Proyecto de Nivel 4
Astronomía

Objetivo general

El objetivo general de este proyecto es que usted practique todos los conceptos estudiados en el nivel 4 del curso. Recuerde que este proyecto debe realizarse de forma **completamente individual**.

Objetivos específicos

1. Ejercitar la implementación de algoritmos para construir y recorrer matrices.
2. Familiarizarse con las librerías pandas y matplotlib.
3. Fomentar la habilidad de descomponer un problema en subproblemas y de implementar funciones que los resuelven.

Contexto

La astronomía es la ciencia que se ocupa del estudio de los cuerpos celestes del universo, incluidos los planetas y sus satélites, los cometas y meteoritos, las estrellas y la materia interestelar, los sistemas de materia oscura, gas y polvo llamados galaxias y los cúmulos de galaxias. La mayoría de la información usada por los astrónomos es recogida por la observación remota, aunque en algunos casos se ha conseguido reproducir en laboratorio la ejecución de fenómenos celestes. También es una de las pocas ciencias en las que los aficionados aún pueden desempeñar un papel activo, especialmente en el descubrimiento y seguimiento de fenómenos como curvas de luz de estrellas variables, descubrimiento de asteroides y cometas, etc.

En este proyecto usted trabajará con los datos de todos los planetas extrasolares confirmados (llamados exoplanetas), descubiertos entre 1988 y 2018 (más de 3000). Esta información se encuentra recopilada en la base de datos llamada Open Exoplanet Catalogue (https://www.kaggle.com/mrisdal/open-exoplanet-catalogue). En el archivo "exoplanetas.csv" usted encontrará una versión simplificada de los datos originales. Suprimimos de este catálogo los planetas que no han sido confirmados como tal, eliminamos algunas columnas y modificamos las clasificaciones de valores numéricos por valores categóricos.

Trabajando con este conjunto de datos usted puede convertirse en un explorador espacial, analizando las características de todos los exoplanetas descubiertos. Los campos de los datos incluyen atributos de estrellas y de planetas, métodos de descubrimiento y (por supuesto) fecha de los descubrimientos.

Le deseamos la mejor de las suertes en esta exploración espacial!!!

Descripción de la aplicación

Parte 1: Cargar los datos

La primera opción de la aplicación debe permitir que se carguen los datos de un archivo csv a un DataFrame. Al usuario se le debe preguntar el nombre del archivo. **La función que implemente esta opción debe recibir como parámetro el nombre del archivo y debe retornar un DataFrame.**

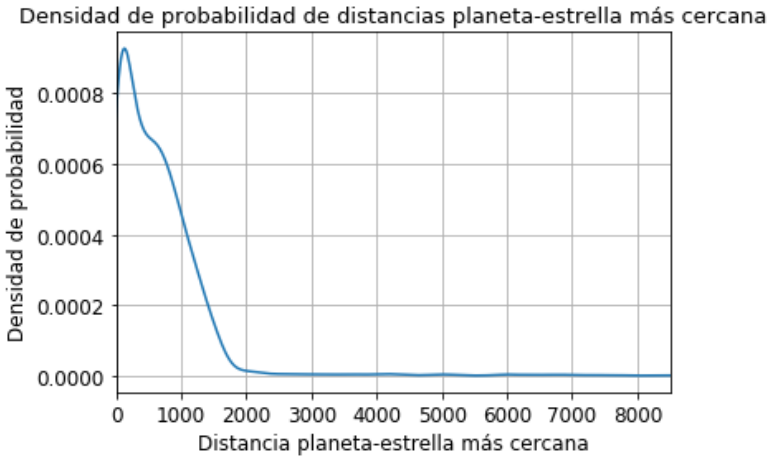
Las columnas del archivo y sus significados son las siguientes:

- NOMBRE: nombre del planeta que se descubrió
- MASA: masa del planeta
- DESCUBRIMIENTO: año de descubrimiento del planeta
- ACTUALIZACION: última vez que se actualizó la información del planeta
- ESTADO_PUBLICACION: medio en el cual se publicó el descubrimiento: Announced on a professional conference, Announced on a website, Published in a refereed paper, Submitted to a professional journal
- TIPO_DETECCION: tipo de detección que se utilizó para descubrir el planeta: Astrometry, Imaging, Microlensing, Other, Primary Transit, Primary Transit, TTV
- RA: hace referencia al Right Ascencion, que es una de las coordenadas del planeta visto desde la tierra
- DEC: hace referencia a Declination, que es la otra coordenada del planeta visto desde la tierra
- DISTANCIA_ESTRELLA: distancia a la estrella más cercana
- MASA_ESTRELLA: masa de la estrella más cercana

Ayuda: **c**uando esté estudiando el problema y el archivo, recuerde que las funciones describe() y unique() pueden serle de utilidad. La función describe() aplicada sobre un DataFrame retorna información estadística de todas las columnas numéricas. La función unique(), aplicada sobre una columna, retorna una lista con los valores que aparezcan en esa columna.

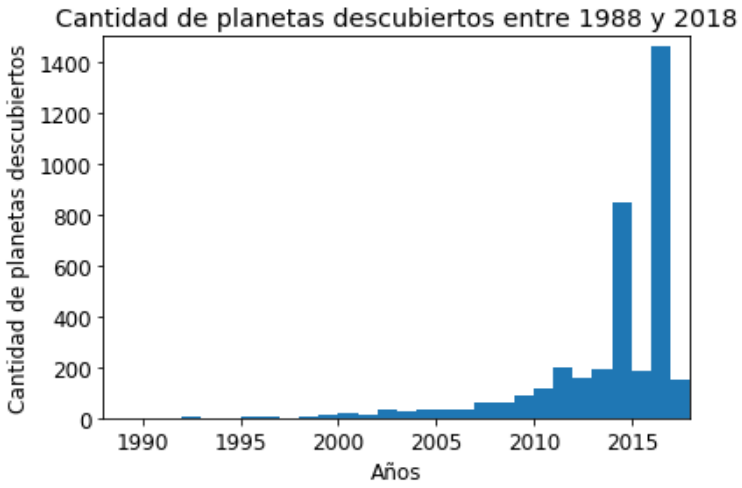
Parte 2: Estudiar las distancias entre los planetas y su estrella más cercana

En esta parte de la aplicación se debe mostrar la distribución de las distancias de cada uno de los planetas a su estrella más cercana. Para esto se debe generar una gráfica de tipo KDE usando la columna DISTANCIA_ESTRELLA. Las gráficas de tipo KDE (kernel density estimation) son una forma de estimar la función de densidad de probabilidad de una variable aleatoria, que en este caso, se trata de la distancia entre un planeta y su estrella más cercana. Estas gráficas se basan en curvas y pueden verse como una versión suavizada de un histograma que intenta llenar los espacios vacíos en la muestra. Debido al efecto de introducir estas curvas, las gráficas por defecto muestran que es posible que un planeta tenga una distancia negativa a su estrella más cercana. Para evitar este problema, es necesario definir límites explícitos para la gráfica usando el parámetro xlim, el cual corresponde a una tupla con el menor valor (0) y el mayor valor esperado de distancia (8500). La siguiente figura muestra la apariencia de esta gráfica.

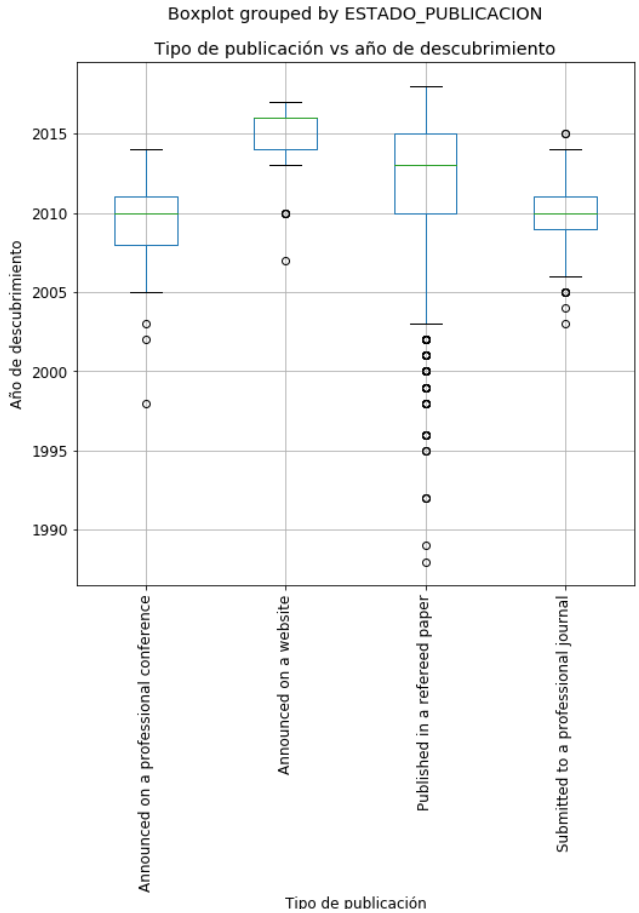


Parte 3: Analizar el descubrimiento de los planetas

La primera opción de esta parte de la aplicación debe calcular y graficar un histograma donde se muestre cuántos planetas fueron descubiertos a lo largo de los años. Para esto se debe crear una gráfica de tipo histograma (hist) con los valores de la columna DESCUBRIMIENTO. El histograma debe separar los datos en 30 grupos (bins). De esta forma se podrá apreciar la cantidad de planetas que se descubrieron cada año, entre 1998 y 2018. La siguiente figura muestra la apariencia de esta gráfica.

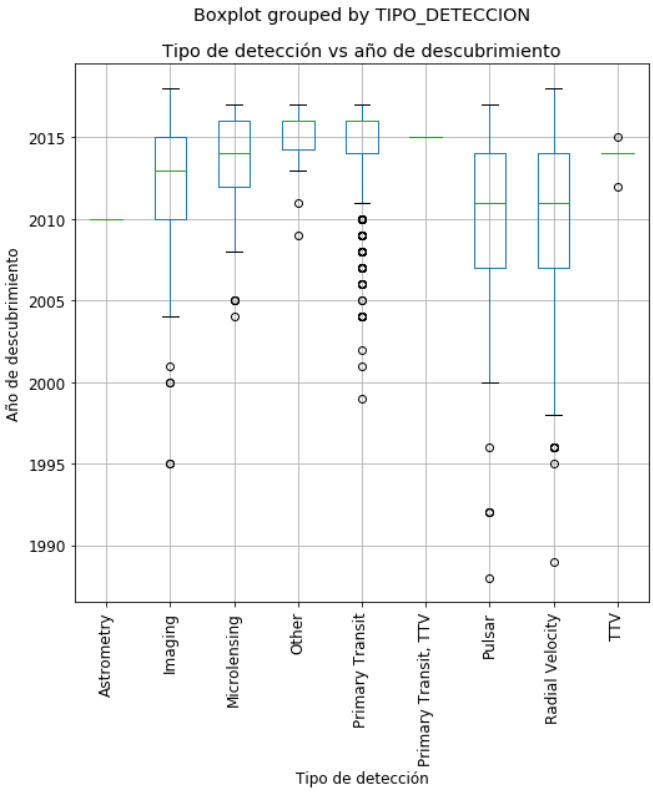


La segunda opción de esta parte debe permitir relacionar los años de descubrimiento de los planetas y los tipos de publicación que se utilizaron para informar los descubrimientos. Para esto se debe crear una gráfica de tipo boxplot, usando la columna DESCUBRIMIENTO y agrupando los datos de acuerdo con la columna ESTADO_PUBLICACION. La siguiente figura muestra la apariencia de esta gráfica.



Ayuda: para que los textos de los tipos de publicación aparezcan verticales y no horizontales (como lo es por defecto), utilice el parámetro rot=90 al momento de crear el boxplot. Adicionalmente, para que los títulos no se sobrepongan entre sí, configure el tamaño de la figura con el parámetro figsize=(8,8) al momento de crear el boxplot.

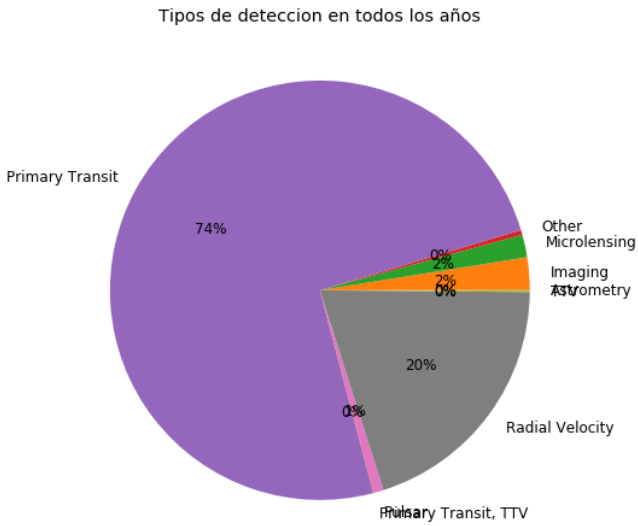
La tercera opción de esta parte debe permitir relacionar los años de descubrimiento de los planetas y los tipos de detecciones que se utilizaron. Para esto se debe crear una segunda gráfica de tipo boxplot, usando la columna DESCUBRIMIENTO y agrupando los datos de acuerdo con la columna TIPO_DETECCION. La siguiente figura muestra la apariencia de esta gráfica.



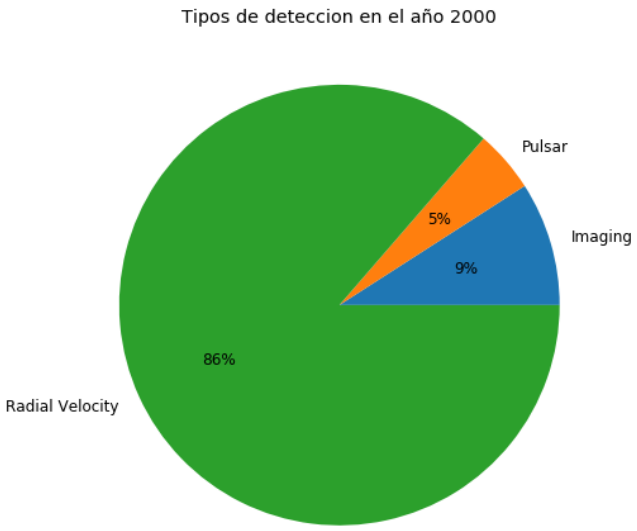
La cuarta opción de esta parte del programa debe permitir generar dos gráficas de tipo pie que muestren:

- los tipos de detección para todos los años en los que se descubrió algún planeta, y
- los tipos de detección para un año específico en el que se descubrió algún planeta.

Se debe dar al usuario la posibilidad de escoger graficar todos los años o introducir al programa un año específico de su elección. La siguiente figura muestra la apariencia de la gráfica de pie teniendo en cuenta todos los años:



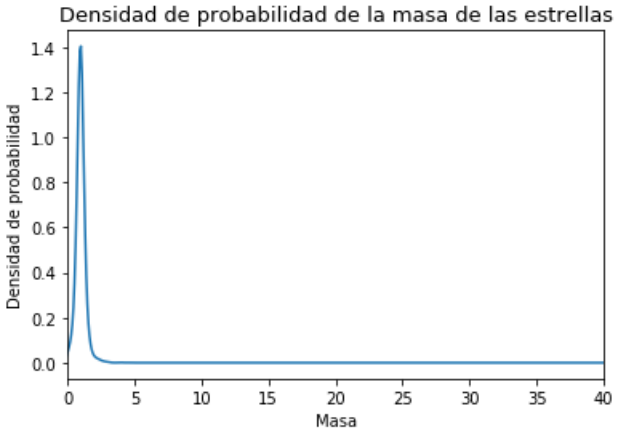
La siguiente figura muestra la apariencia de la gráfica de pie teniendo en cuenta únicamente el año 2000:

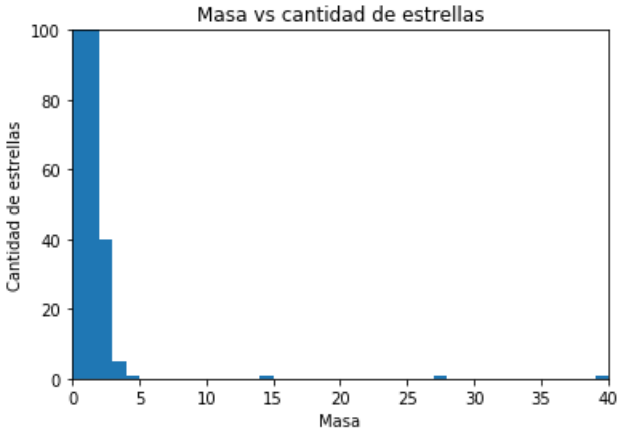


Ayuda: Puede configurar el tamaño de la figura con el parámetro figsize=(8,8) al momento de crear la figura. Por ejemplo: matplotlib.pyplot.figure(figsize=(8, 8)). Para que los valores que se muestran en el pie correspondan a porcentajes, se puede usar el parámetro: autopct='%1.1f%%' al momento de crear el pie.

Parte 4: Estudiar la masa de las estrellas más cercanas a los planetas

En esta parte se trata de generar una gráfica KDE para analizar la masa de las estrellas más cercanas a los planetas y comparar dicha gráfica con un histograma de la misma información, con el propósito de corroborar la similitud de ambas gráficas. La siguiente figura muestra la apariencia de las dos gráficas:





Ayuda:

- Configure el rango del eje x de las dos figuras entre 0 y 40 (donde 40 es el valor máximo de masa). Para esto utilice el método la función xlim así: matplotlib.pyplot.xlim(0,40).
- Configure el rango del eje y del histograma entre 0 y 100. Para esto utilice el método la función ylim así: matplotlib.pyplot.ylim(0,40).

Parte 5: Graficar y analizar el cielo

En esta última parte de la aplicación se desea utilizar las coordenadas RA-DEC de las posiciones de los planetas en el cielo (obtenidas a partir de las columnas RA y DEC del archivo) para graficar los planetas en una imagen representada por una matriz. Cada planeta será representado por un pixel en la imagen (una casilla de la matriz), cuyo color RGB

(Red, Green, Blue) dependerá del tipo de detección que se utilizó para descubrirlo, de acuerdo con la siguiente convención:

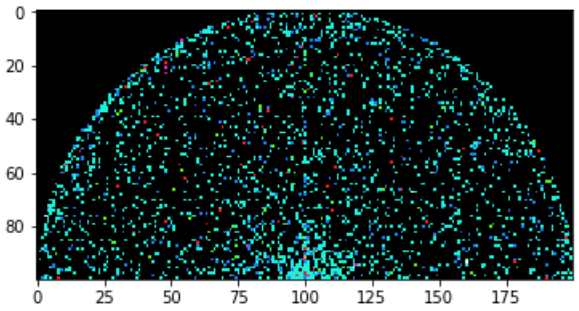
Tipo de detección	Valores RGB	Color
Microlensing	[0.94,0.10,0.10]	
Radial Velocity	[0.1,0.5,0.94]	
Imaging	[0.34,0.94,0.10]	
Primary Transit	[0.10,0.94,0.85]	
Other	[0.94,0.10,0.85]	
Astrometry	[0.94,0.65,0.10]	
TTV	[1.0,1.0,1.0]	

Lo primero que usted debe hacer en este punto es crear una imagen de 100 filas x 200 columnas, con píxeles de color negro. Sobre esta imagen de base se empezarán a cambiar los colores de los píxeles de acuerdo a la posición de cada planeta y el color que le corresponde. Graficar un planeta consiste entonces en cambiar el color del píxel en una casilla de la imagen, cuyas coordenadas (fila y columna) son calculadas a partir de las coordenadas RA-DEC del planeta, de la siguiente manera:

Para un planeta *p* cuyas coordenadas RA-DEC son (RA_p, DEC_p), las coordenadas del píxel correspondiente en la matriz que representa la imagen están dadas por las siguientes ecuaciones:

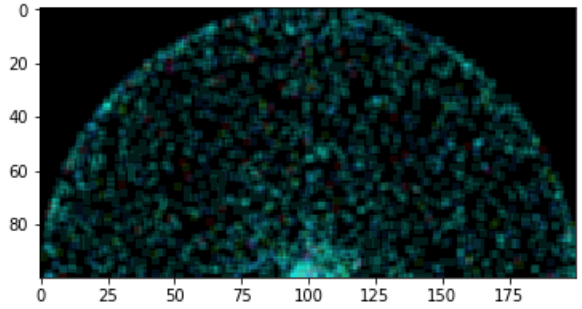
$$Fila_p = 99 - |sin(RA_p) * cos(DEC_p) * 100|$$
$$Columna_p = [cos(RA_p) * cos(DEC_p) * 100] + 100$$

La siguiente figura muestra la apariencia de la imagen de los planetas:

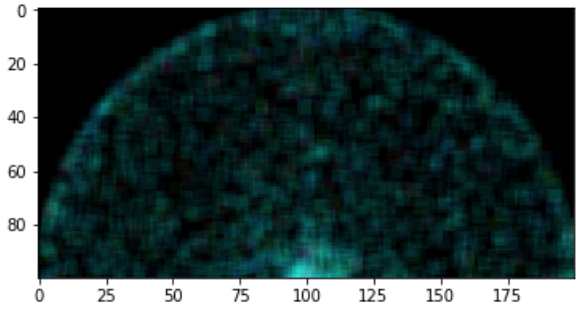


Su programa debe permitirle al usuario crear y visualizar esta imagen. Adicionalmente se desea ofrecer al usuario la posibilidad de filtrar la imagen aplicando una operación de convolución con una máscara de convolución de tamaño definido por el usuario y con todos los coeficientes iguales a 1. Para esto, usted debe implementar una función de convolución que reciba como parámetro la imagen de los planetas y el tamaño definido por el usuario y retorne la imagen filtrada. A manera de ejemplo, las siguientes figuras muestran la imagen filtrada con unas máscaras de 3x3, 5x5 y 7x7 respectivamente, todas con coeficientes iguales a 1.

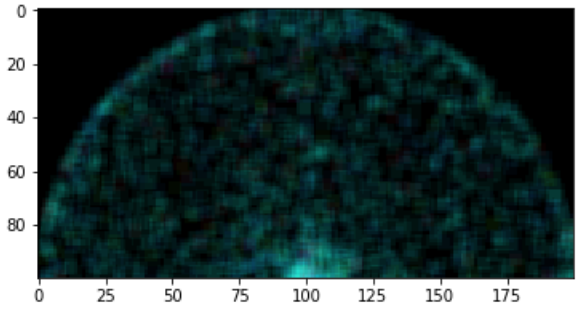
Con una máscara de convolución de 3x3 = [[1,1,1],[1,1,1],[1,1,1]] la imagen filtrada resultante tiene la siguiente apariencia:



Con una máscara de convolución de 5x5 = [[1,1,1,1,1],[1,1,1,1,1],[1,1,1,1,1],[1,1,1,1,1],[1,1,1,1,1]] la imagen filtrada resultante tiene la siguiente apariencia:



Con una máscara de convolución de 7x7 = [[1,1,1,1,1,1,1],[1,1,1,1,1,1,1],[1,1,1,1,1,1,1],[1,1,1,1,1,1,1],[1,1,1,1,1,1,1],[1,1,1,1,1,1,1],[1,1,1,1,1,1,1]] la imagen filtrada resultante tiene la siguiente apariencia:



Actividad 0: Preparación del ambiente de trabajo

1. Cree una carpeta para trabajar, poniéndole su nombre o login.
2. Descargue de BrightSpace el archivo “exoplanetas.csv”, el cual contiene los datos que se van a procesar.
3. Abra Spyder y cambie la carpeta de trabajo para que sea la carpeta donde descargó el archivo con los datos.

Actividad 1: Construir el módulo de funciones

Usando Spyder, cree en su carpeta de trabajo un nuevo archivo con el nombre “exoplanetas.py”. En este archivo usted va a construir el módulo en el que va a implementar las funciones que responden a los requerimientos de la aplicación. **Defina**, **documente** e **implemente** las funciones en su nuevo archivo. Usted puede crear cuántas funciones considere necesarias dentro de su librería o módulo. Mínimo debe haber una función por cada una de las acciones que debe realizar el programa.

Actividad 2: Construir la interfaz de usuario basada en consola

El archivo “consola.py” debe seguir la misma estructura de las consolas que hemos implementado en laboratorios y proyectos anteriores. Esto es: debe existir una función llamada iniciar_aplicacion() para que muestre el menú usando la función mostrar_menu() y permita al usuario seleccionar una opción. El menú que se despliegue debe permitir al usuario ejecutar todas las acciones de la descripción de la aplicación, así como salir (terminar) del programa. Por cada una de las funciones principales de su programa, debe existir una función en la consola que la ejecute, pidiendo previamente los datos necesarios al usuario (si aplica) e imprimiendo por pantalla el resultado de la función. Se sugiere nombrar estas funciones como ejecutar_XX, donde XX es la respectiva función de su módulo.

Cuando haya implementado la función iniciar_aplicacion() corra su programa y verifique que se comporta adecuadamente, permite al usuario seleccionar las opciones que se quieren ejecutar, y termine el programa cuando se le indique.

Actividad 3: Probar el correcto funcionamiento de su programa

Puede probar el correcto funcionamiento de su programa cargando la información que se encuentra en el archivo "exoplanetas.py" o creando su propio archivo de prueba de menor tamaño (respetando el mismo formato) que le permita corroborar que los resultados arrojados por su programa son correctos.

Entrega

1. Comprima los dos archivos: exoplanetas.py y consola.py en un solo archivo .zip. El archivo comprimido debe llamarse "**N4-PROY-login.zip**", donde login es su nombre de usuario de Uniandes.
2. Entregue el archivo comprimido a través de BrightSpace en la actividad designada como "**Proyecto N4**".