipo pokemon era el mejor cuando miras la ventana de tu progra-cueva y  descubres que el mundo está siendo atacado por un

APOCALIPSIS ZOMBIE!!

El mundo está cerca de llegar a su fin. John Router deja de lado su querido LCJ y se enfrenta a los zombies, mientras que Chris Rivers desespera y, sin saber qué hacer, utiliza su querida colección de tazas para defenderse¹.

Sin embargo, tú no te rindes (todavía) y gracias a tus grandes conocimientos de simulación, quieres saber si existe alguna posibilidad de sobrevivir, o si simplemente debes asumir que tu destino será el mismo que el de las tazas de Chris.

3. Problema

Para esta tarea, deberás implementar una simulación síncrona de una eventual apocalipsis zombie. Para ello, deberás tomar en cuenta muchos sucesos que podrían ocurrir en un evento de este tipo y cómo las personas reaccionarían a él. En particular, interesa observar cómo va oscilando la población de personas sanas y la de zombies, cómo se mueven por el mundo y cómo son las interacciones entre ellos (zombies con zombies, personas con personas y zombies con personas).

Al iniciar la simulación, debes preguntar por consola el tiempo de simulación, el número inicial de personas y de zombies y las tasas de llegada de zombies y humanos.

¹ RIP tazas.

4. Desastre

El desastre comienza con un experimento que sale mal e infecta a una cierta cantidad de personas, quienes desarrollan un inusual apetito por otras personas, extendiendo dicha infección a través de su mordida. Estos son los primeros zombies que azotan a la población.

Los zombies se expanden rápidamente por el mundo atacando a personas desprevenidas (no se veían tan "muertos" al principio) por lo que la sociedad colapsa y los únicos sobrevivientes son personas que pueden estar solas o en grupos, los que pueden ir desde una pareja de sobrevivientes hasta una pequeña comunidad. A continuación, se detalla el comportamiento de estos sobrevivientes y de los zombies que intentan comerlos.

4.1. Personas

Las personas son los sobrevivientes del apocalipsis inicial que se baten diariamente entre la vida y la muerte por sobrevivir en un mundo que, al parecer, no los quiere para nada.

Para poder sobrevivir, estas personas se mueven por el mundo evitando o eliminando a los zombies que se encuentran mientras buscan víveres y útiles (como armas y herramientas) que les ayuden a extender su corta esperanza de vida. Una persona se transformará en zombie únicamente si uno de estos muerde directamente a la persona sana (se transforma al instante). Cualquier otro tipo de ataque entre zombies y personas se traducirá en una disminución del nivel de vida de la persona o zombie hasta que llegue a 0 y muera.

Las acciones que las personas ejecutan están afectadas por sus distintas características físicas y psicológicas que se detallan a continuación.

- Velocidad: Determina qué tan rápido se mueve una persona por el mundo.
- Vida: Una abstracción de la resistencia a la muerte que tiene una persona. Esta puede verse afectada por distintos factores, los cuales pueden extenderla o acortarla, como por ejemplo un parche puede aumentarla y un golpe puede disminuirla. Al llegar la vida de una persona a cero, esta muere.
- Habilidad de combate cuerpo a cuerpo: Determina qué tan apto es en el combate cuerpo a cuerpo una persona. Este es importante a la hora de ser atacado por los zombies, determinando qué tan probable es que la persona evite ser mordida y su habilidad para matar a un zombie con sus manos, entre otras cosas.
- Habilidad con armas: Determina la habilidad en el uso de armas de una persona. Esto afecta su puntería y, a través de ella, la cantidad de daño que inflige a un zombie la persona cuando esta armada. Esta habilidad se divide en dos para los dos tipos de armas.
 - Habilidad con armas de corto alcance: Una mayor habilidad permite a la persona hacer más daño con cada golpe y que estos sean más seguros.
 - Habilidad con armas de largo alcance: Una mayor habilidad permite a la persona hacer más daño con cada bala y manejar mejor la munición.
- Personalidad: Las personalidades de una persona determinan su actuar ante las distintas situaciones que enfrentará en el apocalipsis. Estas son detalladas en la sección 4.1.1. Una persona puede tener más de una de estas personalidades pero no tendrá personalidades contrapuestas (como valiente y cobarde).
- Carisma: Habilidad de relacionarse con otras personas o grupos. Determina qué tan bien resultan las relaciones con las demás personas. Alguien con alto carisma tiende a relacionarse con los demás, alguien con bajo carisma tiende a pelear con los demás.
- Cortes: Heridas que tiene la persona. Una persona con cortes perderá vida en el tiempo dependiendo de la gravedad de este. Un corte muy grave podría infectarse y enfermar a la persona.

- **Enfermedad:** Una persona enferma tendrá una limitación de sus habilidades, lo que reducirá sus posibilidades de sobrevivir. Cuanto más grave sea la enfermedad, peor será el efecto sobre las habilidades de la persona, pudiendo incluso reducir su velocidad a cero. La enfermedad puede ir empeorando o mejorando dependiendo de la gravedad y del uso de objetos, como un botiquín.

4.1.1. Personalidades

- **Valiente:** Las personas valientes tienden a enfrentar los peligros en vez de huir de ellos. Ante la presencia de zombies, prefieren luchar contra ellos.
- **Cobarde:** Contra personalidad de valiente. Una persona cobarde tiende a escapar apenas ve peligro. Ante la presencia de zombies preferirían abandonar a su grupo a tener que luchar contra uno.
- **Precavido:** Intenta formar grupos y conseguir armas antes de enfrentarse a los zombies. Si se encuentra solo se esconde en edificios.
- **Descuidado:** Contra personalidad de precavido. La formación de grupos y obtención de armas no es su prioridad. Viaja por el mundo sin preocuparse mucho de su situación.
- **Racional:** Sus decisiones toman en cuenta las distintas condiciones en las que se encuentra. Si está en un grupo grande y hay pocos zombies prefiere luchar. En cambio, si la cantidad de zombies es mayor, prefiere escapar. Si tiene mucha munición y está en grupo, tiende a compartirla con los demás.
- **Estúpido:** Contra personalidad de racional. Toma decisiones sin medir consecuencias. Puede decidir ir a pelear contra miles de zombies a pesar de estar solo, y escapar al ver uno aunque esté en un grupo.

Como podrás notar, algunas personalidades pueden generar un conflicto. Por ejemplo, un cobarde inteligente podría escapar o no al verse en un grupo grande contra un grupo pequeño de zombies. La personalidad que prima sobre las demás queda a su criterio. Deben especificar este criterio en su README.

4.1.2. Interacción con personas

Al encontrarse dos grupos de personas (para esta interacción una persona sola es un grupo de una persona) estas interactuarán entre ellas. Las posibles interacciones y sus implicancias son:

- **Intercambio:** Los grupos intercambian objetos en base a cuales tienen y cuales necesitan.
- **Pelea:** Los grupos entablan combate para tomar posesión de los objetos del otro.
- **Unión:** Los grupos se unen en un grupo más grande.

La interacción que ocurra dependerá del tamaño de los grupos, el nivel de carisma de los integrantes, sus personalidades, donde se encuentran, etc. y queda a su criterio. Deben especificar este criterio en su README.

4.2. Zombies

- **Velocidad:** Determina qué tan rápido se mueve un zombie por el mundo.
- **Vida:** Una abstracción de la resistencia a la re-muerte que tiene un zombie. Esta va disminuyendo gradualmente si el zombie no se alimenta (mucho más lento que la de una persona) o si es atacado por un humano.
- **Mordida:** Determina qué tan buen mordedor es un zombie. Cuanto mayor sea este valor, más posibilidades de morder a una persona tendrá el zombie.
- **Sentidos:** Determina la habilidad de un zombie para detectar a una persona cercana. Cuanto mayor sea este valor el zombie podrá detectar personas más lejanas para ir a comerlas.

- Garras: Los zombies no solo atacan con sus dientes, también con el movimiento (inconsciente) de sus brazos. Un zombie con más garras tendrá una mayor posibilidad de infligir cortes en las personas que ataque, junto con la posibilidad de que estos sean más graves.

4.3. Edificios

Durante la simulación, las personas pueden decidir si entrar o no a un edificio que les dará ciertos beneficios y algún tiempo zombies-free. Una vez que se entre a un edificio, el grupo o persona no debe estar visible para los zombies ni en la ventana de la simulación, ya que se encuentran dentro de un edificio.

Para esconder a una persona, pueden utilizar el comando `entity.hide()`, y para que vuelva a aparecer, una vez terminado su tiempo dentro del edificio, pueden utilizar `entity.show()`.

Si un grupo completo quiere ingresar a un edificio y no se puede ya que superarían la capacidad máxima, queda a su criterio si el grupo se dividirá y entrarán los que tengan una personalidad X, o si prefieren mantenerse unidos y no entrar. Debes especificar en el README cómo fue abordado este tema.

A continuación, se especifican los tipos de edificio.

4.3.1. Safe Zones

Son una cantidad determinada de edificios que proporcionan a las personas un lugar seguro y zombies-free. Sin embargo, estos tienen una capacidad limitada por lo que cada persona o grupo puede permanecer allí un tiempo determinado, sin excepción. La capacidad y el tiempo que proporciona cada Safe Zone viene especificado en el mapa que deben cargar (más información de esto en la sección 6). Una vez que una persona o grupo salen del edificio, no podrán volver a entrar hasta que haya transcurrido un tiempo mayor o igual a $5x$, en donde x es el tiempo que proporciona cada Safe Zone. Notar que no existe un límite de veces que se puede entrar a este tipo de edificios.

4.3.2. Gun Shops

Corresponden a las tiendas de armas que proporcionan a las personas del armamento necesario para defenderse del apocalipsis. Hay Gun Shops que venden armas de corto alcance y otras que venden de largo alcance. Ninguna tienda vende los dos tipos de armas.

Al igual que las Safe Zones, estas tienen una capacidad máxima de personas y una cantidad limitada de armas, por lo que si estas se acaban, la tienda debe dejar de existir en la simulación. La entrega de armas funciona con un sistema de colas, en donde se entrega un arma al primero de la cola, que corresponde al primero en llegar. Además, las armas tienen una cantidad de balas específica (debes determinarla tú), por lo que cuando se agoten, debes eliminar el arma de la persona y de la simulación.

Dado el nivel de apocalipsis existente en el mundo, las armas se regalan y no es necesario que la persona cuente con dinero para adquirir una².

5. Interfaz Gráfica

Adicionalmente, tiene a su disposición el módulo gui con el que podrá ver a través de una interfaz gráfica el comportamiento de su simulación.

En el modulo gui se encuentran los siguientes métodos:

² No es como si pudieran usar el dinero en algo, dada la situación (asumamos que la estupidez humana no llega a tal nivel).

gui.run Este método recibe una función y un int que representa un tiempo en milisegundos. Una vez llamado, se mostrará la gui y se repetirá la función constantemente. Esta función puede corresponder a un tick.

gui.add_entity Este método agrega una entidad a la interfaz.

gui.init Este método inicializa la interfaz y debe ser llamado antes de todos los demás métodos.

5.1. Entity

Clase base que representa los objeteos en la interfaz grafica y se encuentra en el package gui.entities. Posee los siguientes métodos:

__init__(path) En el init se recibe el path de la imagen png que va a corresponder a la entidad.

add_decoration(path) Agrega una decoración sobre la imagen de la entidad. Esta decoración rota junto con la entidad. Si el path es None, quita la decoración.

height() Retorna la altura en pixeles que tiene la entidad.

width() Retorna el ancho en pixeles que tiene la entidad.

y las siguientes properties:

angle Ángulo en grados en los que va a estar rotada la imagen y decoración de la entidad.

cord_x Representa la cordenada horizontal en la que se encuentra la entidad (las coordenadas parten desde la esquina superior izquierda de las entidades).

cord_y Representa la coordenada vertical en la que se encuentra la entidad.

5.1.1. Human

La clase Human hereda de Entidad, por lo que posee los mismos métodos y properties que esta. Lo que cambia es su método **init**. El init de la clase Human recibe el tipo de personalidad (que será representada con diferentes colores) y el tipo de arma que lleva. Opcionalmente, puede recibir los parámetros **cord_x** y **cord_y** para indicar su ubicación inicial. Además, tiene un método **change_weapon(type)** que cambia el arma de la entidad. Por otra parte, esta clase tiene dos atributos adicionales: **personality** y **weapon**

Las personalidades disponibles en esta clase son: Human.BRAVE, Human.COWARD, Human.CARELESS, Human.CAUTIOUS, Human.RATIONAL y Human.STUPID. Los tipos de arma son: Human.WEAPON_LONG, Human.WEAPON_SHORT y Human.NO_WEAPON.

5.1.2. Otras entidades

Adicionalmente, se tienen las entidades Zombie, Bullet, Building y GunShopLong y GunShopShort. Estas entidades son idénticas a la clase Entidad, es decir, no tienen ningún método ni consideración adicional. Los Buildings corresponden a las Safe-Zones, y las tiendas de armas largas y cortas son GunShopLong y GunShopShort, respectivamente.

5.1.3. Debugging

En el módulo entities hay una variable llamada **_debugging**. Cuando se le cambia el valor a True, se pueden ver los contornos de las entidades (por lo general, son más grandes que la entidad para que no desaparezcan al rotar). Siéntanse libres de modificar **este, solo este y nada más que este** atributo de la gui para poder debuggear mejor sus programas.

5.2. Cómo usar estas clases

La idea de estas clases es que ustedes **hereden** de ellas y las extiendan, de forma que puedan darles el comportamiento deseado. Adicionalmente, pueden crear sus propias entidades si así lo estiman conveniente (heredando de la clase Entidad), sin embargo, no es requisito que agreguen ninguna entidad adicional.

6. Archivo Mapa

Se les entregará un archivo mapa.csv que contendrá las especificaciones de donde se deben posicionar los distintos edificios junto con sus características. La primera columna corresponde a la coordenada x, la segunda a la y, la tercera al tipo de edificio (si es Safe Zone o Gun Shop, y en el caso del segundo, diferenciando el tipo de armas que venden), la cuarta la capacidad del edificio y la última el tiempo máximo que puede estar una persona si es una Safe Zone o el stock de armas que posee el Gun Shop.

Por ejemplo:

Archivo	X	Y	TIPO	CAPACIDAD	TIEMPO/CANTIDAD ARMAS
Linea 1	10	40	safe_zone	10	8
Linea2	30	50	gun_shop_short	20	50
Linea3	40	10	gun_shop_long	10	40

Nota: Pueden existir múltiples edificios de cada tipo en el mapa dado, y se probará su tarea con archivos distintos a los entregados. Sin embargo, puede asumir que los edificios nunca se sobrepondrán.

7. Estadísticas

Se debe definir un tiempo de duración para la simulación, un número de personas inicial y un número de zombies inicial. Una vez terminada la simulación, se deben mostrar las siguientes estadísticas:

- Número de Zombies Iniciales/Finales
- Número de Personas Iniciales/Finales
- Número de Conversiones a Zombies
- Número de Muertes de Zombies
- Número de Muertes de Personas
- Tasa de Conversión y Tasa de Muertes por minuto
- Resultado Final... ¿Apocalipsis o no Apocalipsis?³

8. Excepciones

Se espera que se manejen todas las excepciones correctamente (i.e: indicando el tipo de excepción a capturar y lo que se hará con ella. Una excepción genérica no se considerará como 100 % correcta, a menos que sea la única solución posible - explicar en el README.md por qué es la única solución posible).

³ Entenderemos por Apocalipsis la eliminación completa de humanos. Si queda un sobreviviente, todavía hay esperanza... ¿no es así?

9. Testing

Al menos un 50 % de todas las funciones que implemente deben estar testeadas (su programa debe ser modular, i.e: no tener una función que realice más tareas de las necesarias cuando se puede separar en dos funciones - el objetivo general es la reutilización de estas).

10. Notas

Como podrás haber notado, la descripción de las habilidades de las personas y zombies es muy general. Esto es así ya que queda a tu criterio definir el nivel de efecto que estas tienen sobre la simulación. Idealmente, manténgase dentro de efectos racionales. Por ejemplo, una persona muy enferma no debería poder sobrevivir a un encuentro con un zombie y debería tener pocas probabilidades de escapar.

11. Parámetros Iniciales

El programa deberá preguntar antes de iniciar la simulación las tasas de llegada de zombies y humanos, además del número inicial de zombies y humanos. Las tasas deberán ser especificadas en entidades por minuto. Los arribos se producirán con una distribución exponencial.

Además, el programa deberá recibir la duración total de la simulación. Luego de que esta termine, debe imprimir las estadísticas en consola y guardarlas en un archivo de texto.

12. Bonus

Se otorgará una bonificación a criterio del ayudante según la calidad de la simulación y representación fiel de lo que ocurre en el Apocalipsis.

Hint: Si ocurre algún daño físico, se ve sangre, etc. Para esto, cuentan con el método:

```
entidad.agregar_decoracion(path)
```

13. Restricciones y alcances

- Tu programa debe ser desarrollado en Python 3.4
- Esta tarea es estrictamente individual y está regida por el Código de Honor de la Escuela: Clickear para Leer.
- Su código debe seguir la guía de estilos PEP8
- Si no se encuentra especificado en el enunciado asuma que el uso de cualquier librería Python está prohibido. Pregunte por foro si se pueden usar librerías específicas.
- El ayudante puede castigar el puntaje⁴ de tu tarea si le parece adecuado. Se recomienda ordenar el código y ser lo más claro y eficiente posible en la creación de algoritmos.
- Debe adjuntar un archivo `README.md` donde se comenten los alcances y el funcionamiento de su sistema (*i.e.* manual de usuario) de forma *concisa* y *clara*.
- Cree un módulo para cada conjunto de clases. Divídalas por las relaciones y los tipos que poseen en común.
- Cualquier aspecto no especificado queda a su criterio, siempre que no pase por encima de ningún otro.

⁴ Hasta -5 décimas.

14. Entrega

- **Fecha/hora:** 18 de Mayo - 23:59 hrs
- **Lugar:** GIT - Carpeta: Tareas/T04

No subir la carpeta Ejemplos. Tareas que no cumplan con las restricciones señaladas en este enunciado tendrán la calificación mínima (1.0).