Proyecto Programación Orientada a Objetos

Convocatoria de mayo/junio y julio para el curso 2023-2024

Última modificación: 29 de abril de 2024

Índice

- NOTA: Este fichero está sujeto a modificaciones continuas con el fin de que todo quede lo más claro posible.

List of Corrections

1. Planteamiento

Un escenario consta de entidades. El escenario viene dado por un mapa definido por el programador. Hay entidades que se mueven por el escenario y otras se mantienen fijas. Algunas son capaces de tomar decisiones y otras no. Se trata de recrear una situación de confrontación entre dos bandos en una era digital.

Nota: El enunciado es largo para poner bien en contexto cada uno de los requisitos.

2. El proyecto

Se resalta solo algunos de los aspectos importantes que deberás considerar para el desarrollo del proyecto.

2.1. Requisitos del proyecto

Ciudad

Todo **país** tiene un nombre, consta de una **bandera**, un **presidente** (que se ubica en un **palacio**) y de varias municipios. Un **municipio** consta de un **alcalde** (que se ubica en el **ayuntamiento**), un **gobernador** (que se ubica en una **residencia**) y su término municipal está formado por varias ciudades (o pueblos). Cada **ciudad**/pueblo tiene su propio nombre, su propia bandera y un templo religioso con su correspondiente responsable: **párroco**, **imán** o **rabino**, dependiendo de si es una **iglesia**, una **mezquita** o una **sinagoga**. De todas las ciudades **hay una**, considerada como 'la principal' que le da el nombre al municipio y es donde residen el alcalde y el gobernador. Hay varios **tipos de edificios**, cada uno con un aforo y cada uno con un tipo de planta: (palacio/en forma de U), (ayuntamiento/rectangular), (residencia/dos plantas), (iglesia/cruz latina), (mezquita/cuadrada) y (sinagoga/basilical). Las banderas se caracterizan por la imagen del escudo y las franjas de color.

Mapa de la ciudad

Vamos a representar una parte de alguna ciudad mediante un mapa. Un **mapa** es una matriz 2-dimensional, $n \times m$, cuyos elementos están formado por listas de celdas. Inicialmente está vacío y se le van añadiendo celdas indicando, para cada una, la capa que ocupa en la matriz. Una **celda**, $Celda(i, j, k)$, es un elemento de un mapa que se caracteriza por estar en una posición (i, j) y en una capa k dentro del mapa. Se define la capa de un mapa como el siguiente conjunto de celdas: $Capa(k) = \{Celda(i, j, k) | i = 0, \ldots, n-1; j = 0, \ldots, m-1\}$. Notar que la capa k hace referencia a la posición en la que se encuentra la celda en la lista de celdas que se ubica de la posición (i, j).

Un mapa conoce a todas sus celdas y toda celda conoce al mapa en el que se encuentra. Esto permite que a un mapa se le pueda *preguntar* por todas las celdas que se encuentran en una posición (i, j) o por todas las celdas que se encuentran en una capa. De forma similar, a una celda se le puede *preguntar* por todas las celdas que se encuentran en su capa, por todas las celdas que se encuentran en las distintas capas de su posición, por todas las celdas adyacentes que se encuentra en su posición y capa, y por todas las celdas adyacentes que se encuentran a su posición en las distintas capas. El *tipo de dato de retorno* de estos métodos será, siempre que sea posible, un array N-dimensional. También se dispone de un **método** que permite transformar cualquier array N-dimensional en una lista.

Para poder mostrar el mapa en la pantalla, se considerará que todas las celdas son cuadradas y todas tendrán el mismo valor para su lado, medido en píxeles. Su impresión consistirá en mostrar el mapa por capas en orden creciente k=0, k=1, etc.

Distinguimos varios tipos de celdas dependiendo de la capa:

- La $Capa(0)$ constituye el terreno. Todas las posiciones de la primera capa de un mapa está formada por este tipo de celdas. Está formada por los distintos **tipos de suelo**, que pueden ser transitables o no transitables.
  - Todas las **intransitables** tiene asociado el valor `-100`.
    Son intransitables un rio y el asfalto. El primero se caracteriza por la cantidad de litros por hora que circula por él y el asfalto por los decibelios que genera el tráfico cuando pasa por él.
  - Las **transitables** tiene asociado un valor entre 0 y 100 que indica la dificultad para transitar por él. A mayor dificultad mayor es el valor. El valor está restringido a un intervalo que está asociado a cada tipo de celda transitable.
    Son transitables un **carril bici** (con valores en [a=0, b)), una **acera** (con valores en [b, c)) y un **camino** (con valores en [c, d=100]). El primero se caracteriza por su número de carriles, la segunda por su ancho en metros y el tercero por cómo es su composición, que puede ser de guijarros, de piedras planas o de tierra.

- La $Capa(1)$ hace referencia al **entorno**. No todas las posiciones de la segunda capa de un mapa estará formada por este tipo de celdas. Alguna posición de esta capa puede estar vacía (None). Sus celdas crean una variedad de obstáculos, definen límites o simplemente añaden interés al terreno. Estas celdas se caracterizan porque tienen un valor numérico, entre -100 y 100, que influye en la dificultad del terreno.
  La **dificultad total** de una posición (i, j) viene dado por $valor_{final}(i, j) = valor(i, j, 0) + valor(i, j, 1)$; donde $valor(i, j, k)$ es el valor numérico de la celda que está en la posición (i, j) de la capa k. Si no hubiera celda en la capa $k = 1$, entonces el valor a considerar será solo el de la primera capa ($k = 0$). Si el valor total fuera nulo o negativo, se considerará que por la posición (i, j) no se podrá transitar.
  Consideraremos las siguientes celdas de entorno. Los **puentes**, que siempre hará transitable a un río -aunque se puede poner en cualquier sitio- y se caracteriza por su número de apoyos. Un **paso de peatones** que hace transitable al asfalto. Un **hidrante de incendio** que puede ser aéreo - un poste en la acera con más de una toma que hace al suelo intransitable - o puede ser enterrado - se sitúa en una arqueta, con tapa de fundición, bajo el nivel del pavimento de la acera. Un **charco de agua** que dificulta el tránsito en un camino y nunca se pone sobre un río. **Muro** de un edificio, que se puede colocar sobre un camino o una acera y convierte a la celda en intransitable.
- La $Capa(2)$ hace referencia a las **entidades** (estáticas o móviles). No toda la tercera capa de un mapa está formada por este tipo de celdas. Alguna posición de esta capa puede estar vacía (None). Observa que puede existir una celda en la posición (i, j) de la capa $k = 2$ pero no necesariamente debe existir una celda en esa posición en la capa $k = 1$.
  No pueden existir dos celdas-entidad en la misma posición, salvo que se encuentren en una celda-entidad que tenga contenedores.

Entidades de la ciudad

Entidades estáticas

Una entidad **estática** es la que no experimenta cambios en su posición o ubicación en el mapa. Permanece fija en un lugar específico y no se desplaza. Algunas entidades estáticas son: árboles, semáforos, puertas, fuego, puertas, etc. No confundir con las celdas del entorno. A diferencia de algunas celdas de entorno, estas entidades interactúan por lo que pueden cambiar de estado o desaparecer. Se considerarán las siguientes entidades estáticas.

- **Edificios**, formado por varias entidades estáticas. En concreto están formados por:
  - **Muros**. Cada muro (una celda) que se caracteriza por el material con el que está hecho: mampostería, ladrillo u hormigón.
  - **Puerta** del edificio (una celda), que puede estar abierta o cerrada, y a sus lados habrá muros de edificios.

  Hay varios tipos de edificios (ver primer párrafo del enunciado): palacio, ayuntamiento, residencia, iglesia, mezquita y sinagoga.
- Un **fuego**, que mientras que exista estará ardiendo y se caracteriza por su temperatura.
- Una **señal** vertical de tráfico, que siempre se coloca sobre la acera, y se caracteriza por su tipo y norma; por ejemplo de tipo prohibición con la norma de no superar lo 30Km/h. Los tipos posibles son de peligro, de prioridad, de prohibición. Considera 1 o 2 normas para cada uno.
- También está las entidad estática de **salto**, que almacena un par $(\vec{u}, s_m)$ que hace que una entidad móvil salte como si hubiese pisado una cama elástica en la dirección del vector $s_m\vec{u}$.
- Otras entidades estáticas son las **marquesinas** (y destinos) de los transportes. Constan de una *parada*, una *salida* y una *llegada*.
  Cuando un transporte llega a la marquesina se sitúa en la parada y bajan los pasajeros que tuviese ese destino colocándose cada uno de ellos en la parte de llegada que los "almacena" sin ningún orden. Pero cuando una entidad móvil no baja de un transporte, sino que procede de otra parte del mapa, se sitúan en la parte de salida respetando la cola de espera para montarse en el transporte que llegue.
  Notar que estas celdas estáticas de parada, tienen 3 contenedores (parada, entrada y salida) que almacenan entidades móviles, y todas se encuentrarán en la misma posición y capa.

Entidades pensantes

Tienen un nombre y un cerebro que les permite decidir estar en un cierto **estado**. Son pensantes, las siguientes entidades:

- El **presidente** de un país, con los estados: ir en Falcon, estar con el jefe del estado.
- Los **alcaldes**, con los estados: inagurar obra pública, estar en el pleno.
- Los **gobernadores** de un municipio, con los estados: permitir manifestaciones, no se sabe.
- Los **párrocos**, **imanes** y **rabinos** de una ciudad, con los estados: estar en oración, reunirse con sus seguidores.

Entidades móviles

Las entidades **móviles** son entidades pensantes que además se desplazan de acuerdo al movimiento que tenga establecido la entidad móvil. Existen 3 tipos de movimiento: el movimiento de la torre, el alfil y la reina del ajedrez, pero limitados a 1 paso https://es.wikipedia.org/wiki/Leyes_del_ajedrez.

Cuando se añade una entidad móvil, el mapa le asigna **un identificador único** con valores consecutivos y las guarda en un diccionario, aparte de guardarse en la correspondiente capa de la matriz. De forma análoga, cuando se elimina una entidad móvil del mapa, no solo se quita de la lista correspondiente a su posición, sino que también se quita del diccionario.

Existen dos tipos de celdas-entidades móviles: los transportes y los agentes.

Las entidades móviles **transportes** transportan a agentes y tienen un aforo - son contenedores. Cuando llegan a una parada, se pueden montar los agentes (lo que están en la parte/contenedor de salida) o se pueden bajar los pasajeros (colocándose en la parte/contenedor de llegada). Solo se pueden montar los que tienen permiso para ello y solo se bajan los que tengan un ticket para ese destino.

Entre los transportes están los **autobuses escolares** se caracteriza por pertenecer a un colegio, tiene un movimiento de torre y sólo se pueden montar los agentes infantiles. También están los **tranvías**, destinados a agentes adultos, que se caracterizan por el número de vagones que lo componen y por tener un movimiento de rey.

Un trasporte, al construirse, localiza en el mapa no menos de 3 paradas al azar, construye una ruta para visitarlas - mira el estado **PathFollowing** de los agentes para resolver esta parte - y se sitúa directamente en una de esas paradas. Cuando el transporte está en estado **Route** va pasando de una celda a otra a otra de la ruta cada vez que se invoca a su método `update()`. Pero de vez en cuando pasa al estado **Broken** para dejar de moverse. Cuando vuelve al estado **Route** sigue la ruta por la celda en la que se quedó.

Otras entidades móviles son los **agentes**, que se caracterizan por tener forma humanoide (tienen cabeza, tronco y extremidades), una edad, un estado de comportamiento (que se puede consultar en todo momento) y una salud (muerto, fastidiado, normal, bueno, excelente). Existen varios tipos de agentes móviles: ciudadano, pandillero y miembro de las fuerzas de seguridad.

- Los ciudadanos tienen los siguientes estados.
  Cuando un ciudadano llega a una marquesina entra en estado **InLine**: se queda inmóvil en la parada y se queda esperando a un transporte. Cuando llega el transporte se montan en él y sacan un ticket con un destino al azar de la ruta (otra marquesina/parada).
  Si está montado en un transporte estará en estado **Passenger** (estará en la posición donde se encuentre el transporte). Cuando llegan a su destino, se colocarán la zona de llegada y cambiarán de estado.
  Un agente también puede estar en estado **Walker**. En este estado selecciona al azar una posición adyacente a su posición actual de acuerdo. Cuando el móvil pisa una entidad estática de salto, su movimiento cambia y salta a la posición dada por la siguiente ecuación:
  $$\vec{p_t} = \vec{p_0} + \vec{u} s_m t + \frac{\vec{g} t^2}{2} \text{ con } \vec{g} = (0, -9{,}81, 0)$$
  siendo $\vec{p_0}$ la posición de la celdilla donde está la entidad de salto y $\vec{p_t}$ se corresponde a la posición del móvil cuando vuelve a tocar el suelo. Notar que las posiciones se corresponden a las coordenadas físicas $(ancho, alto, largo)$. Adáptalas al problema y explica cómo calculas $\vec{p_t}$ teniendo en cuenta que matemáticamente se cumple que:
  $$\begin{aligned} p_t^x &= p_0^x + u^x s_m t && \text{componte } x \text{ de la ecuación} \\ p_t^y &= p_0^y + u^y s_m t + \frac{g^y t^2}{2} && \text{componte } y \text{ de la ecuación} \\ p_t^z &= p_0^z + u^z s_m t && \text{componte } z \text{ de la ecuación} \end{aligned}$$
  En el estado **PathFollowing** necesitan un punto de destino (si no se le suministra lo selecciona de forma aleatoria). Entonces busca un camino, usando Dijkstra, y realiza los movimientos necesario para recorrer el camino teniendo en cuenta: (1) evitará las marquesinas (para no entrar en estado **InLine**, (2) evitará las entidades estáticas de salto (para no cambiar el trayecto), y (3) los adyacentes de una celda está determinados por el movimiento (hay 3 distintos) - p.e. los adyacentes del movimiento alfil son 4 y están en las diagonales; pero hay 8 adyacentes si se mueve como un rey.
- Los miembros de las **fuerzas de seguridad** tienen un nivel de entrenamiento (numérico). Pueden estar en estado pasivo o de confrontación. Hay dos cuerpos en las fuerzas de seguridad: los locales y los nacionales. Ambos puede usar distintos tipos de **armas** de ataque o de defensa (se indican a continuación).
  - Las **fuerzas locales** pueden ser policías locales y guardias de seguridad. Usan siempre porras extensibles (al cinto).
    - Los **guardias de seguridad**, en confrontación, se arman de gas pimienta.

- Cada **policía local** pertenecen a un municipio y siempre lleva pistola reglamentaria y esposas (al cinto). En caso de confrontación se puede equipar de casco (a la cabeza), chaleco antibalas (al tronco) y escudo (en un brazo). En estado pasivo, no llevan nada salvo lo que están obligados a llevar en el cinto. Cuando están en estado de confrontación siempre cubren cabeza y tronco, en una mano llevan la porra y en la otra el escudo.

- En las fuerzas de **seguridad nacional** están los nacionales y los antidisturbios.
  - Los **antidisturbios** se compone de la que conduce torretas de agua y la que usa drones. Por comodidad supondremos que el antidisturbio de una torreta, al pulsar un botón, lanzará agua a una posición, y la pulsar otro avanzará la torreta (dejará de avanzar al dejar de pulsar). De forma análoga, el antidisturbio que usa un dron, al pulsar un botón el dron avanzará. Cuando no pulsan el botón, ni lanzan agua ni avanza el dron, respectivamente. Cuando un torreta lanza el agua a una posición (y por lo tanto a una casilla) todas las entidades pierden 1 o varias unidades de su estado de salud.
  - Los **nacionales** tienen las mismas armas de ataque y defensa que la policía local, pero además pueden usar un rifle de pelotas de goma (usando los dos brazos) o un dispositivo de electrochoque (tser). En estado de confrontación realizan **formaciones tácticas**, se cubren con casco (a la cabeza) y chaleco antibalas (al tronco). En una formación todos los que están en el exterior llevan escudos y porra, mientras que los que están en el interior llevan rifles.
    Cuando está en confrontación forma parte de una formación. Una formación es un grupo de entidades móviles que se distribuyen en un conjunto de celdas adyacentes formando una figura geométrica. Distinguimos las siguientes formaciones:

```
Formación cruz                    Formación rombo                   Formación aspa.
                F                                 F                               F     F
        F       F                         F       FFF                     F   F     F   F
 F      F       F                  F     FFF     FFFFF                F F     F F       F F
F*F   FF*FF   FFF*FFF  ......     F*F   FF*FF   FFF*FFF  ......       *       *         *      ......
 F      F       F                  F     FFF     FFFFF                F F     F F       F F
        F       F                         F       FFF                     F   F     F   F
                F                                 F                               F     F
t=1    t=2      t=3               t=1    t=2      t=3                t=1    t=2       t=3
```

    Cuando se crea una formación se determina el tipo de formación a realizar en una posición del mapa. La formación pueden ser en cruz, en rombo o en aspa. Entonces se van añadiendo entidades móviles a la formación en las posiciones del mapa de acuerdo a la forma geométrica de la formación. La primera entidad móvil ocupa la posición marcada con $*$, que llamaremos posición líder, y las siguientes van ocupando las posiciones `F` en el orden que tú consideres siempre y cuando sean posiciones libres y siempre empezando por las más cercanas al líder. En las figuras superiores se muestran cómo van creciendo las formas geométricas para distintos tamaños `t=1,2,3, ...`.

    En la figura de la derecha se tienen dos formaciones en aspa para tamaño 2. En ambas formaciones se han añadido 5 entidades móviles `*`. La formación de la derecha está bien construida porque las 5 unidades se han añadido lo más cerca posible de la posición del líder. Sin embargo la formación de la izquierda está mal construida.

```
Formación aspa para t=2
 F   *     F   F
  F *       * *
    *         *
  * F       * *
 *   F     F   F
```

    En estado **Formation**, su movimiento depende de líder. Se limita a seguirlo pero colocándose en la posición relativa de la formación que se le haya asignado. Observa que el líder, aunque forma parte de una formación, no puede estar en estado Formation ya que entonces se seguiría a sí mismo (y no tiene sentido). El líder se moverá según el estado **Walker** e ignorando a las entidades estáticas de salto y las marqusinas; por tanto, cualquier miembro de una formación tampoco podrá saltar rompiendo la formación ni se quedará esperando un transporte. Para no complicar el ejercicio, supondremos que ningún miembro de la formación encuentra obstáculos en su movimiento; es decir, ignorarán también las celdas intransitables (pasará por encima de ellas): ninguno de la formación hará comprobación del tipo de celda donde se encuentra.

    En una formación se puede dar varias ordenes (establecidas por el cerebro el líder):
    - ◇ Seguir: los miembros de la formación siguen al líder según se acaba de explicar.
    - ◇ Nueva formación: puede ordenar construir una nueva formación (otra geometría) con todas las entidades que compongan la formación actual.
    - ◇ Gritar: puede pedir a todos sus miembros que digan una frase. A toda formación se le asocia un sistema de codificación en el momento de su creación. El líder codifica la frase y se la pasa a todos sus miembros, entonces todos los miembros la descodifican para saber lo que dice.

    Justifica cómo el líder manda órdenes a los demás miembros de la formación y, en particular, cómo manda el mensaje a todos los miembros de la formación y/o cómo todos los miembros de la formación están siempre a la escucha de lo que diga su líder. Justifica también que hacen los miembros de la formación.

2.2. Escenarios Evaluables

2.2.1. Escenario Básico

Dispone de las siguientes funcionalidades:

1. Trabaja con los elementos relacionados de una ciudad (país, bandera, edificios, ...)
2. Codifica el mapa de la ciudad (mapas y celdas)
3. Se tienen las entidades estáticas y pensantes.
4. Se tienen las entidades móviles ciudadanos con el estado Walker.
5. Controla errores y genera errores propios. El programa no puede parar nunca por un assert o un raise.
6. Construye no menos de dos objetos de cada tipo.
7. Guardan la información de todos las celdas de un mapa en un fichero de texto (para las que se piden a este nivel).
8. El fichero principal se encontrará en el inicio del proyecto, se llamará `main.py` y contendrá no más de 20 líneas de código.
9. **Cada clase deberá estar definida en un único fichero** y se deberá de colocar en la carpeta más adecuada. Por ejemplo, una estructura jerárquica de carpetas es la siguiente

```
Proyecto
|- main.py
|- readme.md
|- uml.jpg
|-- Pais
|  |- Pais.py
|  |- ...
|-- Mapa
|  |- ...
|-- Celdas
|  |- Terreno
|  |- Entorno
|  |- Entidades
|     |- Estaticas
|     |- Pensantes
|     |- Moviles
|     |- Formaciones
```

En el fichero `readme.md` (con marcas Markdown) indica todo lo que consideres importante mencionar.

Si tuvieras que juntar 2 clases o más en un único fichero, justifica en el fichero `readme.md` los ficheros que tienen más de 2 clases y justifica por qué lo has fusionado.

No presentar (muchos) ficheros denota falta de destrezas en el uso de la programación modular. Presentar pocos ficheros puede suponer suspender la práctica.

2.2.2. Escenario Avanzado

Dispone de las siguientes funcionalidades:

1. El escenario básico.
2. Dispone de las entidades móviles de transportes y usan Dijkstra.
3. Se implementan todos los estado en las entidades ciudadanos
4. Se tienen las distintas fuerzas de seguridad.
5. Se construyen formaciones.
6. El líder de una formación se comunica al resto de elementos de la formación.
7. Implementa, al menos, 3 interfaces abstractas.

2.2.3. Escenario de Simulación

Dispone de las siguientes funcionalidades:

1. El escenario avanzado.
2. Hay entidades móviles que se desplazan por le mapa.
3. Su usa logging para guardar lo que ocurre en el mapa.
4. Se muestra el escenario (de vez en cuando) en el terminal.

2.3. Documentación

Los módulos estarán documentados. Esto significa que:

- Todo módulo comenzará con una explicación breve de su contenido y los autores. En su caso, se indicará la bibliografía o recursos usados para la resolución del problema. No pongas el enunciado del ejercicio.

- Cada clase del módulo irá comentado explicando su propósito y uso.
- También se indicará la especificación informal de cada método. En su caso, se indicará de dónde se ha obtenido, qué se ha modificado y todo aquello que sea relevante al usar una obra modificada.
- **La documentación se exige en todos los escenarios.**

Junto a los ficheros principales (uno para cada escena) se encontrará el fichero `uml.jpg` con el **diagrama UML** del proyecto. Si lo quieres hacer rápido, dibújalo en un papel usando **buena** letra y le sacas una foto. Puedes intentar usar `pyreverse -o salida DirectorioProyecto` que se encuentra en el paquete `pylint` siendo `salida` el texto `png` o `plantuml` (pero no se garantice que funciones correctamente). Si te sobra tiempo, puedes usar http://www.plantuml.com/plantuml/ siguiendo la documentación de https://plantuml.com/es/class-diagram o retocando la salida de pyreverse.

También deberá presentar el fichero ~~`documentacion.md`~~ `readme.md` usando el lenguaje de marcas Markdown de acuerdo a la plantilla que se proporcione.

3. Normas de Entrega

La práctica deberá de presentarse solo por el AV, vía `Tareas`, en las fechas establecidas por la coordinación de 2º.

Todo deberá entregarse en **un único fichero comprimido**. El fichero deberá tener cualquiera de las compresiones libres de patentes: 7z, zip, gzip, bzip, ... **ojo, .rar no está permitido**.

El fichero comprimido contendrá **la siguiente estructuras de directorios** (**no uses acentos o espacios** en los nombres de los directorios/ficheros):

```
Proyecto
|- main.py
|- readme.md
|- uml.jpg
|-- Pais
|   |- Pais.py
|   |- ...
|-- Mapa
|   |- ...
|-- Celdas
|   |- Terreno
|   |- Entorno
|   |- Entidades
|      |- Estaticas
|      |- Pensantes
|      |- Moviles
|      |- Formaciones
```

En ningún caso se presentará la carpeta `venv` del entorno virtual del proyecto. Observa que no aparece en la estructura de directorios. La carpeta `venv` contiene una distribución entera de Python con todas las librerías que uses. Presentar el proyecto con el entorno virtual `venv` es una presentación incorrecta del proyecto y **se suspende** por una presentación con defecto de forma.

Muy importante: Los proyectos con ficheros acentuados o con el entorno virtual venv NO se corregirán y se suspenderá directamente.

29/3/2024: El fichero `readme.md` contendrá la siguiente información:

- Nombre y apellido de los autores.
- Un apartado indicando qué escenarios se han desarrollado.
- Un apartado para cada escenario y para cada punto a desarrollar del escenario se creará un subapartado. En dicho subapartado se indicará lo más relevante que considere.
  Por ejemplo. Si desarrolla el escenario básico se pondrá el apartado *Escenario Básico* y el subapartado para el punto *2. Codificación del mapa de la ciudad (mapas y celdas)* donde puede indicar los métodos implementados y cómo relacionan todas las clases que se necesitan para este punto.

Por ejemplo. Si desarrolla el escenario avanzado se pondrá el apartado *Escenario Avanzado* y un subapartado para el punto *5. Construcción de formaciones* donde deberá explicar lo más importante sobre cómo ha resuelto este punto.
No incluya código Python en ningún caso. Los apartados deben ser muy breves y concisos, pero con la explicación suficiente para entender cómo ha abordado el problema.
- En su caso justifique razonadamente por qué no ha incluido solo una clase por fichero.

En `https://markdown.es/sintaxis-markdown/` tienes las marcas que se usan en Markdown

4. Evaluación

4.1. Criterios para la evaluación

Los aspectos que se tendrán en cuenta durante la evaluación y calificación del trabajo presentado son:

- Funcionamiento correcto de los escenarios presentados. Pude que estén incompletos, pero no pueden tener errores en tiempo de ejecución.
- Presentación del trabajo/informe con todos los apartados mínimos solicitados.
- Usar correctamente las marcas/comandos de Markdown para generar información. Documentación bien estructurada, ordenada y clara. Se dará una plantilla.
- En su caso, explicación, presentación y originalidad de la solución al problema planteado.
- Usar correctamente la metodología de la POO en todos los aspectos que han sido mostrados en las sesiones teorico prácticas, destacando:
  - diseñar e implementar correctamentes los TDAs,
  - definir correctamente las clases y sus relaciones de herencia o clientelismo, si las hubiera,
  - uso correcto de constructores, miembros y métodos (de clase y de objeto);
  - definición e implementación adecuada de clases abstractas e interfaces;
  - manejo de entrada y lecturas de datos;
  - control de excepciones.

4.2. Calificación

La calificación se hace de acuerdo a los escenarios presentados. Para calificar un nivel se tiene que haber superado los escenarios anteriores con, al menos, la calificación que se indica. Se recomienda no desarrollar escenarios posteriores si no ha desarrollado los anteriores, porque pueden no ser evaluados. La calificación que se indica para cada escenario es su calificación máxima a la que puede aspirar.

Escenas	**Máxima calificación**	**Calificación mínima en escenarios anteriores**
Básico	6.9	-
Avanzado	8.9	3.5
Simulación	10	7

Se exigirán los requisitos mínimos, que son los siguientes:

1. Se presenta en su totalidad el *Escenario Mínimo*. Es un requisito esencial. Su no presentación 'completa' supone suspender la práctica y no seguir corrigiendo.
2. *Funcionamiento del programa*
   Todo lo que se presente debe ser un programa que funciones correctamente, cumpliendo lo que se pide en el enunciado.
   Un programa que no funcione, independientemente de los niveles presentados, conlleva un cero automáticamente y no se evaluará nada.
3. *Documentación del proyecto*
   Se presenta toda la documentación del apartado 2.3.

El grupo podrá ser convocado a una entrevista para determinar la nota final. Después de la entrevista se puede mantener la nota obtenida o se puede suspender el proyecto en su totalidad en función de las respuestas dadas sobre cualquier aspecto teórico/práctico sobre el diseño de la solución.