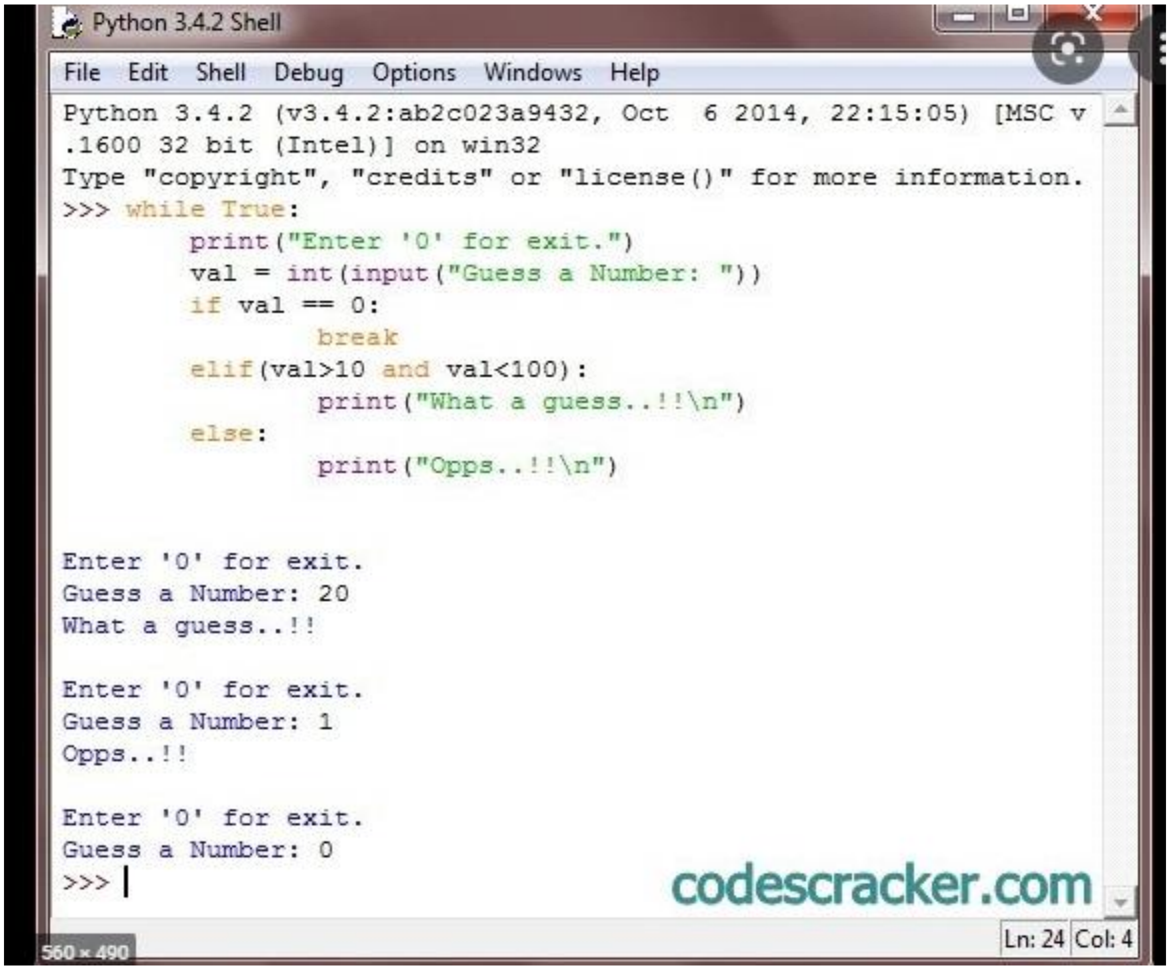
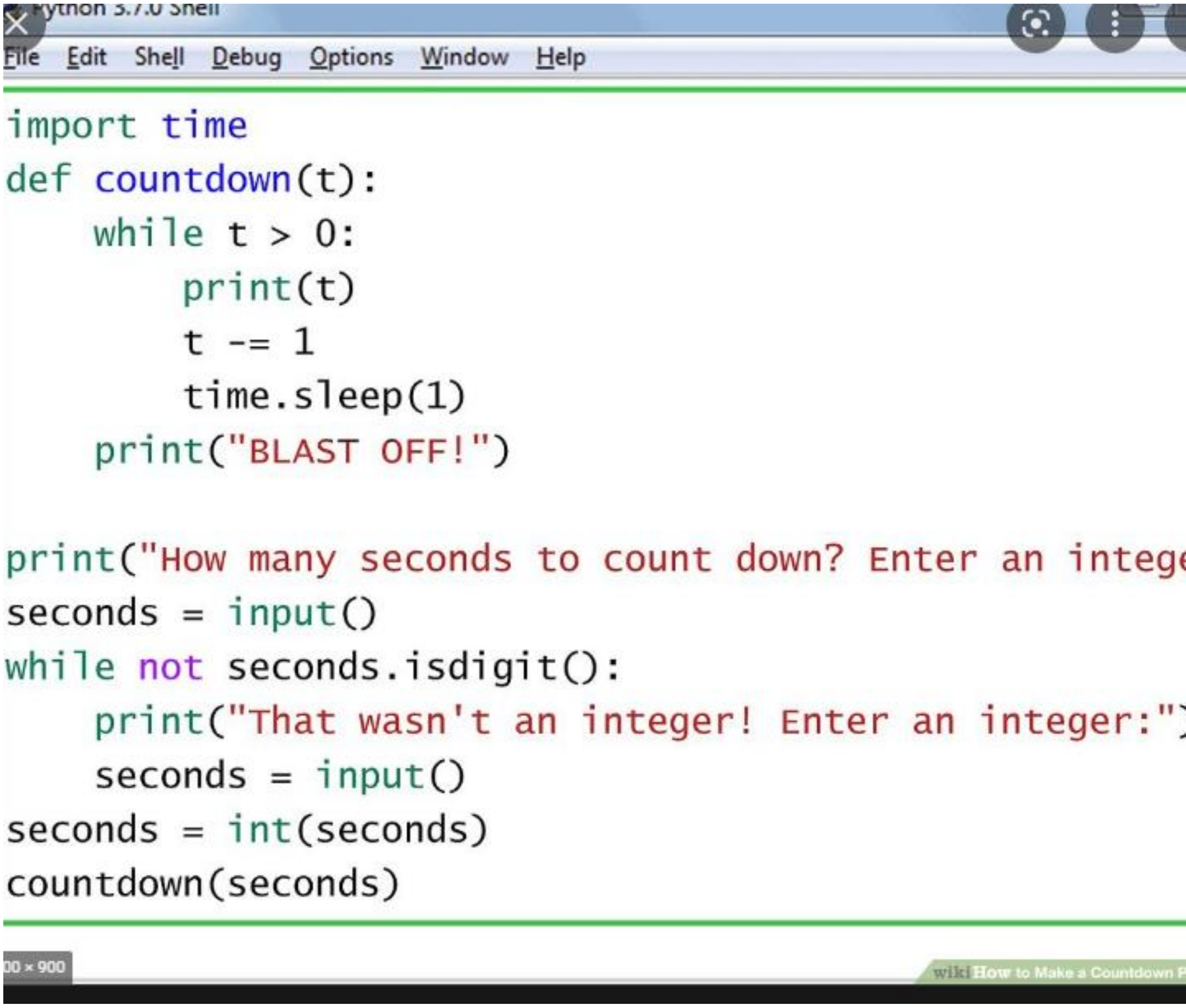
ASSIGNMENT 2

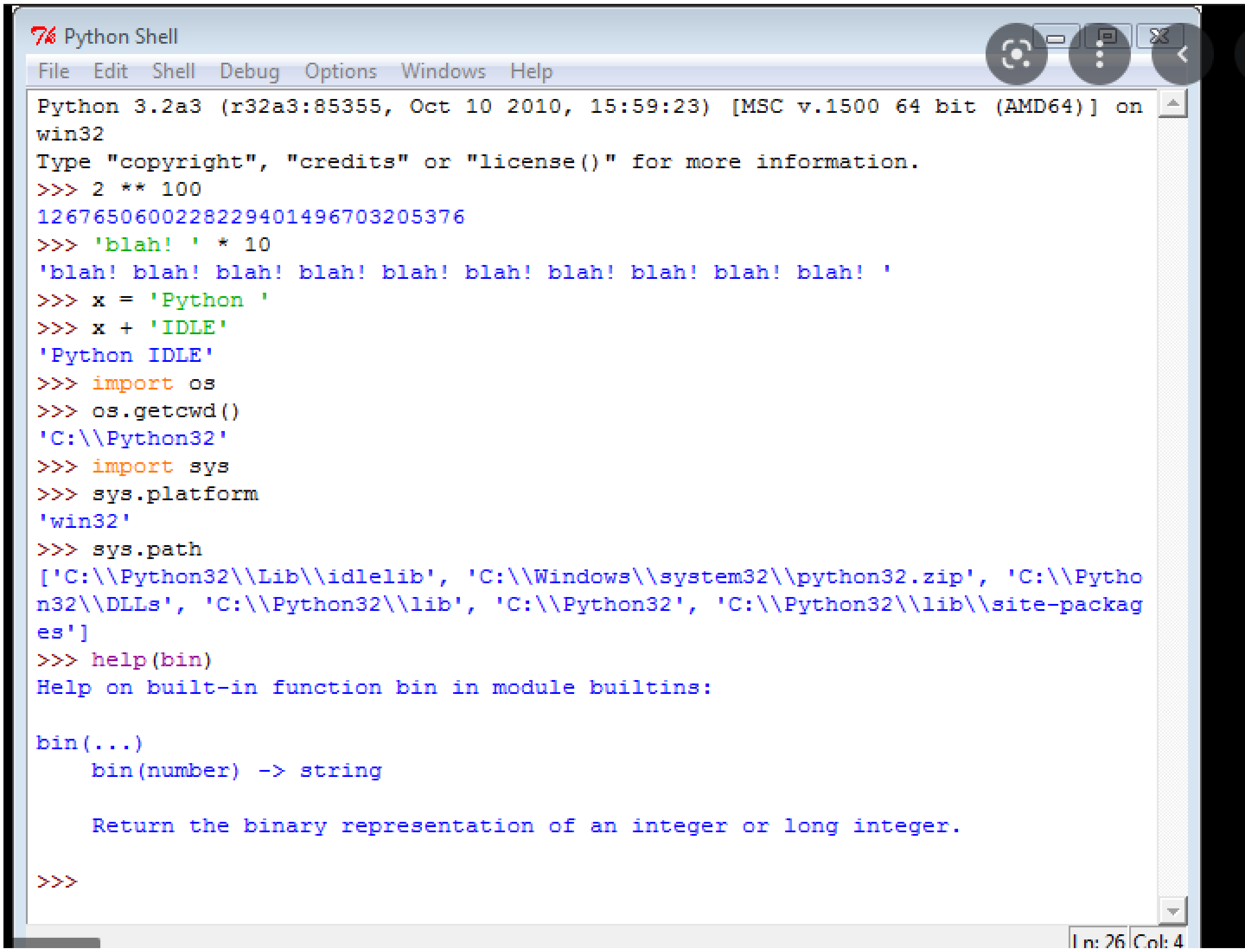
1.PRACTICE PYTHON IN IDLE:



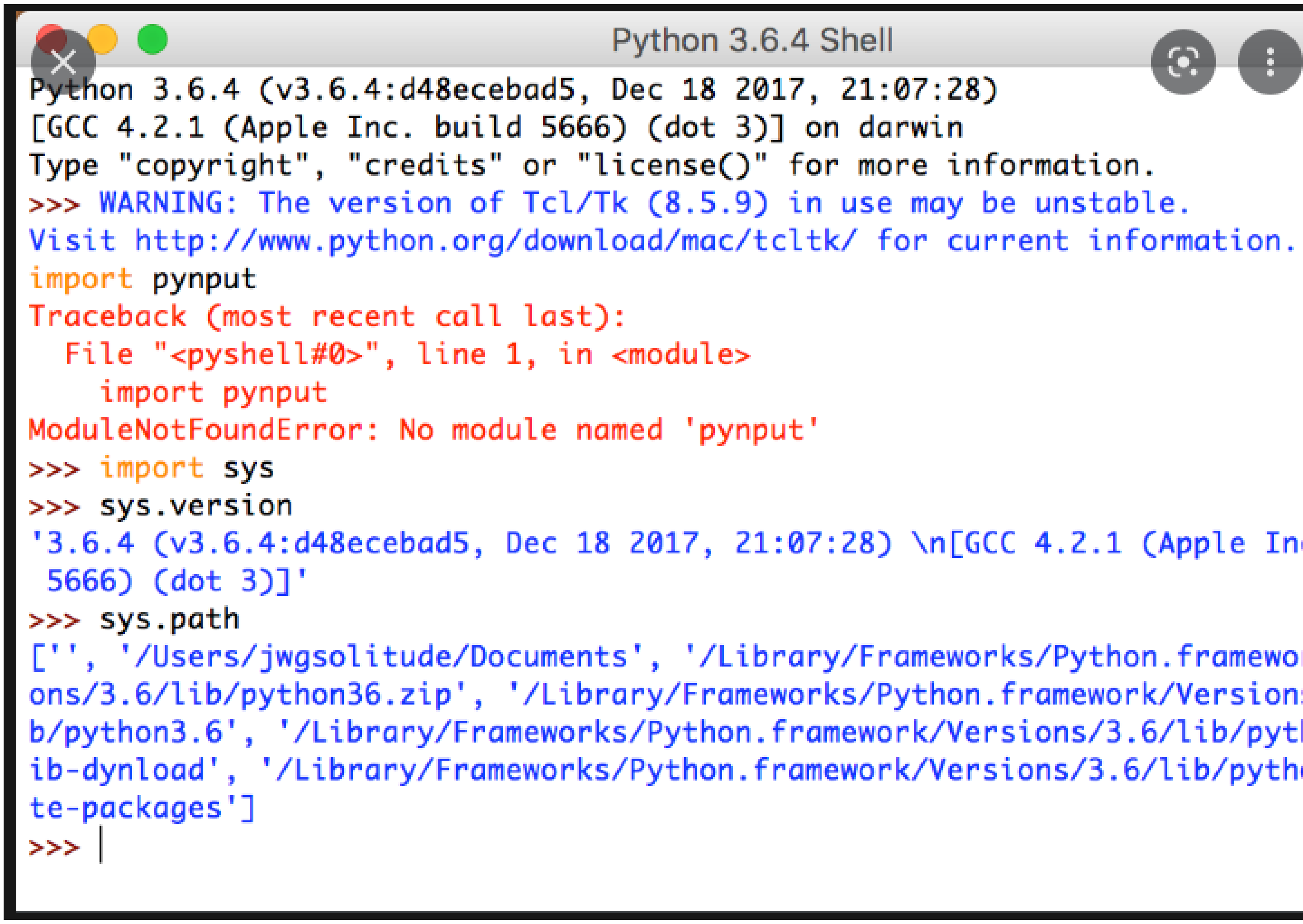
```
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct  6 2014, 22:15:05) [MSC v
.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> while True:
        print("Enter '0' for exit.")
        val = int(input("Guess a Number: "))
        if val == 0:
                break
        elif(val>10 and val<100):
                print("What a guess..!!\n")
        else:
                print("Opps..!!\n")


Enter '0' for exit.
Guess a Number: 20
What a guess..!!

Enter '0' for exit.
Guess a Number: 1
Opps..!!

Enter '0' for exit.
Guess a Number: 0
>>> 
```


Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
00 × 900
wiki How to Make a Countdown P

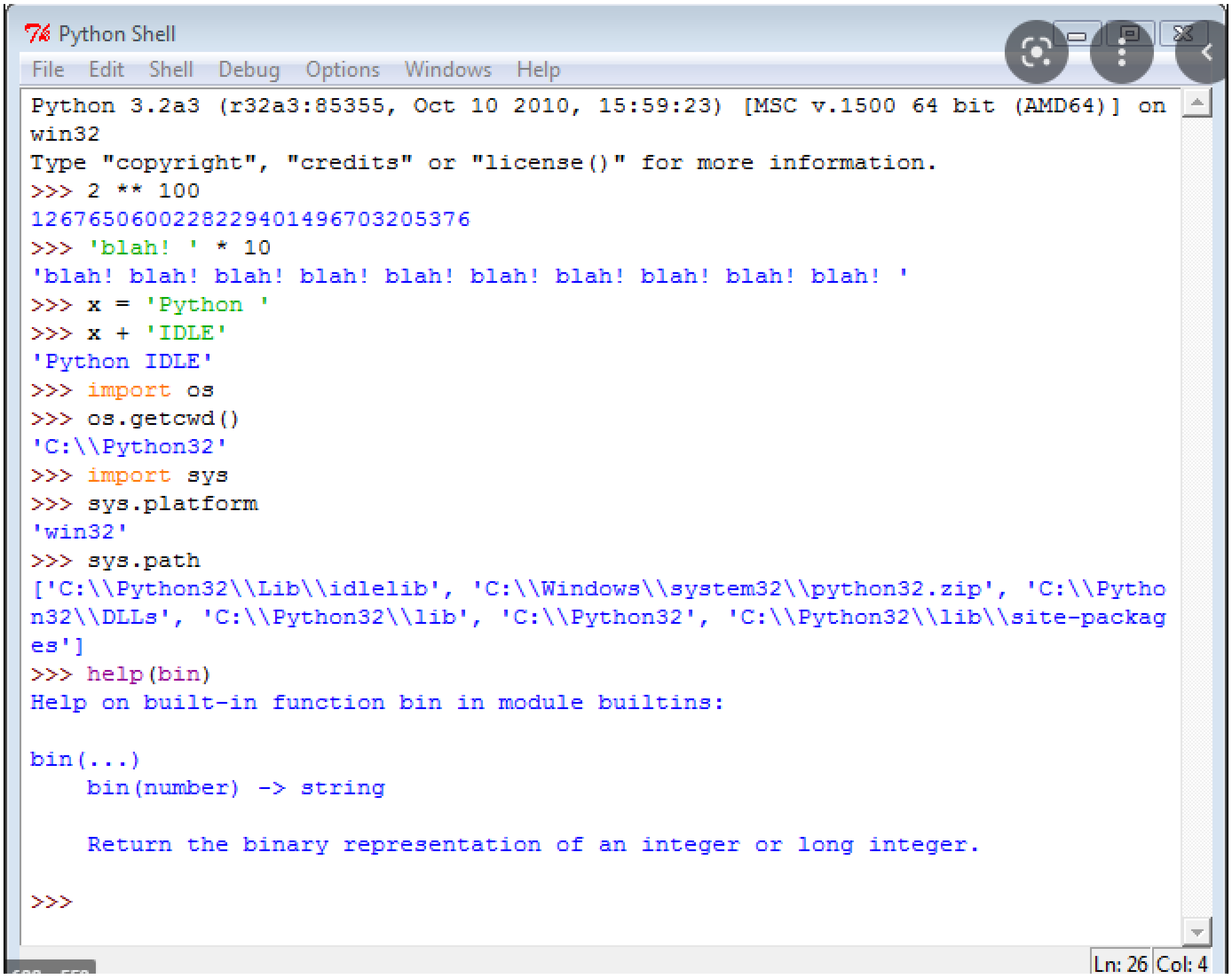
```
import time
def countdown(t):
    while t > 0:
        print(t)
        t -= 1
        time.sleep(1)
    print("BLAST OFF!")

print("How many seconds to count down? Enter an intege
seconds = input()
while not seconds.isdigit():
    print("That wasn't an integer! Enter an integer:"
    seconds = input()
seconds = int(seconds)
countdown(seconds)
```



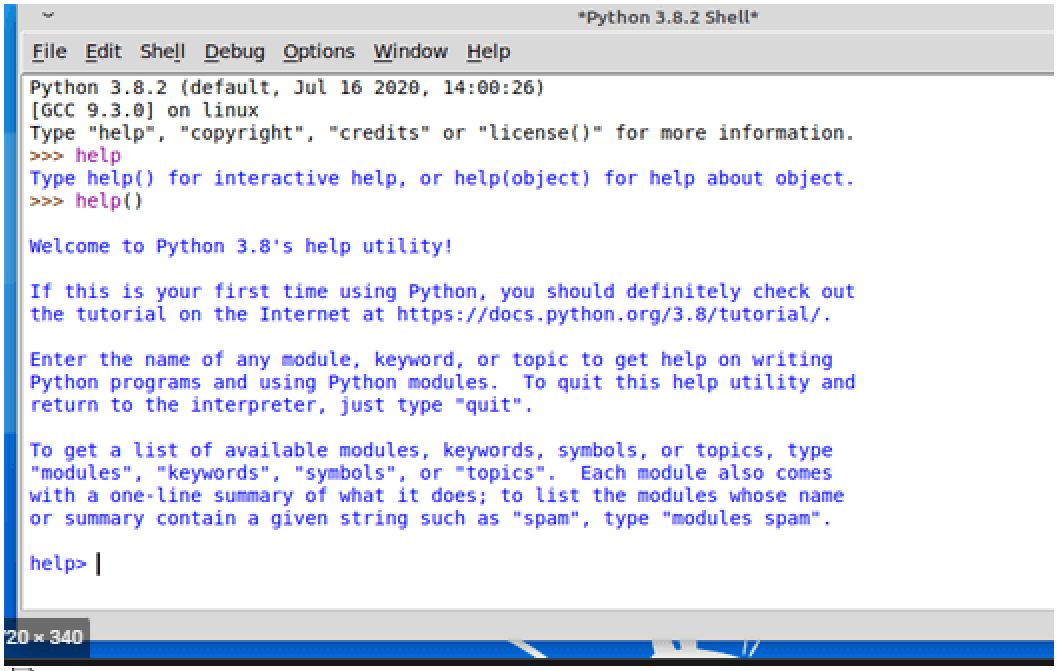
```
Python 3.2a3 (r32a3:85355, Oct 10 2010, 15:59:23) [MSC v.1500 64 bit (AMD64)] on
win32
Type "copyright", "credits" or "license()" for more information.
>>> 2 ** 100
1267650600228229401496703205376
>>> 'blah! ' * 10
'blah! blah! blah! blah! blah! blah! blah! blah! blah! blah! '
>>> x = 'Python '
>>> x + 'IDLE'
'Python IDLE'
>>> import os
>>> os.getcwd()
'C:\\Python32'
>>> import sys
>>> sys.platform
'win32'
>>> sys.path
['C:\\Python32\\Lib\\idlelib', 'C:\\Windows\\system32\\python32.zip', 'C:\\Pytho
n32\\DLLs', 'C:\\Python32\\lib', 'C:\\Python32', 'C:\\Python32\\lib\\site-packag
es']
>>> help(bin)
Help on built-in function bin in module builtins:

bin(...)
    bin(number) -> string

    Return the binary representation of an integer or long integer.

>>>
```



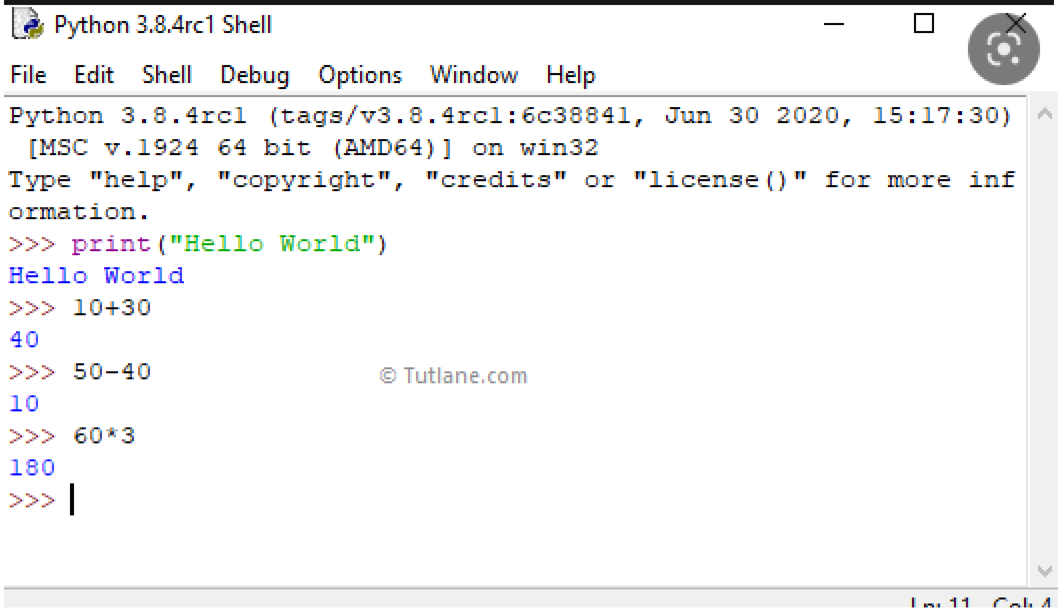
```
Python 3.6.4 (v3.6.4:d48ecebad5, Dec 18 2017, 21:07:28)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> WARNING: The version of Tcl/Tk (8.5.9) in use may be unstable.
Visit http://www.python.org/download/mac/tcltk/ for current information.
import pynput
Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    import pynput
ModuleNotFoundError: No module named 'pynput'
>>> import sys
>>> sys.version
'3.6.4 (v3.6.4:d48ecebad5, Dec 18 2017, 21:07:28) \n[GCC 4.2.1 (Apple In
 5666) (dot 3)]'
>>> sys.path
['', '/Users/jwgsolitude/Documents', '/Library/Frameworks/Python.framewo
ons/3.6/lib/python36.zip', '/Library/Frameworks/Python.framework/Version
b/python3.6', '/Library/Frameworks/Python.framework/Versions/3.6/lib/pyt
ib-dynload', '/Library/Frameworks/Python.framework/Versions/3.6/lib/pyth
te-packages']
>>> |
```



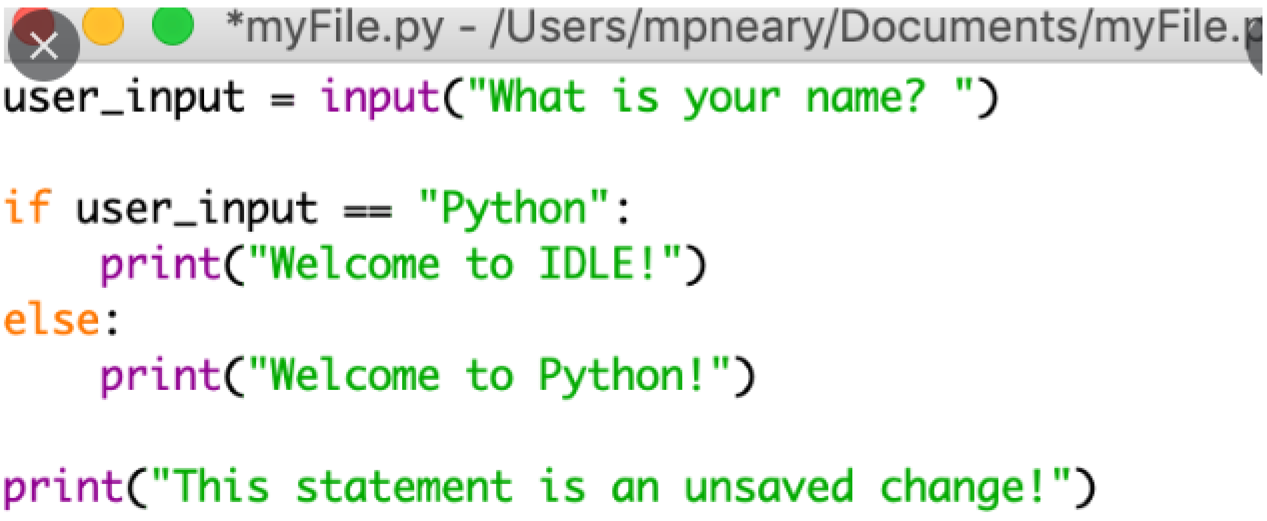
```
Python 3.2a3 (r32a3:85355, Oct 10 2010, 15:59:23) [MSC v.1500 64 bit (AMD64)] on
win32
Type "copyright", "credits" or "license()" for more information.
>>> 2 ** 100
1267650600228229401496703205376
>>> 'blah! ' * 10
'blah! blah! blah! blah! blah! blah! blah! blah! blah! blah! '
>>> x = 'Python '
>>> x + 'IDLE'
'Python IDLE'
>>> import os
>>> os.getcwd()
'C:\\Python32'
>>> import sys
>>> sys.platform
'win32'
>>> sys.path
['C:\\Python32\\Lib\\idlelib', 'C:\\Windows\\system32\\python32.zip', 'C:\\Pytho
n32\\DLLs', 'C:\\Python32\\lib', 'C:\\Python32', 'C:\\Python32\\lib\\site-packag
es']
>>> help(bin)
Help on built-in function bin in module builtins:

bin(...)
    bin(number) -> string

    Return the binary representation of an integer or long integer.

>>>
```



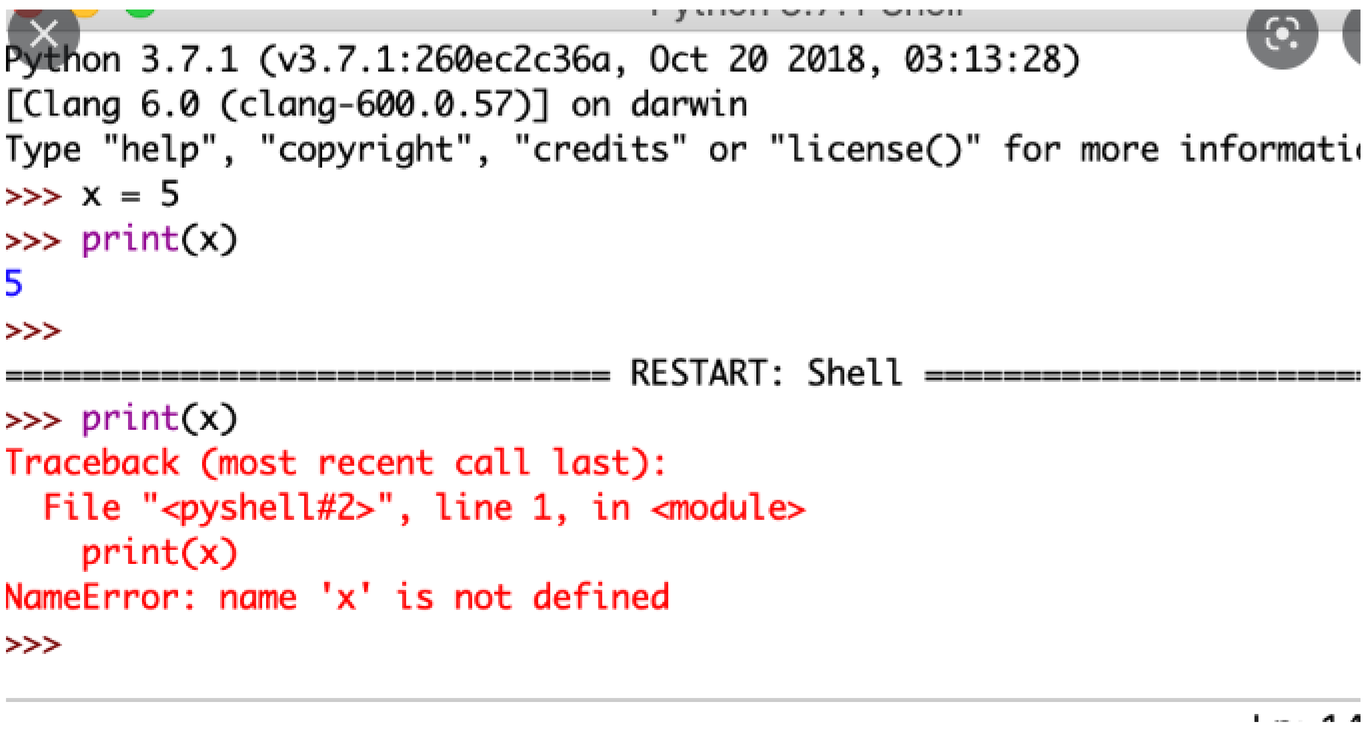
```
Python 3.8.2 (default, Jul 16 2020, 14:00:26)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>> help
Type help() for interactive help, or help(object) for help about object.
>>> help()

Welcome to Python 3.8's help utility!

If this is your first time using Python, you should definitely check out
the tutorial on the Internet at https://docs.python.org/3.8/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To quit this help utility and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics".  Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".

help> 
```



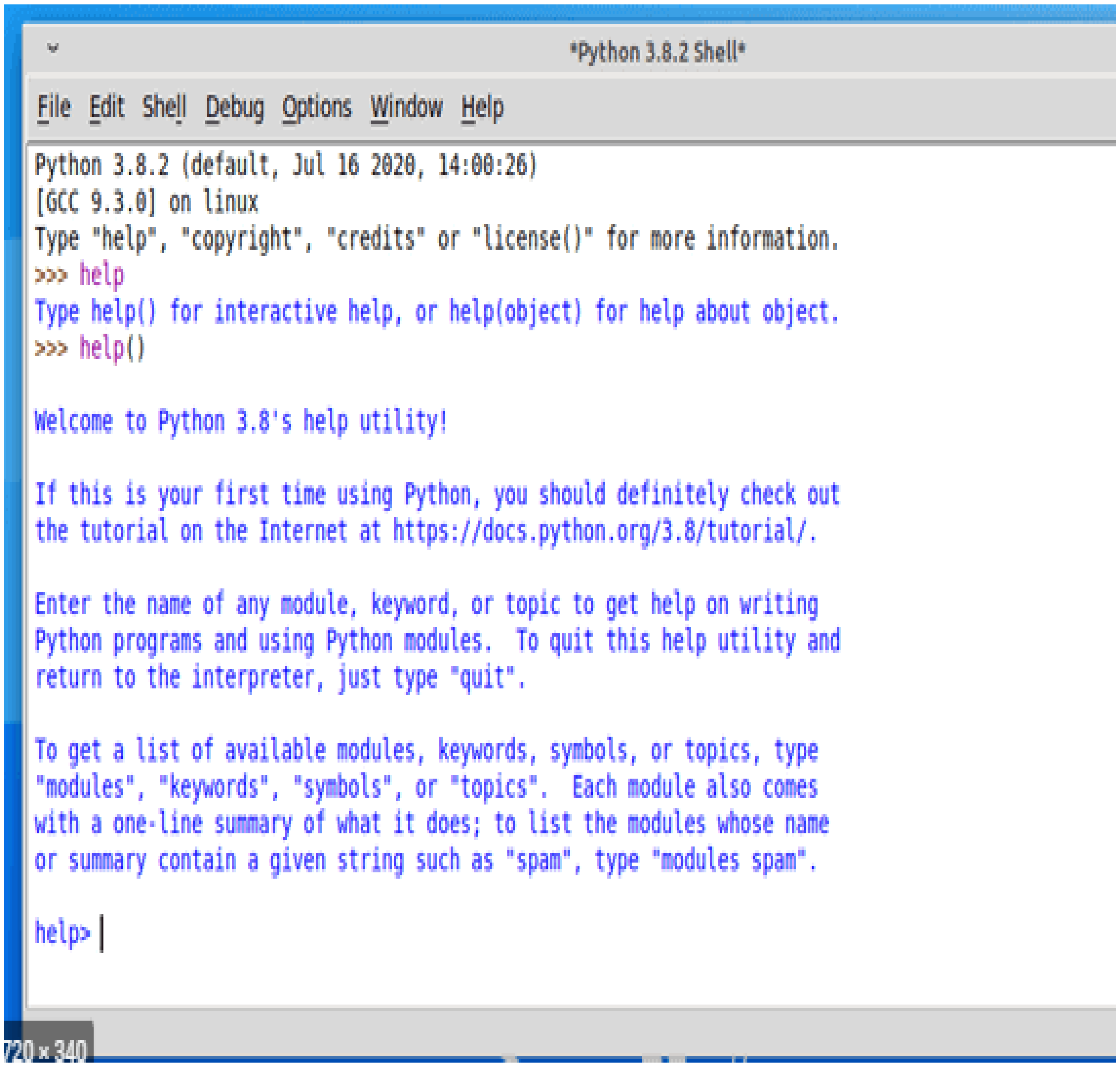
```
Python 3.8.4rc1 (tags/v3.8.4rc1:6c38841, Jun 30 2020, 15:17:30)
 [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more inf
ormation.
>>> print("Hello World")
Hello World
>>> 10+30
40
>>> 50-40
10
>>> 60*3
180
>>> 
```


*myFile.py - /Users/mpneary/Documents/myFile.p

```
user_input = input("What is your name? ")

if user_input == "Python":
    print("Welcome to IDLE!")
else:
    print("Welcome to Python!")

print("This statement is an unsaved change!")
```

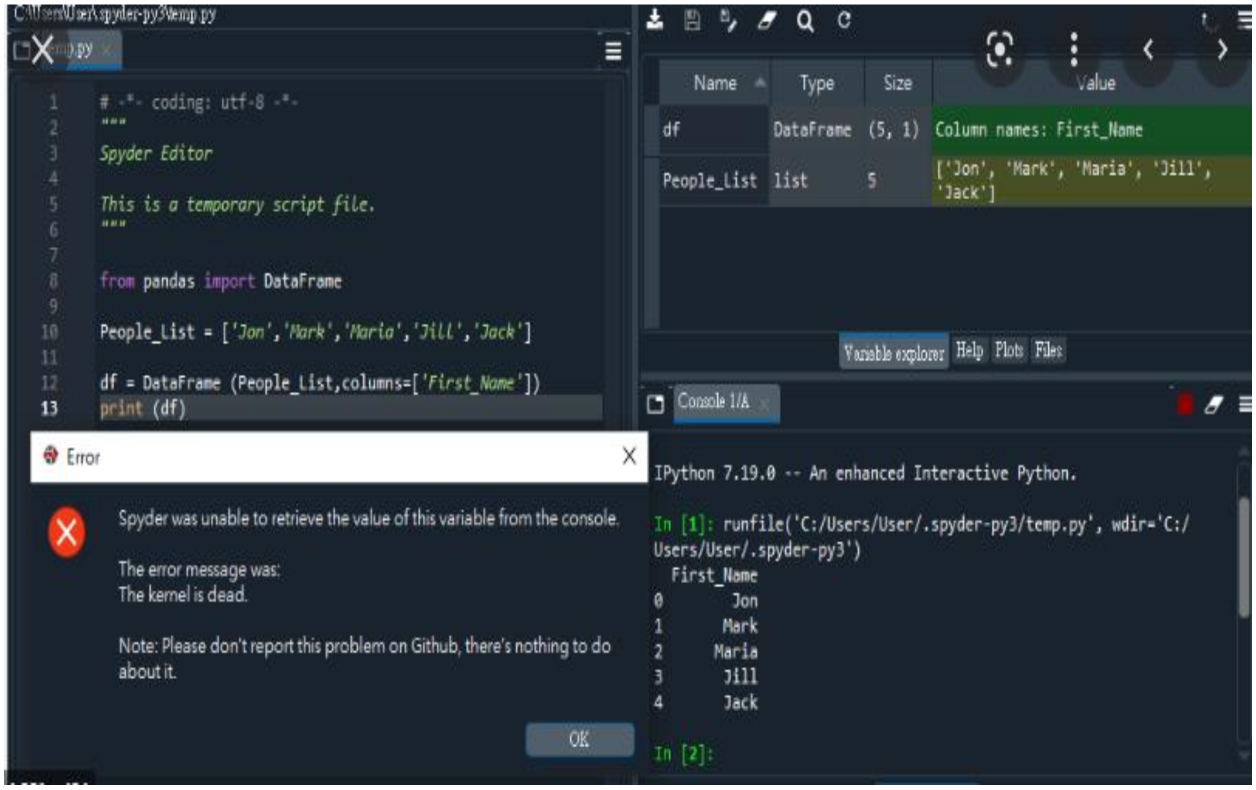


```
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 03:13:28)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license()" for more informati
>>> x = 5
>>> print(x)
5
>>>
================================ RESTART: Shell =======================
>>> print(x)
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    print(x)
NameError: name 'x' is not defined
>>>
```



```
Python 3.8.2 (default, Jul 16 2020, 14:00:26)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license()" for more information.
>>> help
Type help() for interactive help, or help(object) for help about object.
>>> help()

Welcome to Python 3.8's help utility!

If this is your first time using Python, you should definitely check out
the tutorial on the Internet at https://docs.python.org/3.8/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To quit this help utility and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics".  Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".

help> 
```

2.practice python usi



C:\Users\User\.spyder-py3\temp.py

temp.py

```
# -*- coding: utf-8 -*-
"""
Spyder Editor

This is a temporary script file.
"""

from pandas import DataFrame

People_List = ['Jon','Mark','Maria','Jill','Jack']

df = DataFrame (People_List,columns=['First_Name'])
print (df)
```

Name	Type	Size	Value
df	DataFrame	(5, 1)	Column names: First_Name
People_List	list	5	['Jon', 'Mark', 'Maria', 'Jill', 'Jack']

Error

Spyder was unable to retrieve the value of this variable from the console.

The error message was:
The kernel is dead.

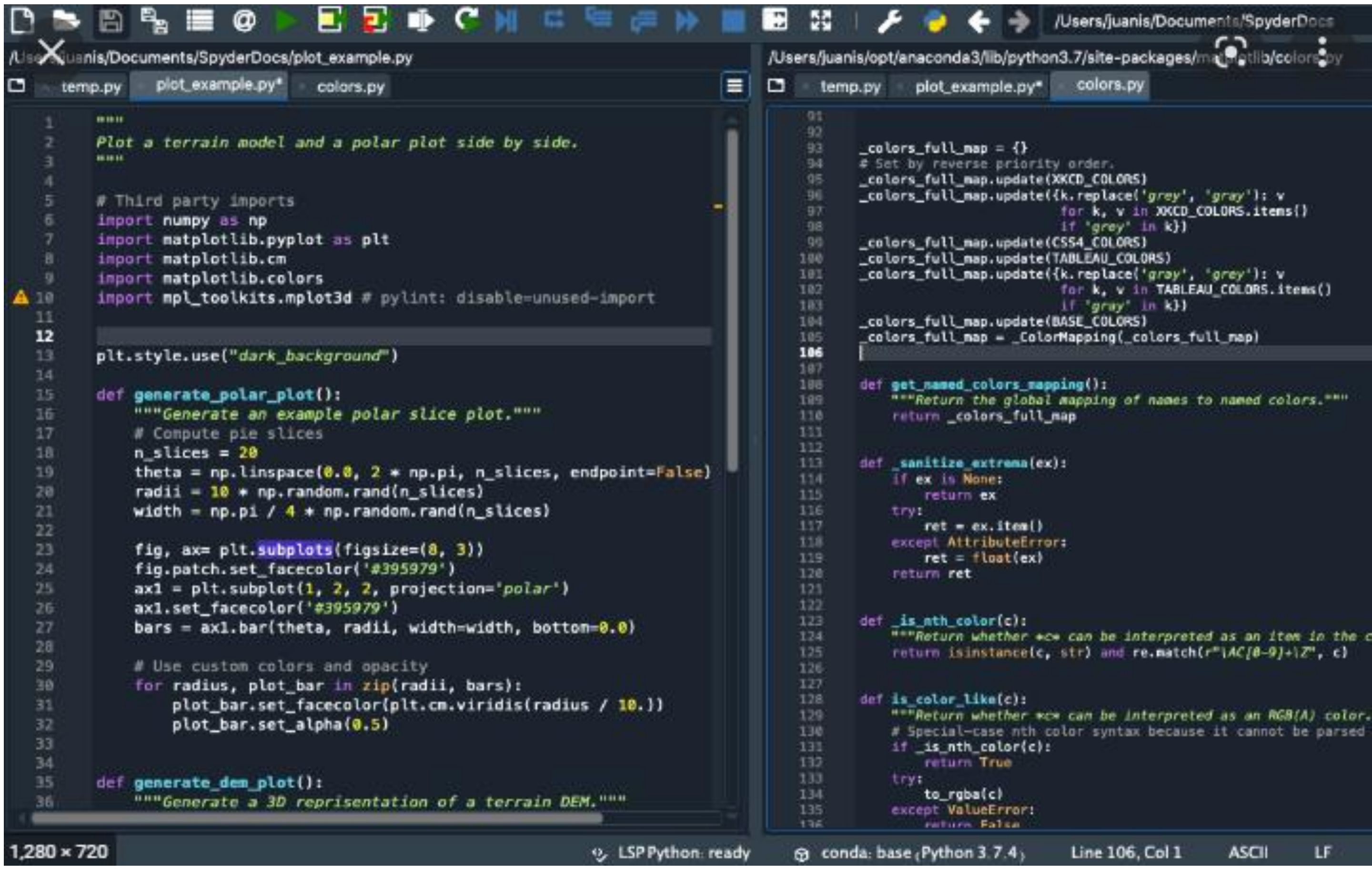
Note: Please don't report this problem on Github, there's nothing to do about it.

OK

Console 1/A

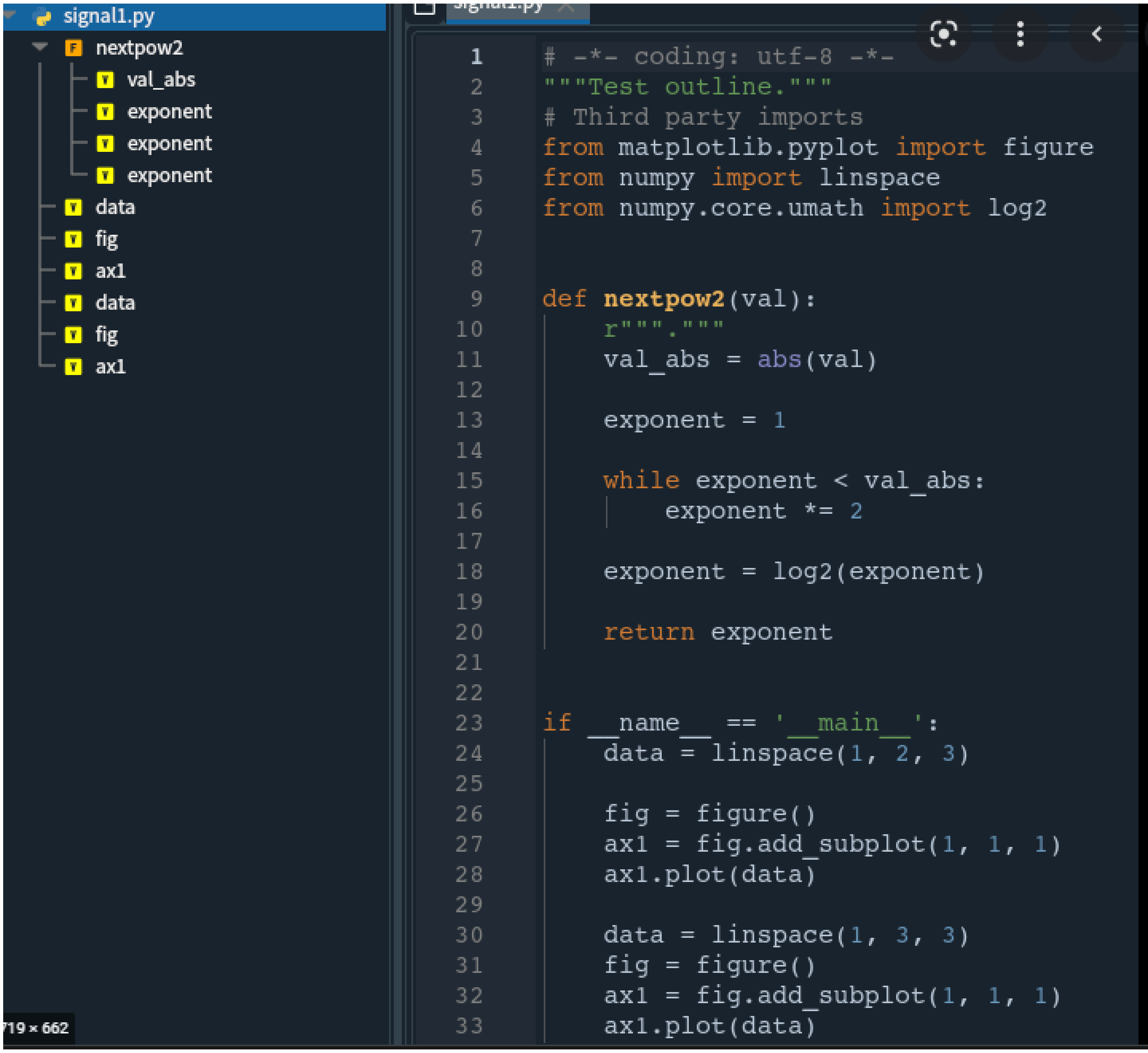
```
IPython 7.19.0 -- An enhanced Interactive Python.

In [1]: runfile('C:/Users/User/.spyder-py3/temp.py', wdir='C:/
Users/User/.spyder-py3')
  First_Name
0        Jon
1       Mark
2      Maria
3       Jill
4       Jack

In [2]:
```

ng spyder

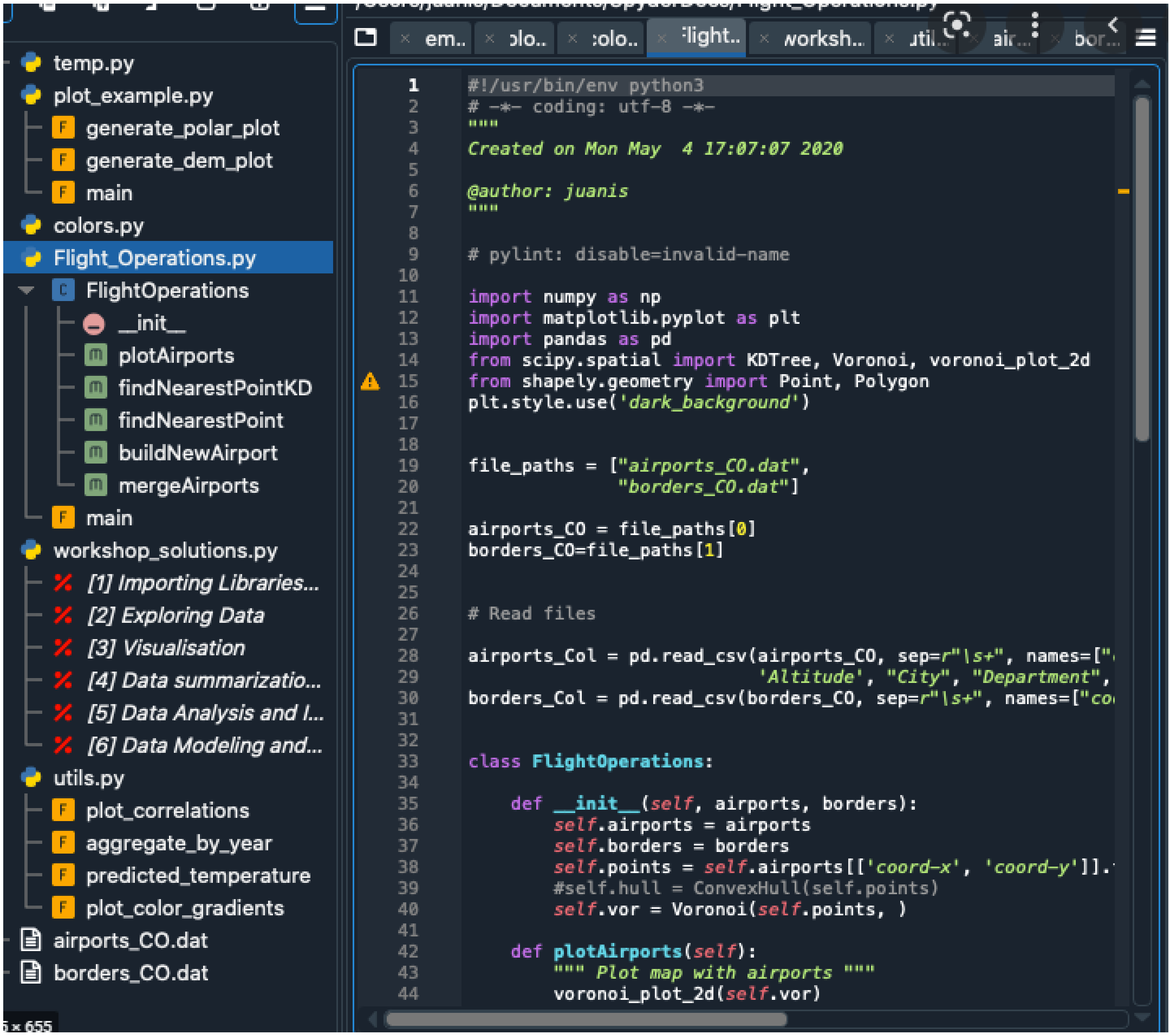

/Users/juanis/Documents/SpyderDocs
/Users/juanis/Documents/SpyderDocs/plot_example.py
temp.py
plot_example.py*
colors.py
/Users/juanis/opt/anaconda3/lib/python3.7/site-packages/matplotlib/colors.py
1,280 × 720
LSP Python: ready
conda: base (Python 3.7.4)
Line 106, Col 1
ASCII
LF

```python
"""
Plot a terrain model and a polar plot side by side.
"""

# Third party imports
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.cm
import matplotlib.colors
import mpl_toolkits.mplot3d # pylint: disable=unused-import


plt.style.use("dark_background")

def generate_polar_plot():
    """Generate an example polar slice plot."""
    # Compute pie slices
    n_slices = 20
    theta = np.linspace(0.0, 2 * np.pi, n_slices, endpoint=False)
    radii = 10 * np.random.rand(n_slices)
    width = np.pi / 4 * np.random.rand(n_slices)

    fig, ax= plt.subplots(figsize=(8, 3))
    fig.patch.set_facecolor('#395979')
    ax1 = plt.subplot(1, 2, 2, projection='polar')
    ax1.set_facecolor('#395979')
    bars = ax1.bar(theta, radii, width=width, bottom=0.0)

    # Use custom colors and opacity
    for radius, plot_bar in zip(radii, bars):
        plot_bar.set_facecolor(plt.cm.viridis(radius / 10.))
        plot_bar.set_alpha(0.5)


def generate_dem_plot():
    """Generate a 3D reprisentation of a terrain DEM."""
```

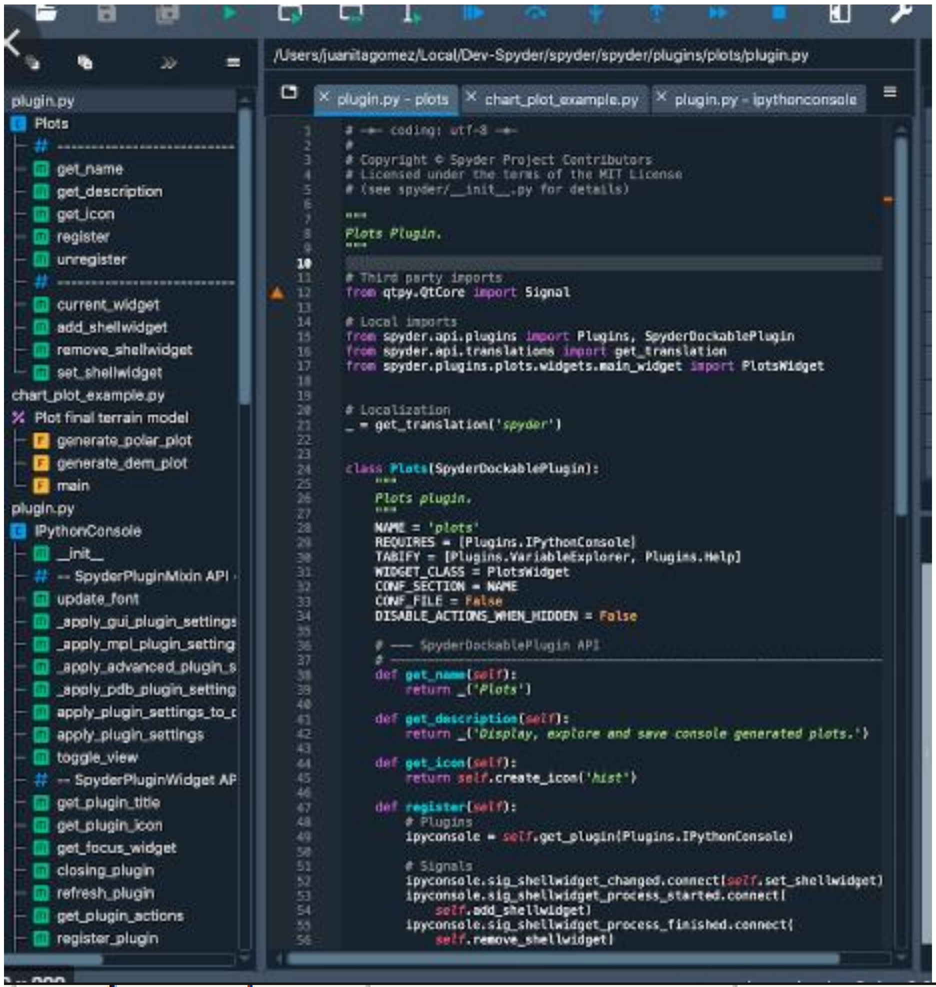
```python


_colors_full_map = {}
# Set by reverse priority order.
_colors_full_map.update(XKCD_COLORS)
_colors_full_map.update({k.replace('grey', 'gray'): v
                         for k, v in XKCD_COLORS.items()
                         if 'grey' in k})
_colors_full_map.update(CSS4_COLORS)
_colors_full_map.update(TABLEAU_COLORS)
_colors_full_map.update({k.replace('gray', 'grey'): v
                         for k, v in TABLEAU_COLORS.items()
                         if 'gray' in k})
_colors_full_map.update(BASE_COLORS)
_colors_full_map = _ColorMapping(_colors_full_map)


def get_named_colors_mapping():
    """Return the global mapping of names to named colors."""
    return _colors_full_map


def _sanitize_extrema(ex):
    if ex is None:
        return ex
    try:
        ret = ex.item()
    except AttributeError:
        ret = float(ex)
    return ret


def _is_nth_color(c):
    """Return whether *c* can be interpreted as an item in the c
    return isinstance(c, str) and re.match(r"\AC[0-9]+\Z", c)


def is_color_like(c):
    """Return whether *c* can be interpreted as an RGB(A) color.
    # Special-case nth color syntax because it cannot be parsed
    if _is_nth_color(c):
        return True
    try:
        to_rgba(c)
    except ValueError:
        return False
```



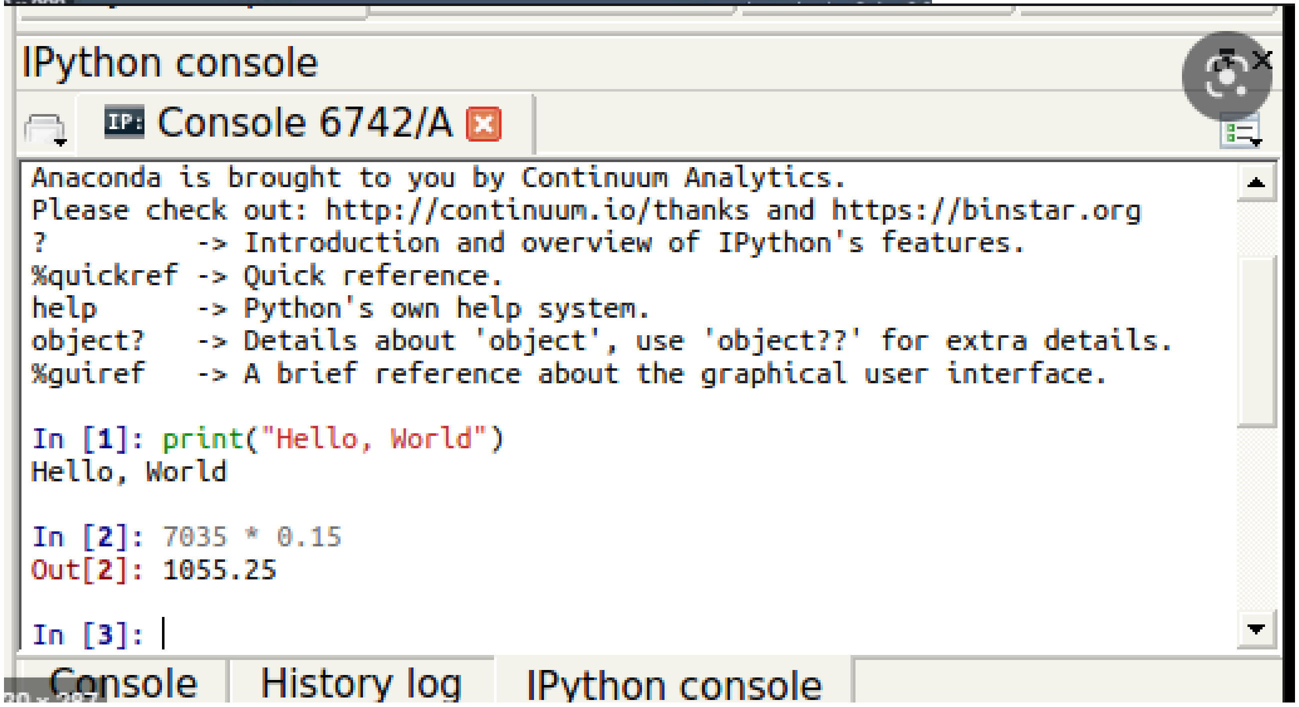
```python
# -*- coding: utf-8 -*-
"""Test outline."""
# Third party imports
from matplotlib.pyplot import figure
from numpy import linspace
from numpy.core.umath import log2


def nextpow2(val):
    r"""."""
    val_abs = abs(val)

    exponent = 1

    while exponent < val_abs:
        exponent *= 2

    exponent = log2(exponent)

    return exponent


if __name__ == '__main__':
    data = linspace(1, 2, 3)

    fig = figure()
    ax1 = fig.add_subplot(1, 1, 1)
    ax1.plot(data)

    data = linspace(1, 3, 3)
    fig = figure()
    ax1 = fig.add_subplot(1, 1, 1)
    ax1.plot(data)
```



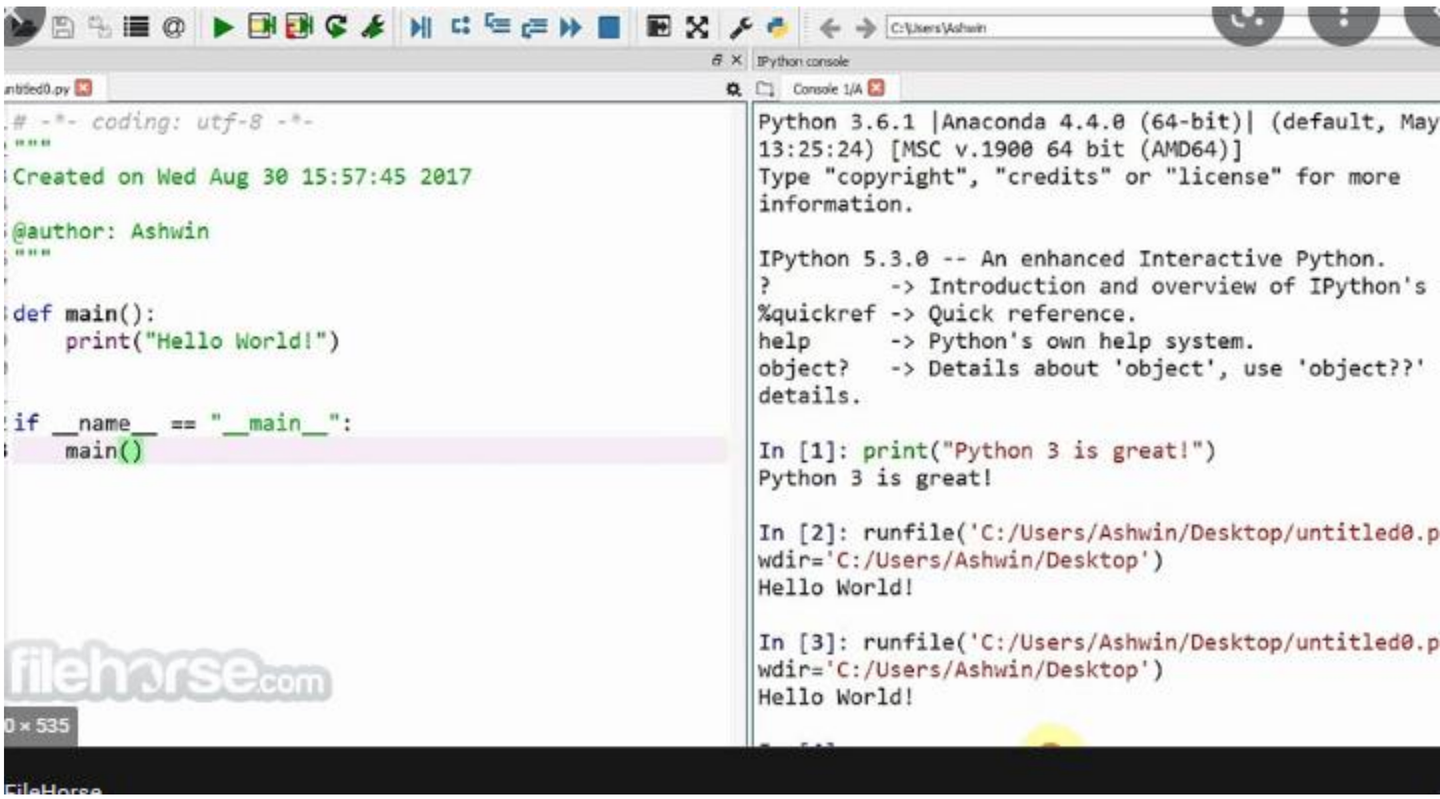
```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Mon May  4 17:07:07 2020

@author: juanis
"""

# pylint: disable=invalid-name

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from scipy.spatial import KDTree, Voronoi, voronoi_plot_2d
from shapely.geometry import Point, Polygon
plt.style.use('dark_background')


file_paths = ["airports_CO.dat",
              "borders_CO.dat"]

airports_CO = file_paths[0]
borders_CO=file_paths[1]


# Read files

airports_Col = pd.read_csv(airports_CO, sep=r"\s+", names=["
                           'Altitude', "City", "Department",
borders_Col = pd.read_csv(borders_CO, sep=r"\s+", names=["co


class FlightOperations:

    def __init__(self, airports, borders):
        self.airports = airports
        self.borders = borders
        self.points = self.airports[['coord-x', 'coord-y']].
        #self.hull = ConvexHull(self.points)
        self.vor = Voronoi(self.points, )

    def plotAirports(self):
        """ Plot map with airports """
        voronoi_plot_2d(self.vor)
```



```python
# -*- coding: utf-8 -*-
#
# Copyright © Spyder Project Contributors
# Licensed under the terms of the MIT License
# (see spyder/__init__.py for details)

"""
Plots Plugin.
"""

# Third party imports
from qtpy.QtCore import Signal

# Local imports
from spyder.api.plugins import Plugins, SpyderDockablePlugin
from spyder.api.translations import get_translation
from spyder.plugins.plots.widgets.main_widget import PlotsWidget


# Localization
_ = get_translation('spyder')


class Plots(SpyderDockablePlugin):
    """
    Plots plugin.
    """
    NAME = 'plots'
    REQUIRES = [Plugins.IPythonConsole]
    TABIFY = [Plugins.VariableExplorer, Plugins.Help]
    WIDGET_CLASS = PlotsWidget
    CONF_SECTION = NAME
    CONF_FILE = False
    DISABLE_ACTIONS_WHEN_HIDDEN = False

    # --- SpyderDockablePlugin API
    # ------------------------------------------------------------------
    def get_name(self):
        return _('Plots')

    def get_description(self):
        return _('Display, explore and save console generated plots.')

    def get_icon(self):
        return self.create_icon('hist')

    def register(self):
        # Plugins
        ipyconsole = self.get_plugin(Plugins.IPythonConsole)

        # Signals
        ipyconsole.sig_shellwidget_changed.connect(self.set_shellwidget)
        ipyconsole.sig_shellwidget_process_started.connect(
            self.add_shellwidget)
        ipyconsole.sig_shellwidget_process_finished.connect(
            self.remove_shellwidget)
```



```
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://binstar.org
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.
%guiref   -> A brief reference about the graphical user interface.

In [1]: print("Hello, World")
Hello, World

In [2]: 7035 * 0.15
Out[2]: 1055.25

In [3]: |
```

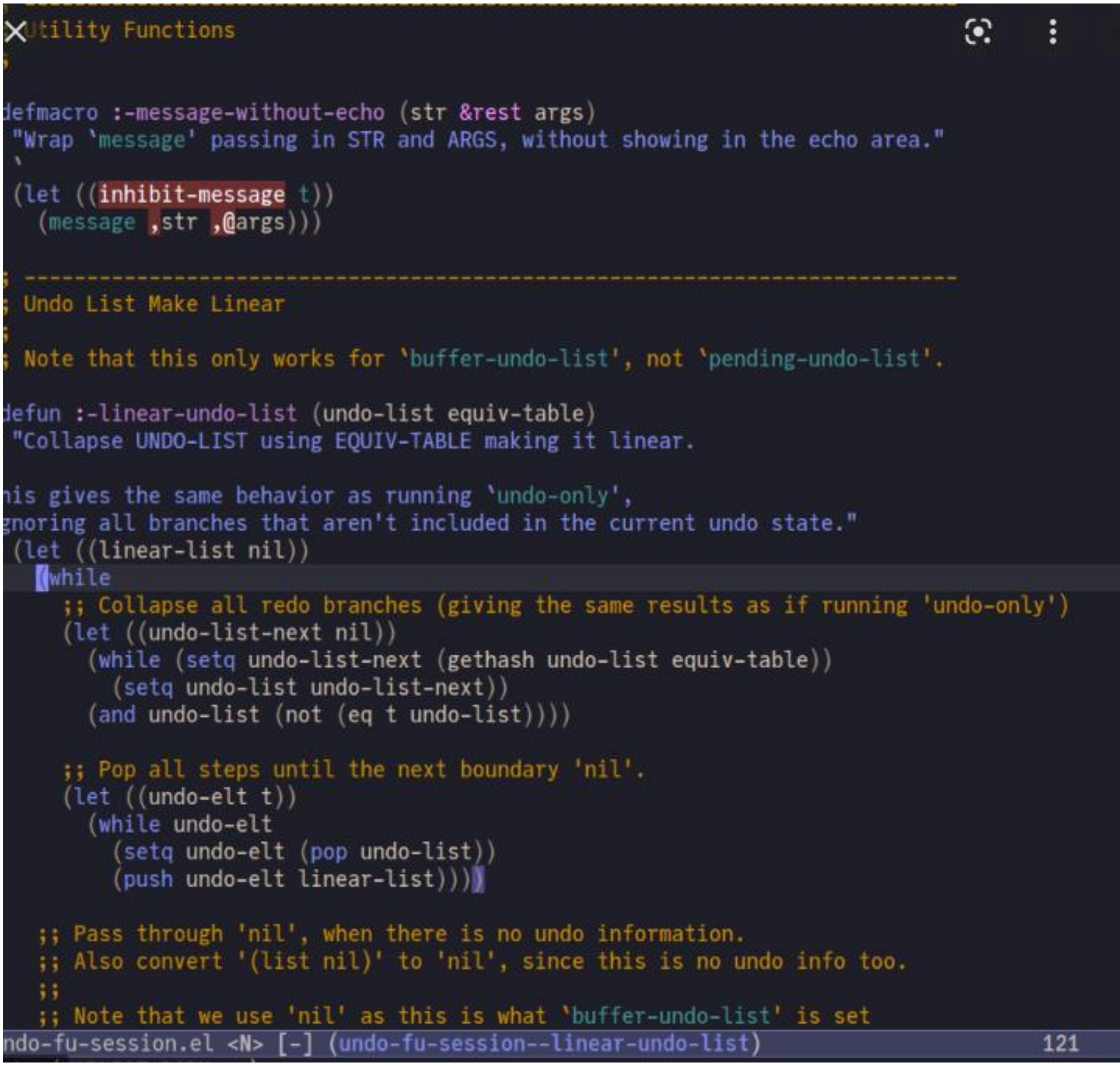


```
# -*- coding: utf-8 -*-
"""
Created on Wed Aug 30 15:57:45 2017

@author: Ashwin
"""

def main():
    print("Hello World!")


if __name__ == "__main__":
    main()
```

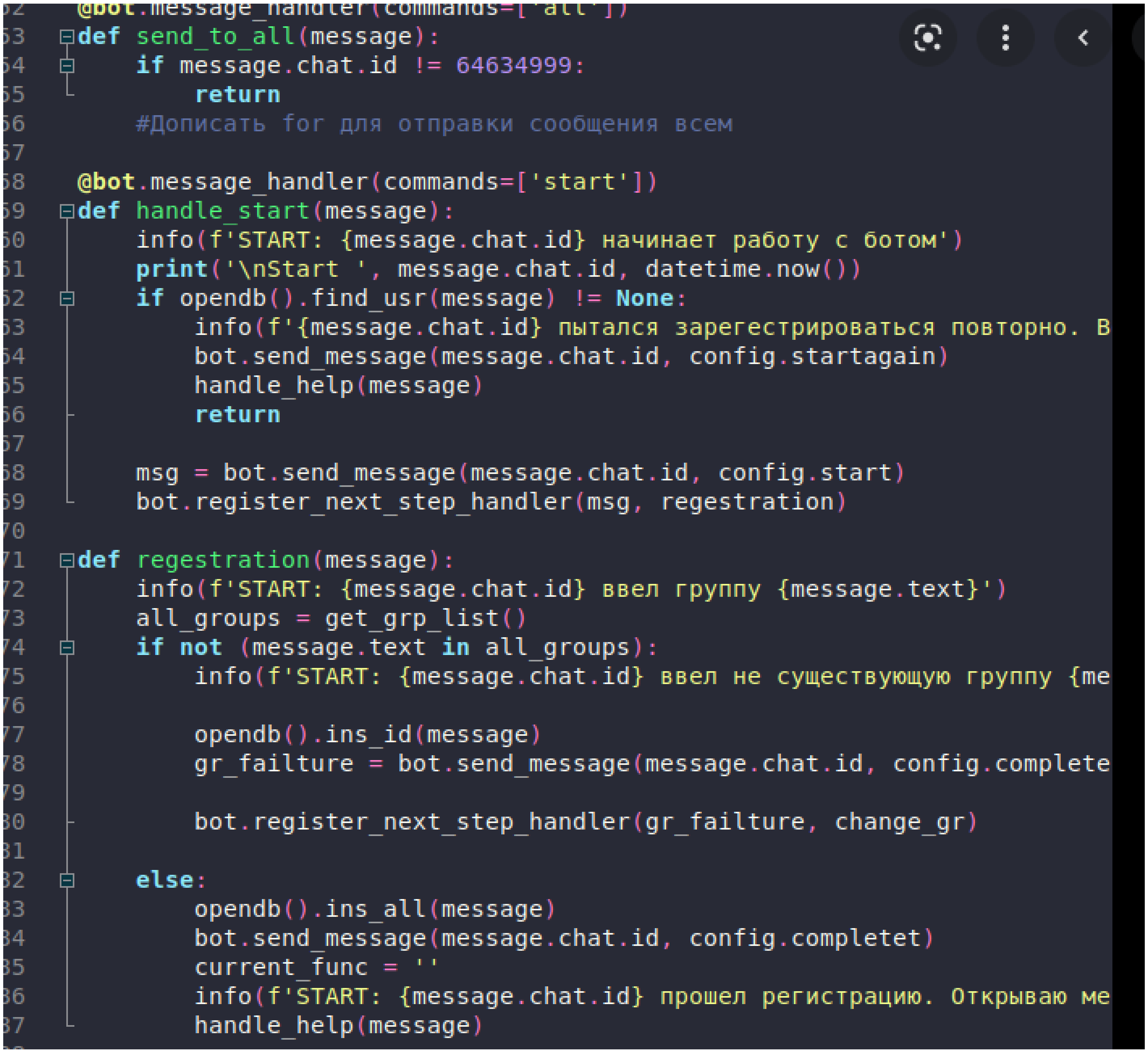
```
Python 3.6.1 |Anaconda 4.4.0 (64-bit)| (default, May
13:25:24) [MSC v.1900 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more
information.

IPython 5.3.0 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object', use 'object??'
details.

In [1]: print("Python 3 is great!")
Python 3 is great!

In [2]: runfile('C:/Users/Ashwin/Desktop/untitled0.p
wdir='C:/Users/Ashwin/Desktop')
Hello World!

In [3]: runfile('C:/Users/Ashwin/Desktop/untitled0.p
wdir='C:/Users/Ashwin/Desktop')
Hello World!
```

0 × 535

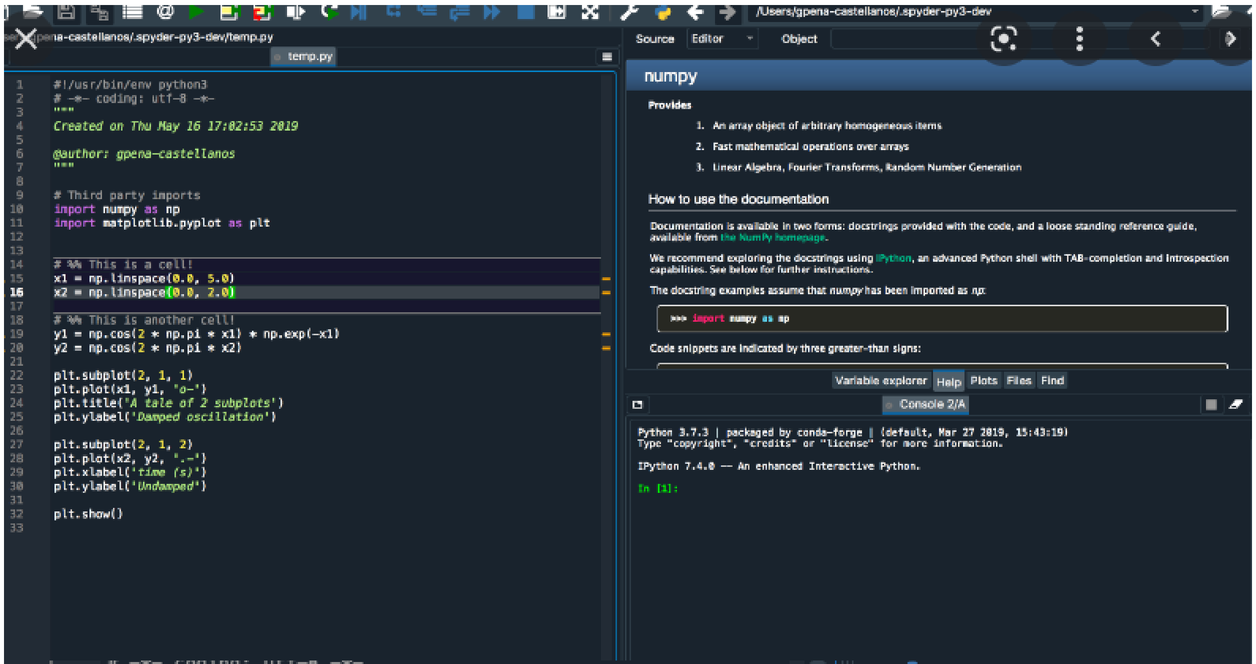
FileHorse



```
tility Functions
;

defmacro :-message-without-echo (str &rest args)
 "Wrap `message' passing in STR and ARGS, without showing in the echo area."
 `
 (let ((inhibit-message t))
   (message ,str ,@args)))

; -------------------------------------------------------------------------
; Undo List Make Linear
;
; Note that this only works for `buffer-undo-list', not `pending-undo-list'.

defun :-linear-undo-list (undo-list equiv-table)
 "Collapse UNDO-LIST using EQUIV-TABLE making it linear.

his gives the same behavior as running `undo-only',
gnoring all branches that aren't included in the current undo state."
 (let ((linear-list nil))
   (while
     ;; Collapse all redo branches (giving the same results as if running 'undo-only')
     (let ((undo-list-next nil))
       (while (setq undo-list-next (gethash undo-list equiv-table))
         (setq undo-list undo-list-next))
       (and undo-list (not (eq t undo-list))))

     ;; Pop all steps until the next boundary 'nil'.
     (let ((undo-elt t))
       (while undo-elt
         (setq undo-elt (pop undo-list))
         (push undo-elt linear-list))))

   ;; Pass through 'nil', when there is no undo information.
   ;; Also convert '(list nil)' to 'nil', since this is no undo info too.
   ;;
   ;; Note that we use 'nil' as this is what `buffer-undo-list' is set
ndo-fu-session.el <N> [-] (undo-fu-session--linear-undo-list)                121
```



```python
@bot.message_handler(commands=['all'])
def send_to_all(message):
    if message.chat.id != 64634999:
        return
    #Дописать for для отправки сообщения всем

@bot.message_handler(commands=['start'])
def handle_start(message):
    info(f'START: {message.chat.id} начинает работу с ботом')
    print('\nStart ', message.chat.id, datetime.now())
    if opendb().find_usr(message) != None:
        info(f'{message.chat.id} пытался зарегестрироваться повторно. В
        bot.send_message(message.chat.id, config.startagain)
        handle_help(message)
        return

    msg = bot.send_message(message.chat.id, config.start)
    bot.register_next_step_handler(msg, regestration)

def regestration(message):
    info(f'START: {message.chat.id} ввел группу {message.text}')
    all_groups = get_grp_list()
    if not (message.text in all_groups):
        info(f'START: {message.chat.id} ввел не существующую группу {me

        opendb().ins_id(message)
        gr_failture = bot.send_message(message.chat.id, config.complete

        bot.register_next_step_handler(gr_failture, change_gr)

    else:
        opendb().ins_all(message)
        bot.send_message(message.chat.id, config.completet)
        current_func = ''
        info(f'START: {message.chat.id} прошел регистрацию. Открываю ме
        handle_help(message)
```


/Users/gpena-castellanos/.spyder-py3-dev
na-castellanos/.spyder-py3-dev/temp.py
temp.py
Source
Editor
Object
Variable explorer
Help
Plots
Files
Find
Console 2/A

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Thu May 16 17:02:53 2019

@author: gpena-castellanos
"""

# Third party imports
import numpy as np
import matplotlib.pyplot as plt


# %% This is a cell!
x1 = np.linspace(0.0, 5.0)
x2 = np.linspace(0.0, 2.0)

# %% This is another cell!
y1 = np.cos(2 * np.pi * x1) * np.exp(-x1)
y2 = np.cos(2 * np.pi * x2)

plt.subplot(2, 1, 1)
plt.plot(x1, y1, 'o-')
plt.title('A tale of 2 subplots')
plt.ylabel('Damped oscillation')

plt.subplot(2, 1, 2)
plt.plot(x2, y2, '.-')
plt.xlabel('time (s)')
plt.ylabel('Undamped')

plt.show()
```

numpy

Provides

1. An array object of arbitrary homogeneous items
2. Fast mathematical operations over arrays
3. Linear Algebra, Fourier Transforms, Random Number Generation

How to use the documentation

Documentation is available in two forms: docstrings provided with the code, and a loose standing reference guide, available from the NumPy homepage.

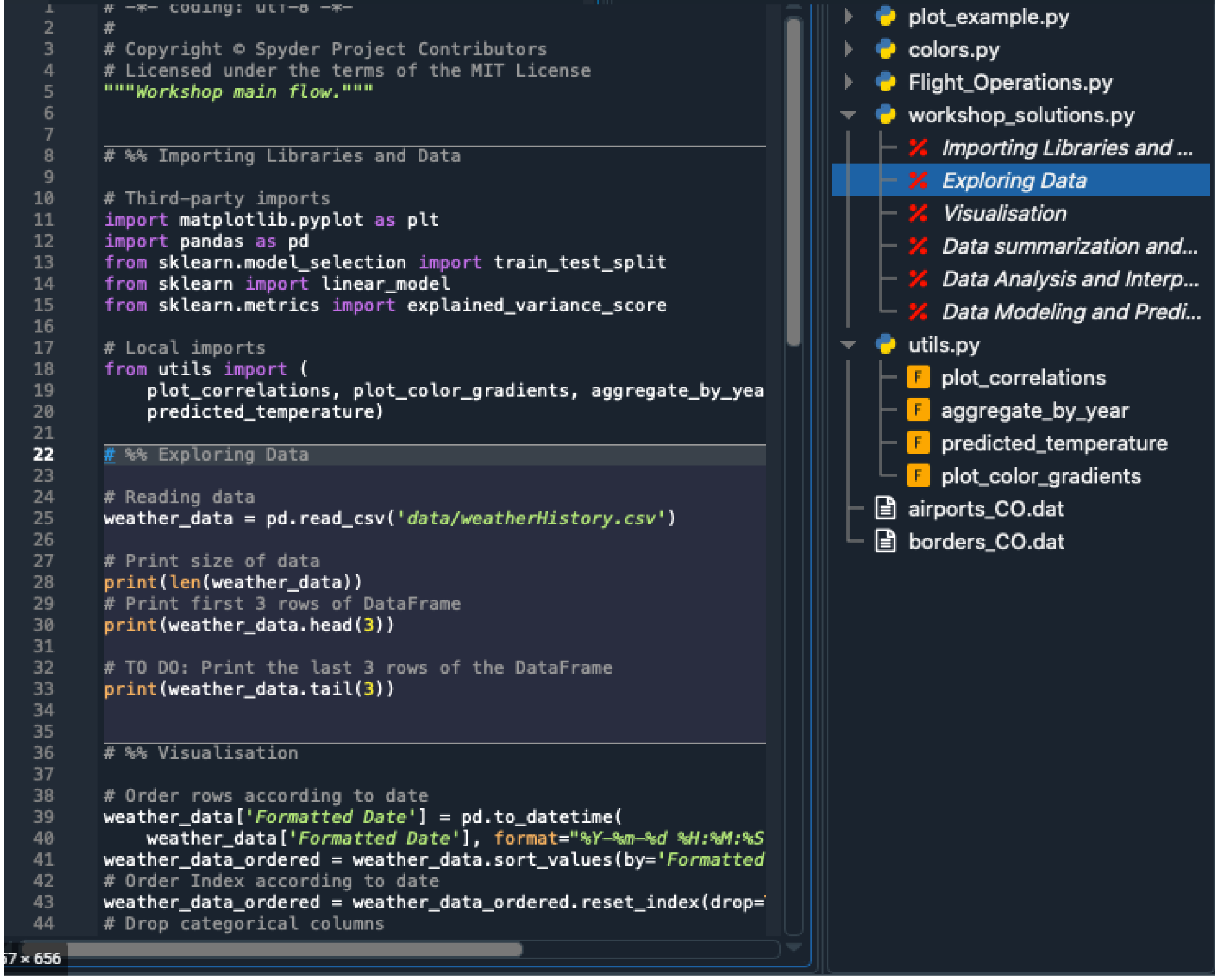
We recommend exploring the docstrings using IPython, an advanced Python shell with TAB-completion and introspection capabilities. See below for further instructions.

The docstring examples assume that *numpy* has been imported as *np*:

```
>>> import numpy as np
```

Code snippets are indicated by three greater-than signs:

```
Python 3.7.3 | packaged by conda-forge | (default, Mar 27 2019, 15:43:19)
Type "copyright", "credits" or "license" for more information.

IPython 7.4.0 -- An enhanced Interactive Python.

In [1]:
```



```python
# -*- coding: utf-8 -*-
#
# Copyright © Spyder Project Contributors
# Licensed under the terms of the MIT License
"""Workshop main flow."""


# %% Importing Libraries and Data

# Third-party imports
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn import linear_model
from sklearn.metrics import explained_variance_score

# Local imports
from utils import (
    plot_correlations, plot_color_gradients, aggregate_by_yea
    predicted_temperature)

# %% Exploring Data

# Reading data
weather_data = pd.read_csv('data/weatherHistory.csv')

# Print size of data
print(len(weather_data))
# Print first 3 rows of DataFrame
print(weather_data.head(3))

# TO DO: Print the last 3 rows of the DataFrame
print(weather_data.tail(3))


# %% Visualisation

# Order rows according to date
weather_data['Formatted Date'] = pd.to_datetime(
    weather_data['Formatted Date'], format="%Y-%m-%d %H:%M:%S
weather_data_ordered = weather_data.sort_values(by='Formatted
# Order Index according to date
weather_data_ordered = weather_data_ordered.reset_index(drop=
# Drop categorical columns
```

3.webpage creation using python

```
blog
  -templates
     -blog
        -index.html
```

Add the following code in **index.html**.

```
<!DOCTYPE html>

<html lang="en">

<head>
    <meta charset="utf-8" />
    <link rel="stylesheet" href="css/style.css">
    <link href="images/favicon.ico" rel="shortcut icon">
    <title>First Blog</title>
</head>

<body>

<div class="container">
    <h1>First Blog</h1>
    <h2>Title</h2>
    <h3>Posted on date by author</h3>
    <p>Body Text</p>

</div>

</body>

</html>
```

Now, we'll create our blog URLs. Create the file **urls.py** in the blog directory and write the URL path for serving the index page.

```
from django.urls import path

from . import views

urlpatterns = [


    path('', views.home),


]
```