Basic band gap search engine:

Script:

```
from pymatgen.ext.matproj import MPRester

from pymatgen.electronic_structure.plotter import BSDOSPlotter


mpr = MPRester("Insert API key here!")


material_id = input("Enter your material ID: ")


bs = mpr.get_bandstructure_by_material_id(material_id)

dos = mpr.get_dos_by_material_id(material_id)


bsp = BSDOSPlotter()

ax_bs, ax_dos = bsp.get_plot(bs,dos)
```

Explanation:

The goal of this script is to take a material and graph its electronic band structure using its band structure and density of state values. This code was a simple warm up prompt for me to begin to understand how to use the API client. Although unfortunately my Conda environment that I set up in class that worked (excluding crystal tool kit) was somehow lost on my school computer, I spent many days attempting to troubleshoot creating an environment where all of the criteria of the environment.yml were met, unfortunately it seemed as if I was not enough of an expert on the Microsoft interface of Conda to be able to adequately recreate what we had done in class, however in this script and the next I utilized all tools that I could to verify that these scripts would run, I couldn't gather example outputs due to my Conda system not working correctly.

This script first imports the necessary packages needed in this script. It assigns the API key which gives general materials project database access to queries after the mpr variable. The user is asked to enter their material ID. We use the API from materials project database to find the band structure and density of states of that material. And utilize the pymatgen electronic structure plotter to obtain the plot. There is a chance that no plot exists for various materials, however for most that would not be an issue.

Magnetism data:

Script:

```
from pymatgen.ext.matproj import MagnetismRester
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np



mpr = MagnetismRester("API KEY HERE")


material_id = input("Enter your material ID: ")


mag_max = mpr.get_total_magnetization_max_by_material_id(material_id)
mag_min = mpr.get_total_magnetization_min_by_material_id(material_id)


print(f"{mag_max} is the total maximum magnetization of {material_id}")
print(f"{mag_min} is the total minimum magnetization of {material_id}")


material_id_2 = input("Enter your second material ID: ")


mag_max_2 = mpr.get_total_magnetization_max_by_material_id(material_id_2)
mag_min_2 = mpr.get_total_magnetization_min_by_material_id(material_id_2)


print(f"{mag_max_2} is the total maximum magnetization of {material_id_2}")
print(f"{mag_min_2} is the total minimum magnetization of {material_id_2}")


material_id_3 = input("Enter your third material ID: ")
```

```
mag_max_3 = mpr.get_total_magnetization_max_by_material_id(material_id_3)
mag_min_3 = mpr.get_total_magnetization_min_by_material_id(material_id_3)

print(f"{mag_max_3} is the total maximum magnetization of {material_id_3}.")
print(f"{mag_min_3} is the total minimum magnetization of {material_id_3}.")

material_id_4 = input("Enter your fourth material ID: ")

mag_max_4 = mpr.get_total_magnetization_max_by_material_id(material_id_4)
mag_min_4 = mpr.get_total_magnetization_min_by_material_id(material_id_4)

print(f"{mag_max_4} is the total maximum magnetization of {material_id_4}.")
print(f"{mag_min_4} is the total minimum magnetization of {material_id_4}.")

mag_max1234 = [mag_max,mag_max_2,mag_max_3,mag_max_4]
mag_min1234 = [mag_min,mag_min_2,mag_min_3,mag_min_4]
material_id1234 = [material_id,material_id_2,material_id_3,material_id_4]

data = {
   "material id": material_id1234,
   "maximum magnetism (A/m)": mag_max1234,
   "minimum magnetism (A/m)": mag_min1234,
}

df = pd.DataFrame(data)

print(df)
```

```
x = np.arange(len(material_id1234))

width = 0.35


fig, ax = plt.subplots(figsize=(10, 6))


rects1 = ax.bar(x - width/2, mag_max1234, width, label='Max Magnetism', color='skyblue')


rects2 = ax.bar(x + width/2, mag_min1234, width, label='Min Magnetism', color='lightcoral')


ax.set_ylabel('Magnetism (A/m)')

ax.set_xlabel('Material ID')

ax.set_title('Maximum and Minimum Magnetism by Material ID')

ax.set_xticks(x)

ax.set_xticklabels(material_id1234)

ax.legend()


fig.tight_layout()


plt.show()
```

Explanation:

For this script I took on a more complex concept, which is utilizing a subrestor from the materials project API to compare the maximum and minimum magnetism values of various materials, both numerically and visually. First the script imports all the necessary packages. Then using the magnetism rester and the materials project API key, we can inquire about the magnetic properties of the user's input of 4 materials, the program assigns the resulting values to variables and arranges them in lists, these are then turned into a pandas data table for a simpler analysis from the user. The final action is plotting a vertical bar graph, with the y axis representing the magnitude of the magnetism (A/m) and the x axis having 4 regions for the different material ID values present, the graph has 2 bars per region, each with the corresponding value (total maximum magnetism and total minimum magnetism) of the given material.