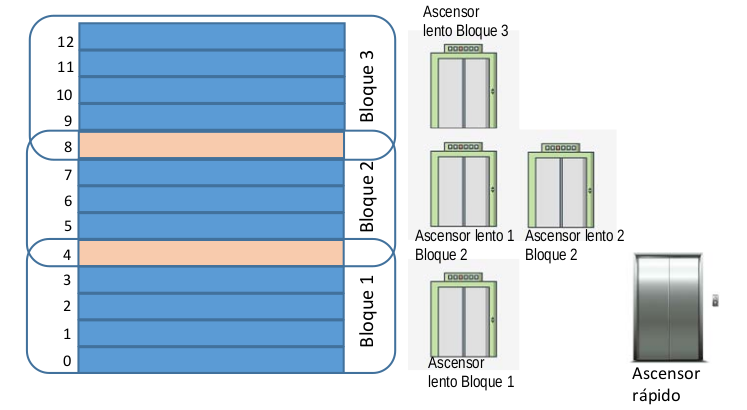
Universidad Complutense de Madrid.

Facultad de Ingeniería Informática

Robótica.



**Práctica 2B.**

**Resolución de problemas con búsqueda.**

Grupo 5:

Frederick Ernesto Borges Noronha

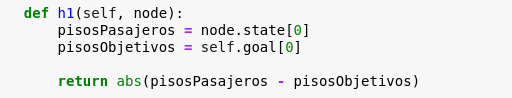
Víctor Manuel Cavero Gracia

**APARTADO A**

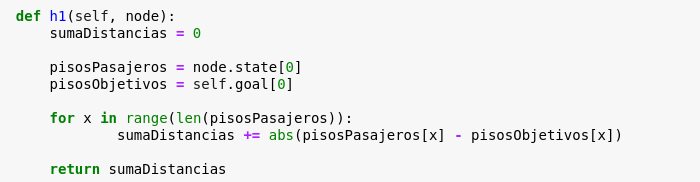
Discute alternativas para la función de coste y heurísticas para el problema, así como sus propiedades de admisibilidad y consistencia. Se puede suponer que el ascensor rápido tarda la mitad que los lentos y que todos los lentos tardan igual. Aunque el objetivo final es que todas las personas lleguen a su destino en el menor tiempo total también se pueden considerar otros aspectos como el tiempo de espera de las personas en las plantas o el número de paradas. En el problema original los ascensores se pueden mover a la vez por el edificio, pero si crees que es necesario hacer alguna simplificación del problema deberás justificar la respuesta adecuadamente.

Hemos definido principalmente dos heurísticas en el desarrollo de nuestro problema.

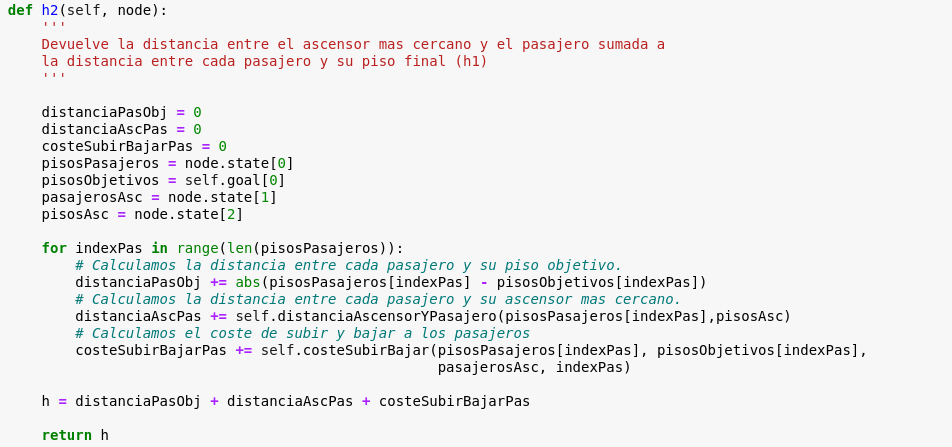
En la solución con **un solo pasajero y un ascensor** definimos una h1 sencilla que simplemente esta basada en la distancia que existe entre el pasajero y su piso de destino, de tal manera que el coste de subir/bajar el ascensor es de una unidad. Esta diferencia esta realizada en valor absoluto para evitar valores negativos.



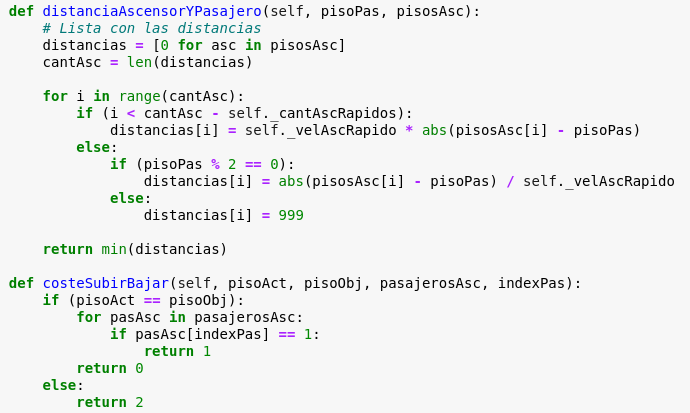
Durante el desarrollo de la solución con **varios pasajeros y un ascensor** tuvimos que redefinir h1 para sumar cada una de las distancias entre el pasajero y su piso de destino.



Cuando procedimos a realizar la solución con **varios pasajeros y varios ascensores** nos dimos cuenta de que la heurística realizada anteriormente no era la óptima por lo que tuvimos que mejorarla y así el tiempo de ejecución del problema se reduciría.

****

A parte de calcular la distancia de cada pasajero a su piso de destino, calculamos su distancia al ascensor más cercano (teniendo además en cuenta para el cálculo el coste del ascensor rápido en el caso de que el pasajero se encuentre en un piso par, haciendo que está heurística sea admisible con la presencia de ascensores rápidos) y asignamos un coste de subir y bajar el pasajero del ascensor. Hemos utilizado para ello las dos funciones siguientes:



La heurística finalmente toma como resultado la suma de estos tres cálculos realizada para cada uno de los pasajeros existentes.

**APARTADO B**

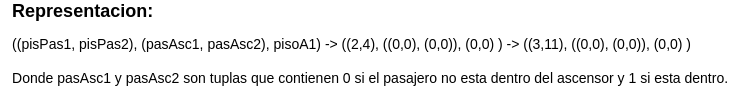
Especificar otras instancias de problema cambiando la situación inicial y final y cómo afectan a la resolución del problema.

**APARTADO C**

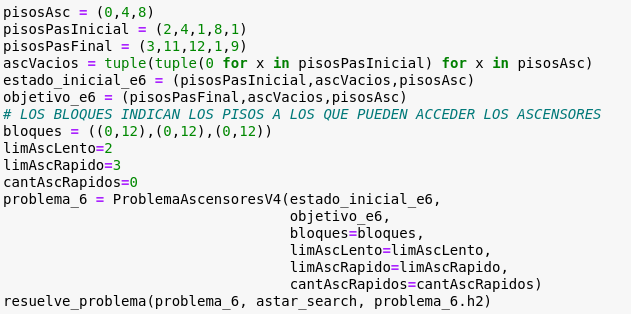
Explicar si la representación elegida permite modificar los siguientes datos.

**- Cambiar plantas de origen y destino de los pasajeros:**

Podemos ver como gracias a nuestra definición de estado se pueden variar sin ningún tipo de problema el origen y destino de los pasajeros. Estos, obviamente, se tendrán en cuenta en las operaciones.

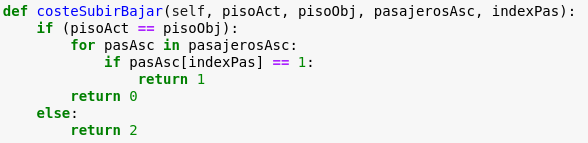


Para introducir las plantas de origen simplemente cambiamos el estado inicial y en concreto la tupla *(pisPas1, pisPas2…)* que son las que almacenan el piso de los pasajeros, y para cambiar las plantas de destino se modifica el estado objetivo. También esto queda totalmente probado en los ejemplos presentes en el documento, tales como:



**- Tener en cuenta el tiempo de subir y bajar pasajeros para cambiar de ascensor.**

De hecho, hemos tenido en cuenta el coste de subir y bajar del ascensor en la realización de la heurística *h2*, asignando un coste de 1 para la operación de subir o bajar y un coste de 2 en el caso de que el pasajero no se encuentre en la posición objetivo ya que debería llegar al piso y bajar del ascensor.

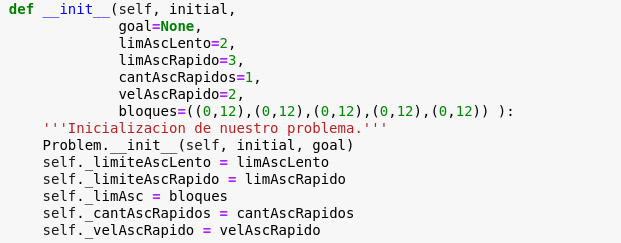


**- Incluir nuevos pasajeros**

Al igual que en la primera de las modificaciones solo tendrían que introducirse los nuevos pasajeros en la tupla *(pisPas1, pisPas2…)* con su piso inicial (en el estado inicial) y su piso destino (en el estado objetivo).

**- Cambiar la capacidad de los ascensores**

Se podría cambiar modificando las variables *limAscLento* y *limAscRapido* que son introducidas en la declaración del problema. Por defecto reciben los valores del problema presentado en la práctica.

****

**- Añadir nuevos ascensores, por ejemplo, un ascensor rápido que se mueva entre las plantas impares.**

Podríamos cambiar el estado inicial del ascensor rápido poniéndole que este empiece en el piso número 1 y cambiando el bloque en el que actua el ascensor, este se le pasa en al inicializar el problema, por lo tanto no sería necesario modificar el código de la representación.

**APARTADO D**

Incluir análisis de resultados (eficiencia, optimalidad) de los algoritmos y heurísticas elegidos. Se debe incluir una comparativa entre las mejores opciones estudiadas.