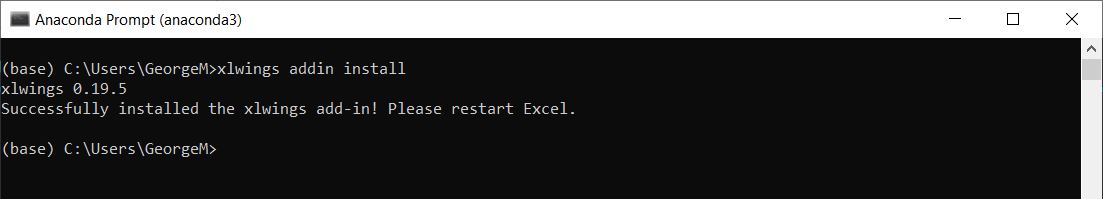
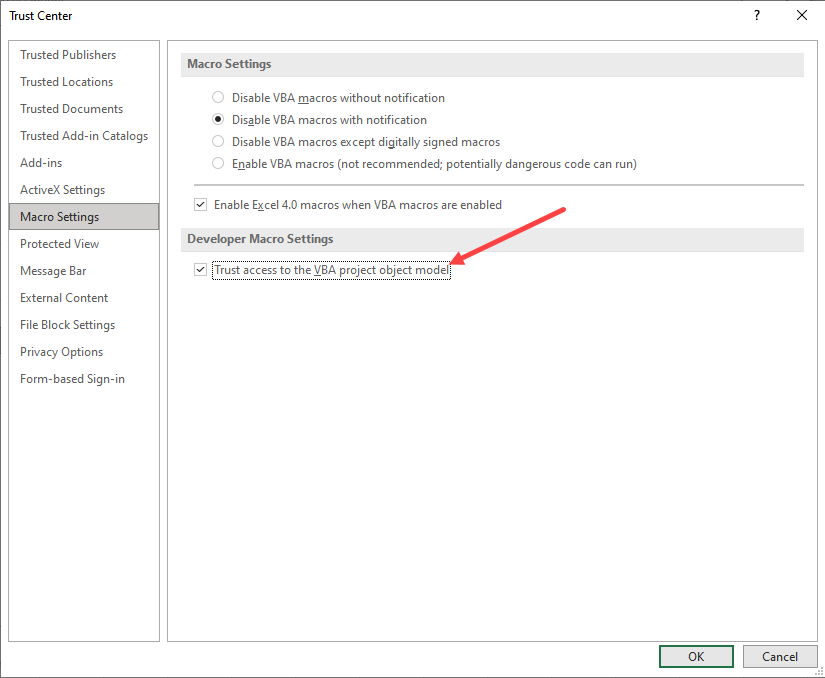
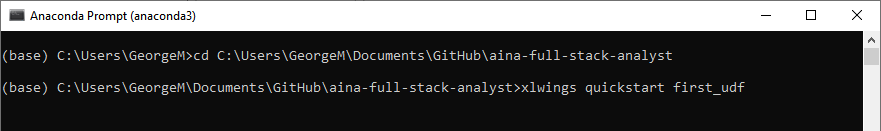
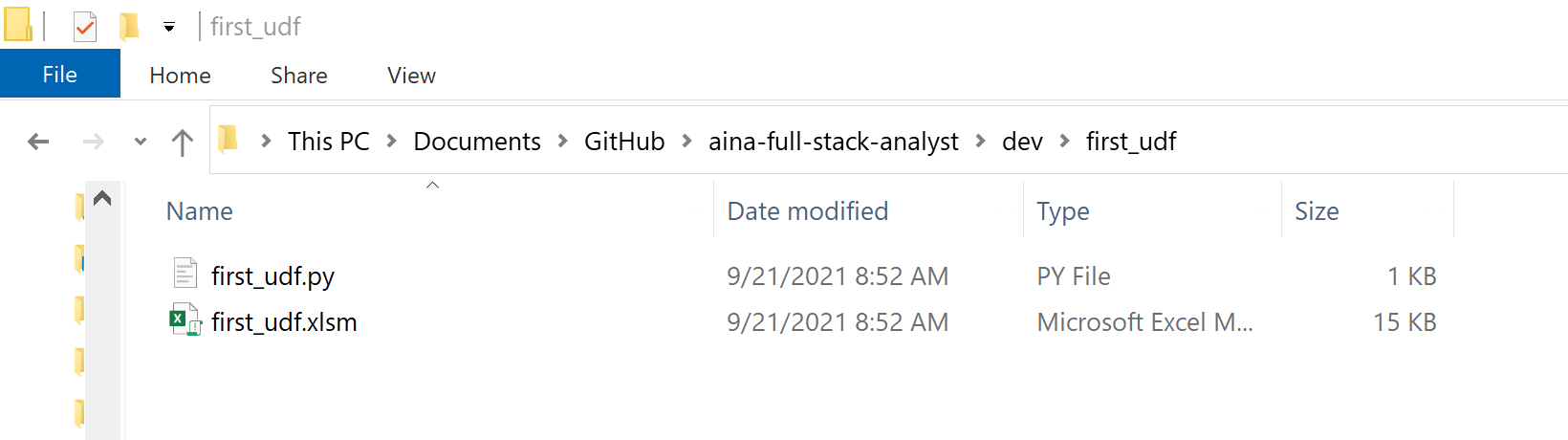
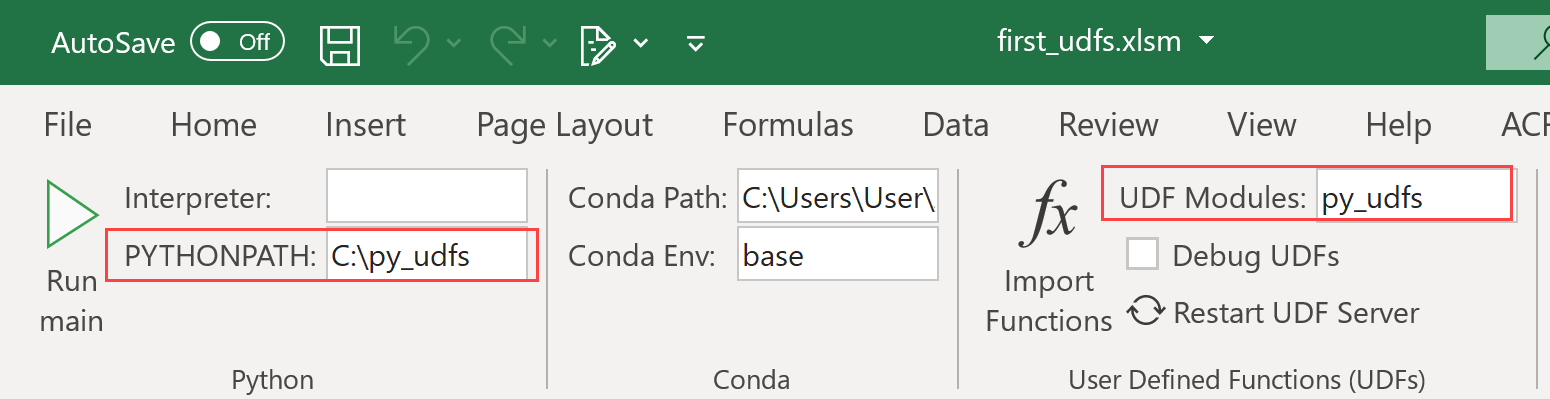
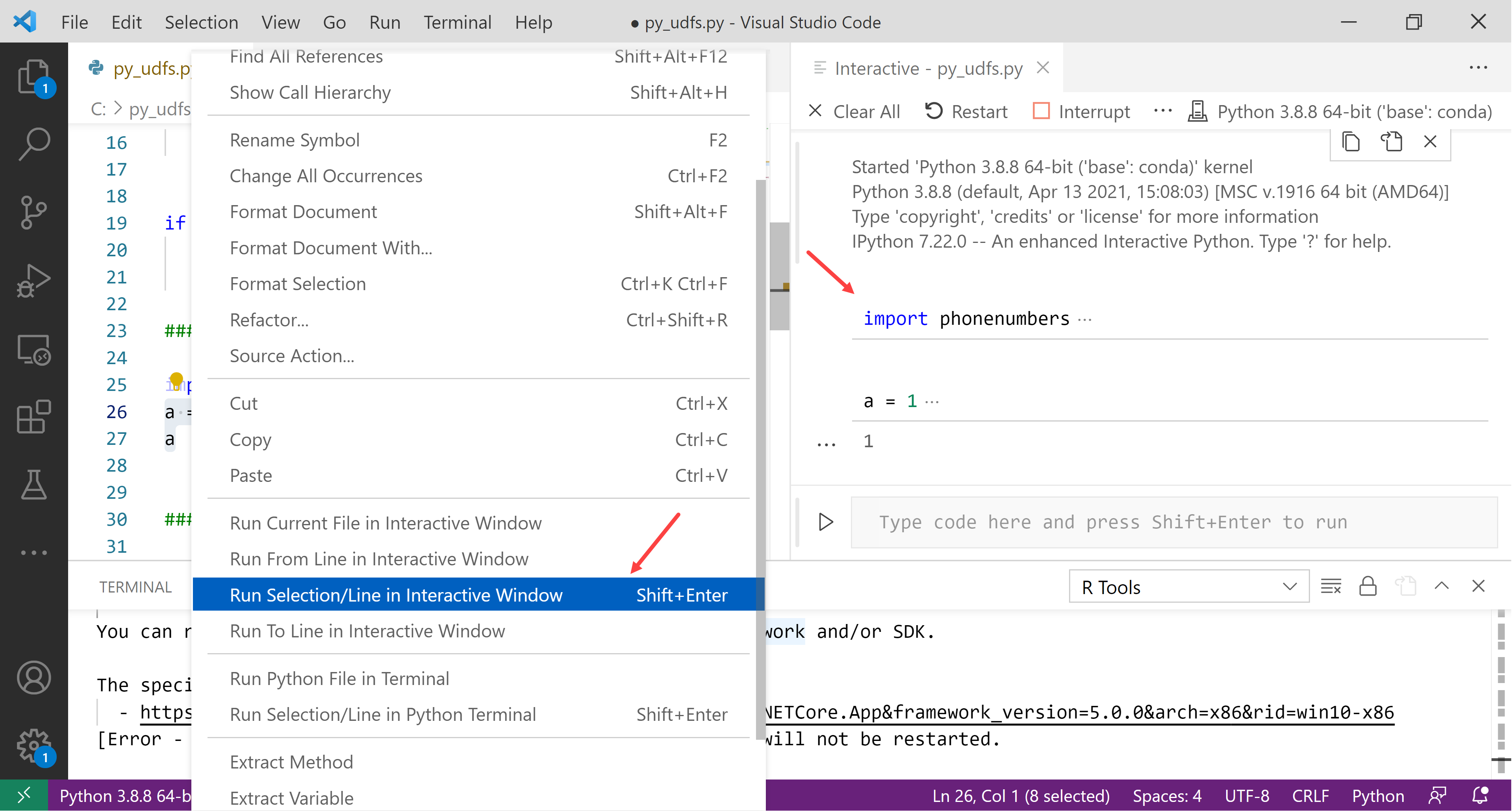
**UDFs with xlwings: demo notes**

1. xlwings comes pre-installed with Anaconda, but you do need to install an Excel add-in to work with UDFs:
   1. Open Anaconda Navigator
   2. Execute xlwings addin install
   3. You should see (and follow!) a message like this:  
      
2. In Excel, go to File > Options > Trust Center > Macro Settings.
3. Click on “Trust access to the VBA project object model”  
   
4. Open Anaconda Navigator
   1. Optional: use the cd command to set the directory where you want
5. Enter xlwings quickstart first\_udf in the cursor  
   
6. This will open a first\_udf subfolder in the selected directory.   
   
7. Try moving your UDF file to somewhere else on your computer like C:\py\_udfs\my\_py\_udfs.py.
   1. Set your PYTHONPATH to C:\py\_udfs and UDF Modules to py\_udfs. Do *not* include the *.py* extension for the UDF Modules name.   
      
8. Go ahead and click Import Functions. If you’re not currently in a *.xlsm* workbook, you will need to create one.
9. Open up your py\_udfs.py file. We will take a look at the hello() function. The idea of a function is to take some input, then do something to it behind the scenes and return something to the user.
   1. You can run the *.py* script in VS Code by right-clicking your selection and running. You may need to download an extension for this.



1. Let’s create a UDF that can look up the timezone of a phone number. The phone number needs to be listed with the country code, such as +1-440-879-8399 (dashes are not necessary)

import phonenumbers

from phonenumbers import timezone

@xw.func

def phone\_timezone(phone\_number, country\_code):

    number\_parse = phonenumbers.parse(phone\_number, country\_code)

    return timezone.time\_zones\_for\_number(number\_parse)

1. For another example, we can create a UDF that detects whether an email address is in a valid format. This will return a True or False to Excel:

import re

valid\_email\_re = r'\b[A-Za-z0-9.\_%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,}\b'

@xw.func

def valid\_email(email):

    if(re.fullmatch(valid\_email\_re, email)):

        return True

    else:

        return False

1. Finally, we can define a function that will take a range and return descriptive statistics of it using pandas. Because we are using an outside data type in this case (the DataFrame) we need an extra argument:

import pandas as pd

@xw.func

@xw.arg("df", pd.DataFrame, index=False, header=True)

def describe(df):

    return df.describe()