

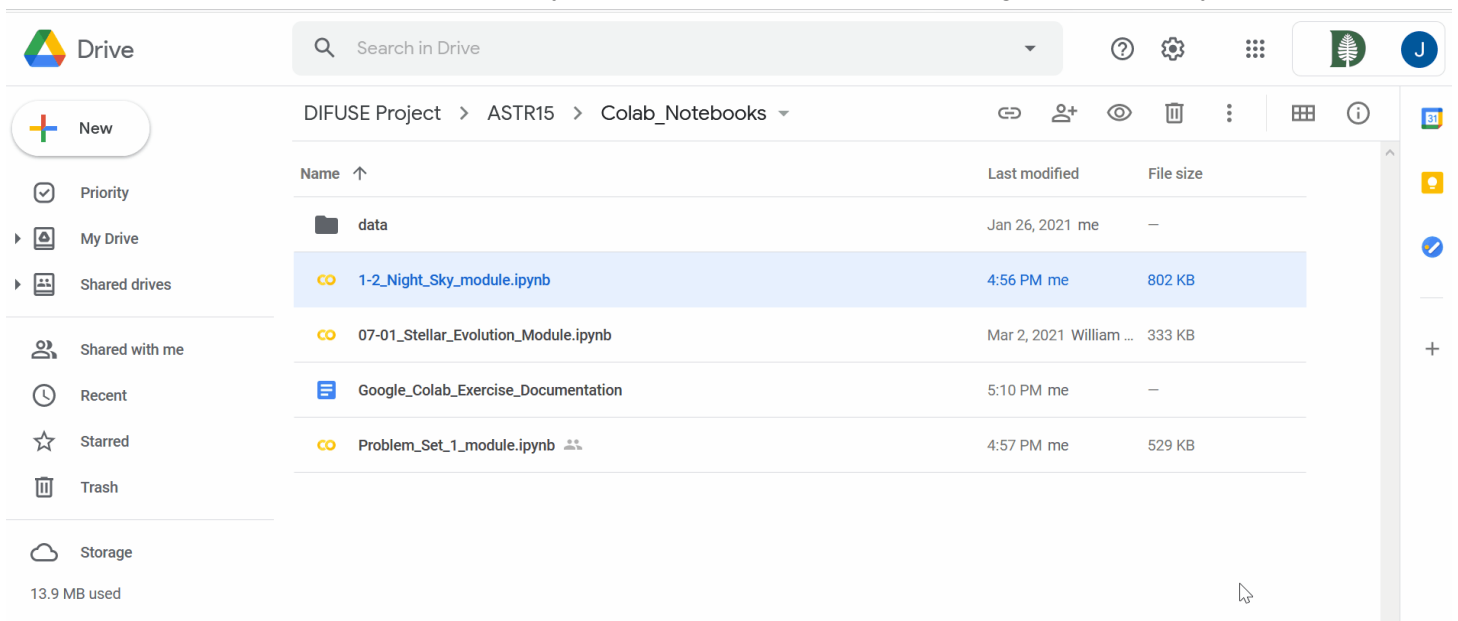
Documentation for Google Colab Exercises

Instructions for using Google Colab Notebook

Google Colab is a tool which can be used to run Jupyter Notebooks or create stand-alone notebooks for writing, editing, and running code in Python. The Colab Notebooks are run on Google's servers, which means you don't have to install anything on your own computer and also gives you access to GPU support!

Opening the Notebook in Google Colab

- First, you must save a copy in your own Google Drive folder
- Then double click on the .ipynb file and select "Open with Google Colaboratory"



The screenshot shows the Google Drive web interface. The left sidebar contains navigation options: 'New', 'Priority', 'My Drive', 'Shared drives', 'Shared with me', 'Recent', 'Starred', 'Trash', and 'Storage' (13.9 MB used). The main area displays the 'DIFUSE Project > ASTR15 > Colab_Notebooks' folder. A table lists the files in this folder:

Name	Last modified	File size
data	Jan 26, 2021 me	—
1-2_Night_Sky_module.ipynb	4:56 PM me	802 KB
07-01_Stellar_Evolution_Module.ipynb	Mar 2, 2021 William ...	333 KB
Google_Colab_Exercise_Documentation	5:10 PM me	—
Problem_Set_1_module.ipynb	4:57 PM me	529 KB

Blocks consist of **text** or **code**:

- If you want to edit a text block (e.g., for answering a question), simply double-click the block or click “SHOW CODE.”
- Similarly, if you want to edit a code block or see hidden code, double-click the code block. You can also hide/show the code by right clicking on the block, selecting “Forms,” and then clicking “show code” or “hide code.”

Problem_Set_1_module.ipynb

File Edit View Insert Runtime Tools Help

+ Code + Text

RAM Disk

Editing

Problem Set 1

Instructions: Save a copy of this Colab notebook to your own Google Drive folder **adding your name to the filename** and open in Google Colaboratory. When you have completed the exercise you will need to upload a .pdf of the Colab notebook to canvas. Instructions on how to download a .pdf of this notebook are found at the end of the exercise.

Documentation: Additional resources on how to use Google Colab, enter answers, view code, and which Python libraries are used can be found in the documentation file.

Honor Statement: Recall that your name on this worksheet is considered to be a reaffirmation of your commitment to the Dartmouth Honor Principle. No online resources, phones, or calculators are allowed on these worksheets, although you may use your textbooks and notes.

Please fill in your name below by double clicking the text

Running blocks of code:

- Blocks of code have a play button on the left hand side - to run the code, just press the play button, or you can select the block and press Shift+Enter.
- If you would like to run all blocks of code for the entire Notebook, you can select the Runtime dropdown menu at the top of the screen and “Run all” or “Restart and Run all”

Problem_Set_1_module.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

Editing

[2] Input the date and time for the calculations by clicking the arrows on the date chooser. Input time by typing it into the form. **Note:** time MUST be in HH:MM:SS 24-hr format

obs_date: 2020 / 04 / 01

obs_time: 09:00:00

You successfully set your observation date and time to: 2020-04-01 09:00:00

Step 2 Now select 'Shattuck Observatory, Dartmouth' and run the block of code

[3] Choose the observatory from the dropdown that you'd like to use

obs_choice: Kitt Peak Observatory, Arizona

You successfully set your observatory choice to Kitt Peak Observatory, Arizona

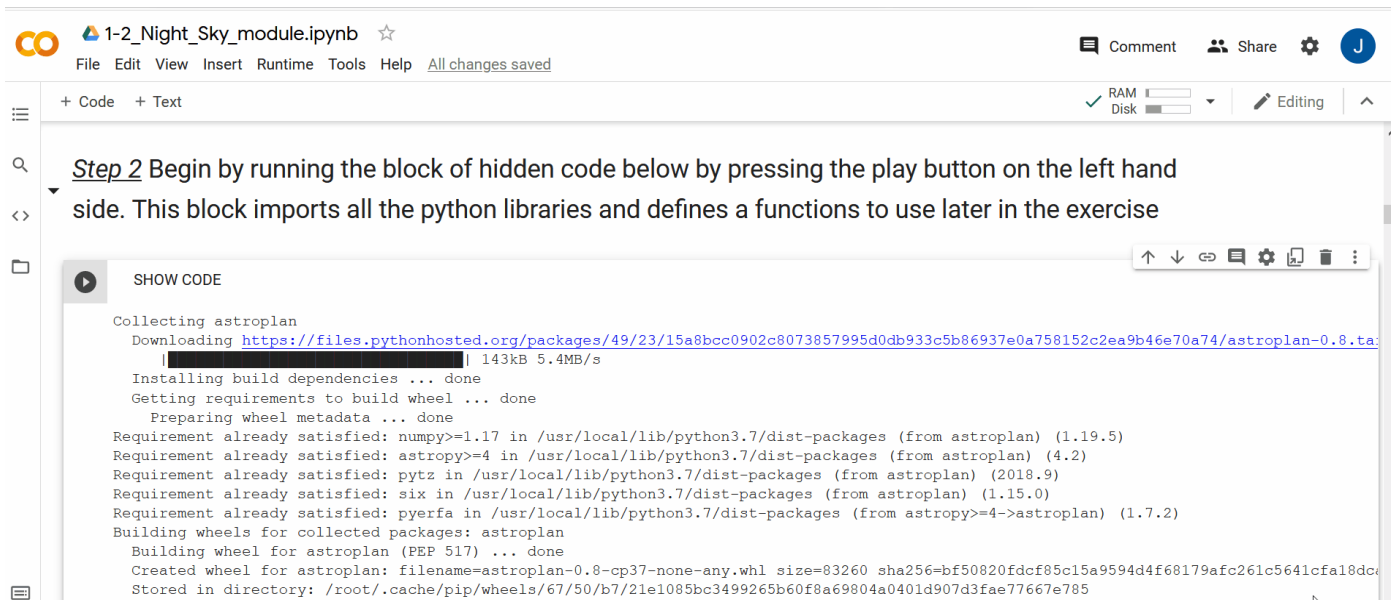
Inputting information in Forms:

- Some of the blocks make use of Forms, which consist of an interactive interface that is meant for the user to input text, numbers/dates, or select from a list of options
- For *text input*, click on the empty line and type in the necessary information
- For *number or date inputs*, arrows allow you to change the number/date or you can manually type in the necessary information
- For *dropdown menus*, simply click and select the option you'd like to choose
- **NOTE:** once you have input the information into the forms, *you must run the block of code* for the information you input to be stored as variables and for the functions to be performed

The screenshot shows a Jupyter Notebook window titled "1-2_Night_Sky_module.ipynb". The interface includes a top menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help". On the right, there are buttons for "Comment", "Share", and a user profile icon. Below the menu, a toolbar shows "+ Code" and "+ Text". The main area displays a code cell with the text "you can ignore it." followed by a form block. The form block has a title "Number of group members" and a text input field containing "number_group_members: 4". Below the input field, a message states "You have successfully set the size of your group!". The form block also includes a toolbar with icons for undo, redo, copy, paste, and other actions. Below the form block, there is a code cell with the text "[2] Type the names of your group members here". This cell contains four text input fields, each labeled "group_member_1_name:", "group_member_2_name:", "group_member_3_name:", and "group_member_4_name:", each with a placeholder "Insert text here".

Examining hidden code:

- All of the code is hidden in this Colab Notebook to make the document easier to read and look through while completing the exercises
- If you'd like to examine the code, click the "SHOW CODE" button
 - *Be careful not to accidentally modify the code or you might end up with an unexpected error or "bug"*
- To hide the code again, right click the block, go to Forms, and select "Hide code"

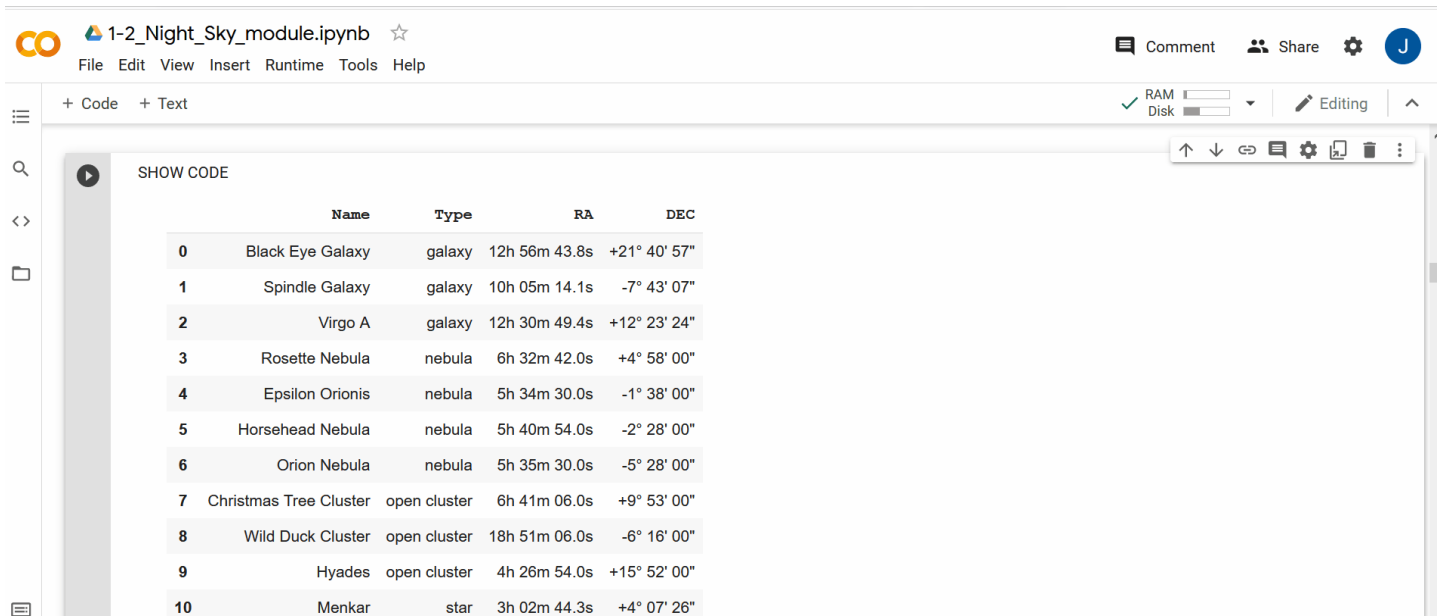


The screenshot shows a Google Colab notebook interface. At the top, the title bar reads "1-2_Night_Sky_module.ipynb" with a star icon. Below it is a menu bar with "File", "Edit", "View", "Insert", "Runtime", "Tools", and "Help", followed by the text "All changes saved". On the right side of the title bar are icons for "Comment", "Share", a settings gear, and a user profile icon labeled "J". Below the title bar is a toolbar with "+ Code" and "+ Text" buttons, and a status bar showing "RAM" and "Disk" usage. The main content area shows a code block with a "SHOW CODE" button on the left. The code block contains the following text:

```
Collecting astroplan
  Downloading https://files.pythonhosted.org/packages/49/23/15a8bcc0902c8073857995d0db933c5b86937e0a758152c2ea9b46e70a74/astroplan-0.8.ta
  143kB 5.4MB/s
Installing build dependencies ... done
Getting requirements to build wheel ... done
Preparing wheel metadata ... done
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.7/dist-packages (from astroplan) (1.19.5)
Requirement already satisfied: astropy>=4 in /usr/local/lib/python3.7/dist-packages (from astroplan) (4.2)
Requirement already satisfied: pytz in /usr/local/lib/python3.7/dist-packages (from astroplan) (2018.9)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from astroplan) (1.15.0)
Requirement already satisfied: pyerfa in /usr/local/lib/python3.7/dist-packages (from astropy>=4->astroplan) (1.7.2)
Building wheels for collected packages: astroplan
  Building wheel for astroplan (PEP 517) ... done
  Created wheel for astroplan: filename=astroplan-0.8-cp37-none-any.whl size=83260 sha256=bf50820f85c15a9594d4f68179afc261c5641cfa18dc
  Stored in directory: /root/.cache/pip/wheels/67/50/b7/21e1085bc3499265b60f8a69804a0401d907d3fae77667e785
```

Dealing with Errors:

- If you enter data or information into forms which are in an unacceptable format or are out of the range of acceptable values you will see an error
- Do not fear! Look carefully at the information you have entered and look at the error message for clues about what may have triggered it
- If you believe the error might be because you have accidentally altered the code, try downloading a new version of the exercise and start over



The screenshot shows a Jupyter Notebook window titled "1-2_Night_Sky_module.ipynb". The interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help), a toolbar with icons for code and text cells, and a status bar showing RAM and disk usage. The main content area displays a table of celestial objects with columns for Name, Type, RA, and DEC. The table contains 11 rows of data, indexed from 0 to 10.

	Name	Type	RA	DEC
0	Black Eye Galaxy	galaxy	12h 56m 43.8s	+21° 40' 57"
1	Spindle Galaxy	galaxy	10h 05m 14.1s	-7° 43' 07"
2	Virgo A	galaxy	12h 30m 49.4s	+12° 23' 24"
3	Rosette Nebula	nebula	6h 32m 42.0s	+4° 58' 00"
4	Epsilon Orionis	nebula	5h 34m 30.0s	-1° 38' 00"
5	Horsehead Nebula	nebula	5h 40m 54.0s	-2° 28' 00"
6	Orion Nebula	nebula	5h 35m 30.0s	-5° 28' 00"
7	Christmas Tree Cluster	open cluster	6h 41m 06.0s	+9° 53' 00"
8	Wild Duck Cluster	open cluster	18h 51m 06.0s	-6° 16' 00"
9	Hyades	open cluster	4h 26m 54.0s	+15° 52' 00"
10	Menkar	star	3h 02m 44.3s	+4° 07' 26"

Python Libraries Used (linked to webpages for additional documentation)

Pandas: easy to use data organization, analysis, and manipulation tool. Primarily used for creating, viewing, and accessing *dataframes*, which is just a Pandas version of a table.

Numpy: one of the fundamental packages for scientific computing with Python. Primarily used in the creation and manipulation of *arrays*, which are just matrices.

Matplotlib: a fundamental package for plotting and visualization of data in Python. All the plots are constructed using matplotlib.

Datetime: a Python module for manipulating dates and times in Python. Used for performing operations on dates and times (makes time and date math a lot easier!).

Astropy: the primary Python library for everything Astronomy! We use Astropy for:

1. Retrieving the RA and Dec of celestial objects
2. Calculating the Altitude and Azimuth of celestial objects at different times based on location

3. Creating and manipulating *SkyCoord* objects: high-level objects that provide a flexible interface for celestial coordinate representation, manipulation, and transformation between systems
4. Creating and manipulating *EarthLocation* and *Observer* objects: used for storing a location on the Earth and for storing information about an observer's location and environment, respectively
5. Creating and manipulating *FixedTarget* objects: contain coordinates and metadata for a celestial object that is "fixed" with respect to the celestial sphere
6. Creating and manipulating *Time* objects: used to represent and manipulate times and dates for astronomy, similar to the Datetime module but specific to the ways time is represented in Astronomy.
7. Creating and manipulating *Time* objects: used to represent and manipulate times and dates for astronomy, similar to the Datetime module but specific to the ways time is represented in Astronomy
8. The *Lomb-Scargle Model*: used to fit a sinusoidal model to data at each frequency, with a larger power reflecting a better fit. With this in mind, it is often helpful to plot the best-fit sinusoid over the phased data.

Astroplan: a library built on top of Astropy that is specifically for planning the observation of celestial objects. Astroplan is primarily used for creating the "SkyPlots" which are a 2-D representation of the night sky for a given location.

1. Creating the "SkyPlots" which are a 2-D representation of the night sky for a given location
2. Creating the Airmass plots, which are an estimation of the relative distance of atmosphere light from a celestial object has to pass through to reach the surface of the Earth

Astroquery: a library built on top of Astropy for querying astronomical web forms and databases. We use Astroquery for retrieving images of your selected objects from an online database.

Sympy: a library for symbolic mathematics in Python. We use Sympy to formulate and solve a fairly simple algebraic equation but can be used in many more complex ways for solving a wide range of mathematical problems.