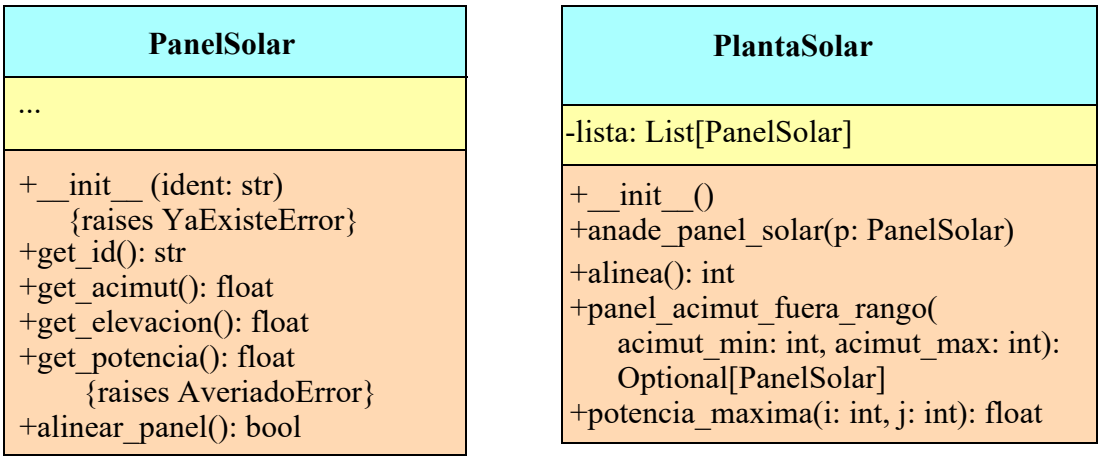
Examen de Prácticas de Programación
Julio 2021
(Grados en Física y Matemáticas)

Se dispone en el módulo panel.py de una clase llamada PanelSolar que permite guardar la información y operaciones de un panel solar utilizado para producir electricidad. La clase responde al siguiente diagrama de clases, en el que se omiten los atributos por no ser relevantes para el problema. Sus métodos hacen lo siguiente:



- *constructor*: se le pasa el identificador del panel como parámetro. Lanza YaExisteError si se ha intentado crear un panel solar con el mismo identificador que otro anterior
- get_id(): Retorna el identificador del panel.
- get_acimut(): Retorna el acimut del panel, en grados, que nos indica hacia dónde apunta en la dirección horizontal. El Sur se representa con el valor 180 grados. Si el panel está averiado retorna math.nan.
- get_elevacion(): Retorna la elevación del panel, en grados, que nos indica hacia dónde apunta en la dirección vertical, sobre el horizonte. La vertical se representa con el valor 90 grados. Si el panel está averiado retorna math.nan.
- get_potencia(): Retorna la potencia eléctrica, en vatios, que el panel está produciendo en este instante. Lanza la excepción AveriadoError si el panel está averiado y no consigue generar electricidad.
- alinear_panel(): Alinea el panel con el sol. Retorna True si se ha conseguido alinear, y False en caso contrario, por ejemplo si fallan las comunicaciones o un motor.

En el mismo módulo panel.py se encuentran ya creadas las siguientes excepciones:

- AveriadoError, que representa la condición de avería de un panel solar que le impide generar electricidad.
- YaExisteError, que indica que se ha intentado crear un panel solar con el mismo identificador que otro anterior.

Lo que se pide es implementar el módulo planta_solar.py que contendrá dos elementos. En primer lugar, la clase PlantaSolar, que representa una planta compuesta por un conjunto de paneles solares, siendo cada uno un objeto de la clase PanelSolar. La clase PlantaSolar responde al diagrama de clases que se muestra arriba y debe contener el siguiente atributo privado:

- lista: un lista de objetos de la clase PanelSolar.

Los métodos de la clase PlantaSolar deben hacer lo siguiente:

- *constructor*: crea la lista vacía.
- anade_panel_solar(): Añade al final de la lista un nuevo panel solar que se pasa como parámetro.
- alinea: Alinea todos los paneles de la planta llamando a alinear_panel() para cada uno y, cuenta los que resulten correctamente alineados y no estén averiados. Finalmente retorna esa cuenta.

  Recordar que para saber si un panel se alinea correctamente se puede comprobar el valor retornado por alinear_panel(). Y para saber si está averiado, se puede mirar si el resultado de get_acimut() o get_elevacion() vale math.nan.
- panel_acimut_fuera_rango(): Encuentra y retorna el primer panel de la lista que no esté averiado y cuyo acimut esté fuera del intervalo cerrado [acimut_min, acimut_max]. Retorna None si no hay paneles que cumplan esa condición. Obligatoriamente se debe utilizar un algoritmo de búsqueda.

  Recordar que para saber si un panel está averiado, se puede mirar si el resultado de get_acimut() vale math.nan.
- potencia_maxima(): Retorna la potencia máxima de los paneles de la lista cuyos índices están en el intervalo cerrado [i, j], aunque retorna math.nan si algún panel de ese intervalo está averiado. Para detectar los paneles averiados debe tratarse la excepción AveriadoError lanzada por get_potencia(), retornando en ese caso math.nan.

El otro elemento del módulo planta_solar.py que se pide, es un programa principal que pruebe todos los métodos de la clase PlantaSolar, realizando los siguientes pasos:

a. Crea un objeto de la clase PlantaSolar.

b. Invoca al método anade_paneles(), que se ofrece hecho y que añade 20 paneles solares a la planta. Si se lanzase YaExisteError en este paso, debe ponerse un mensaje de error en pantalla y continuar con los siguientes pasos.

c. Invoca a alinea() y muestra en pantalla el valor retornado

d. Muestra en pantalla la potencia máxima entre los índices 1 y 8

e. Muestra en pantalla la potencia máxima entre los índices 2 y 10

f. Muestra en pantalla el identificador del primer panel con acimut fuera del rango [3.0, 20.0]º, o None si no hubiese ninguno

g. Muestra en pantalla el identificador del primer panel con acimut fuera del rango [0.0, 40.0]º, o None si no hubiese ninguno

Resultados esperados:

```
b) Error al añadir panel: El identificador id3 ya existía
c) Alineados OK: 18
d) Potencia máx. entre índices 1 y 8: 108.0 W
e) Potencia máx. entre índices 9 y 12: nan W
f) Primer panel acimut fuera [3.0, 20.0]: id17, acimut= 21.0º
g) Primer panel acimut fuera [0.0, 40.0]: None
```

Valoración. Se valoran principalmente los elementos que funcionen correctamente al ejecutar las instrucciones correspondientes del main(). Se valora también la eficiencia, el estilo de

programación y la documentación.

1) Encabezamiento de la clase y constructor: 1 punto
2) anade_panel_solar(): 0,5 puntos
3) alinea(): 2,5 puntos
4) panel_acimut_fuera_rango(): 3 puntos
5) potencia_maxima(): 3 puntos