a) Modify **b_soup_1.py** so that the program's only output is the final sequence of table cell value **list**s: no **bsyc_temp.txt** file, no intermediate results being displayed, etc.

Modify the code at the end of the program so that the table cell values are accumulated into a **list** of **list**s, representing the table of rows, something like this:

```
daily_yield_curves = [
        [ … header list … ],
        [ … first data list … ],
        …
        [ … final data list … ]
    ]
```

The first "inner" **list** should represent the header row:

```
['Date', '1 mo', '2 mo', '3 mo', '6 mo', '1 yr', '2 yr',
 '3 yr', '5 yr', '7 yr', '10 yr', '20 yr', '30 yr']
```

Following that should be a **list** for each data row. Be sure to convert each interest rate value from a string to a **float**:

```
['01/02/19', 2.40, 2.40, 2.42, 2.51, 2.60, 2.50,
             2.47, 2.49, 2.56, 2.66, 2.83, 2.97]
...
['09/13/19', 1.99, 1.98, 1.96, 1.92, 1.88, 1.79,
             1.76, 1.75, 1.83, 1.90, 2.17, 2.37]
...
```

Modify **b_soup_1.py** again to create a file named **daily_yield_curves.txt** containing a neatly formatted table of this information for the year **2020**.

a. Investigate **matplotlib**'s 3D Surface Plot and Wireframe Plot (have a look at **https://matplotlib.org/Matplotlib.pdf**, and use Google to search for example code). Produce a 3D Surface Plot (**plot_surface** function) of the daily yield curves, with days since 01/02/20 on the X axis, months to maturity on the Y axis (from 1 month to 360 months), and rate on the Z axis. Orient the plot in such a way that this yield curve evolution surface is reasonable to look at. Set axis labels like **'trading days since 01/02/20'**, **'months to maturity'**, and **'rate'** so that the user can tell which axis represents which dimension in the plot. After you have produced a Surface Plot, produce a Wireframe Plot (**plot_wireframe** function) of the same information. (You do *not* need to save screenshots of your plots.)

The Y axis should show *months to maturity*. You will have to "convert" the column labels into the appropriate integer number of months. You can be unclever about this and use a **list** like **[1, 2, 3, 6, 12, 24, 36, 60, 84, 120, 240, 360]**, or you can be more clever and set up a **dict** mapping from column name to number of months, like **cn_to_nm = { '1 mo' : 1, '2 mo' : 2, …, '30 yr' : 360 }**. *It is okay to be unclever.*

Hint: You will need to create an **ndarray** of the interest rate values from the **daily_yield_curves list** of **list**s in order to produce plots.

matplotlib facilities for creating 3D Surface Plots and Wireframe Plots make use of **numpy ndarrays**. Recall that you can convert a **list** of **list**s to a 2-dimensional **ndarray** using **np.array()**. As an example, try:

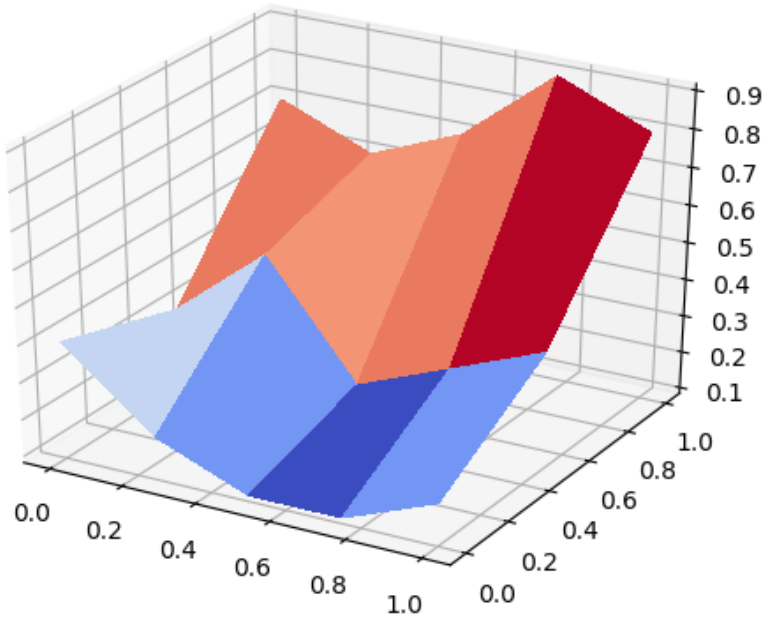
```
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
from matplotlib import cm

X = np.array([ [ 0, .25, .5, .75, 1 ] ]) # 1 by 5

Y = np.array([ [ 0.0 ],                  # 3 by 1
               [ 0.5 ],
               [ 1.0 ] ])

Z = np.array([ [ .4, .2, .1, .1, .2 ],   # 3 by 5
               [ .3, .5, .2, .3, .4 ],
               [ .7, .6, .7, .9, .8 ] ])

fig = plt.figure()
ax = fig.add_subplot(1, 1, 1, projection='3d')
surf = ax.plot_surface(X, Y, Z, cmap=cm.coolwarm,
                       linewidth=0, antialiased=False)
plt.show()
```

As the last step in creating a plot, the statement

```
plt.show()
```

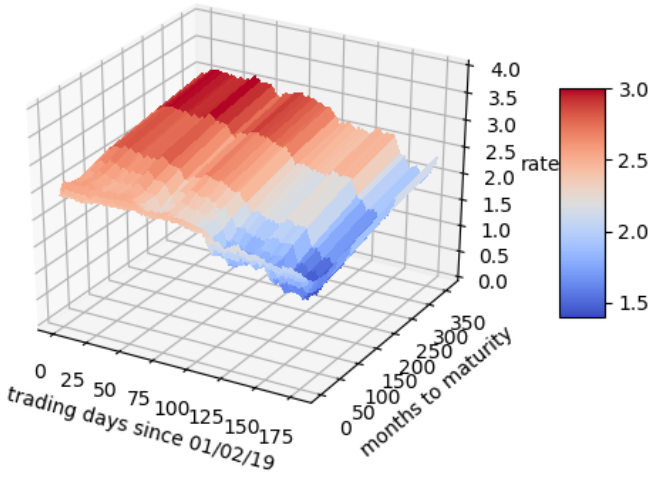
causes the plot to be drawn on your screen. After the plot has been drawn, click the close button, **X**, in the upper right corner so that your program can continue.

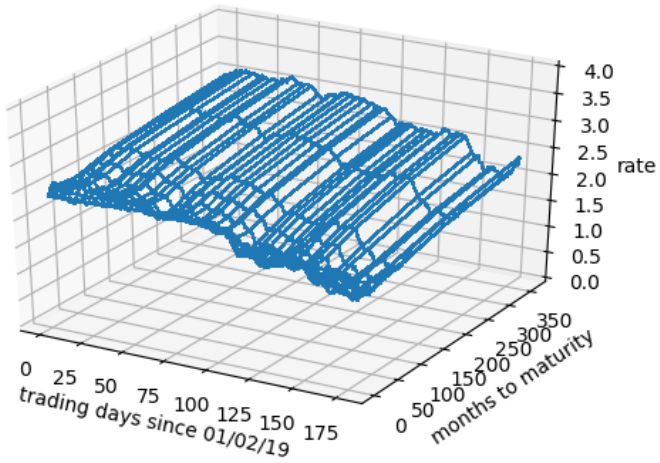
A surface plot of these test **ndarrays**, **X**, **Y**, and **Z**, should look very similar to the screen shot shown here:



Surface and Wireframe Plots of the yield curve data should look somewhat like the two shown below, but with data for the year 2020 instead of for the year 2019. The plots for 2020 will drop dramatically during March, when the Federal Reserve shifted to near 0% short term interest rates due to Covid-19.

Notice that "trading days since 01/02/19" is simply a *count* of the number of trading days. Do not try to include weekends and holidays, since no trading is done on those days and hence no data is available.





b. Our interest rate table is a natural Pandas **DataFrame**, with trading dates as rows and bond maturities as columns. From the **daily_yield_curves list** of **list**s, create a **DataFrame** named **yield_curve_df** with the date strings as the row labels (**'01/02/2020'**, …, **'08/28/2020'**, …), the bond maturities as the column labels (**'1 mo'**, …, **'30 yr'**), and the corresponding interest rate values as the row/column item values. Use appropriate slices/loops/comprehensions involving **daily_yield_curves** to create **yield_curve_df**.

DataFrame has a **plot()** member function that uses **matplotlib**. You can use **yield_curve_df.plot()** to create a plot with rows on the horizontal axis, values on the vertical axis, and with each column represented as a different line. You will still need to use

```
plt.show()
```

to make the plot be drawn on your screen. Since the rows are trading days, this plot will be of the *time series* of interest rates for each maturity: 1 month, 2 months, 3 months, …, 30 years. You will see that during 2020, interest rates for all maturities have fallen. During March, short term rates fell dramatically when the Fed changed policy due to Covid-19.

Generally, it is considered more risky to lend for longer periods of time, so a "normal" yield curve slopes up: interest rates are lowest at 1 month, higher at 1 year, higher still at 10 years, and highest at 30 years. This is what we see for most days during 2020.

If we *transpose* **yield_curve_df**, so that trading dates become the columns and maturities become the rows, then a **plot()** will show us the daily yield curve for every trading day so far this year. This will be an unreadable mess with over 100 lines.

From **yield_curve_df** create a **DataFrame** object named **by_day_yield_curve_df**, containing the transpose of **yield_curve_df** *but* only including a column for every 20^{th} trading day, that is, day 0, day 20, day 40, …, day 240. The column labels should be **'01/02/20'**, **'01/31/20'**, **'03/02/20'**, … if you do this correctly. You will need to modify the row labels from **'1 mo'**, **'2 mo'**, and so forth, to the corresponding integer number of months—1, 2, …, 360—in order for the plot's horizontal axis to make sense.

The by-maturity time series plot and the by-trading-day yield curve plots should look similar to the examples shown here, except for 2020 data rather than 2019 data: