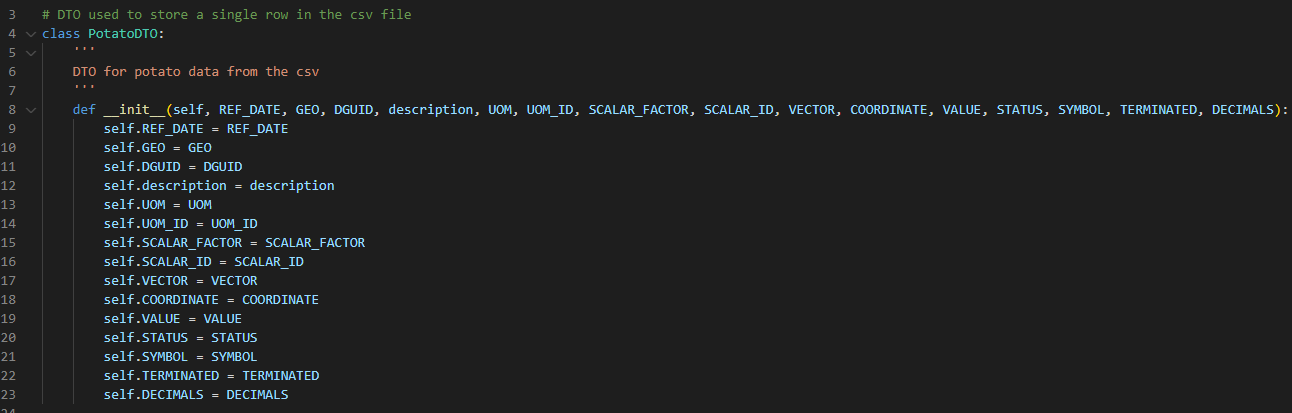
**CST8333 Project By Sebastien Ramsay**

[Rams0130@algonquinlive.com](mailto:Rams0130@algonquinlive.com)

**Evidence of Learning**



This is the import statement for the csv api library.



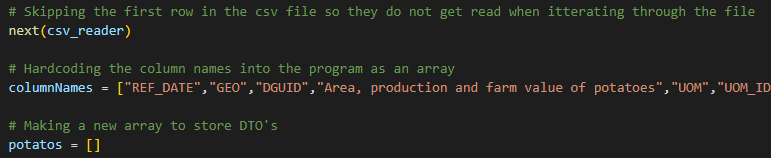
This is the data transfer object used to hold a single row of data.



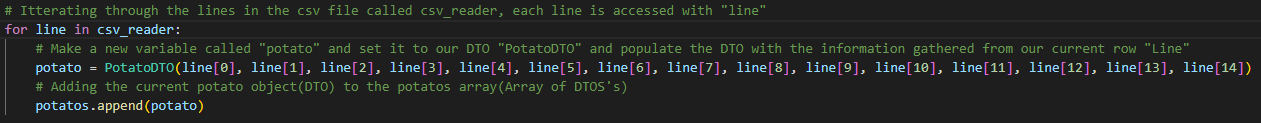
This is where I read the file from our current directory and set it to a var called csv\_file



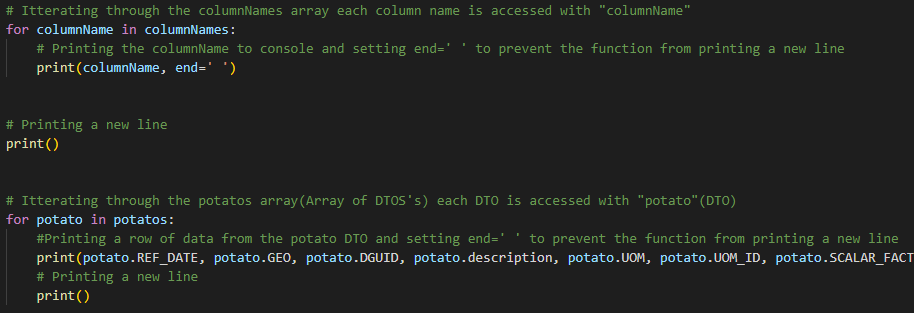
This is where I used the api to convert the file



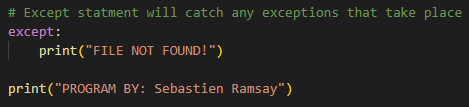
This is skipping the column names at the top of the csv file and hardcoding them in instead. Also making an empty array so I can append DTO’s to it.



This is iterating through the information received from the api and storing the information in DTO’s to append them to the array we just made.

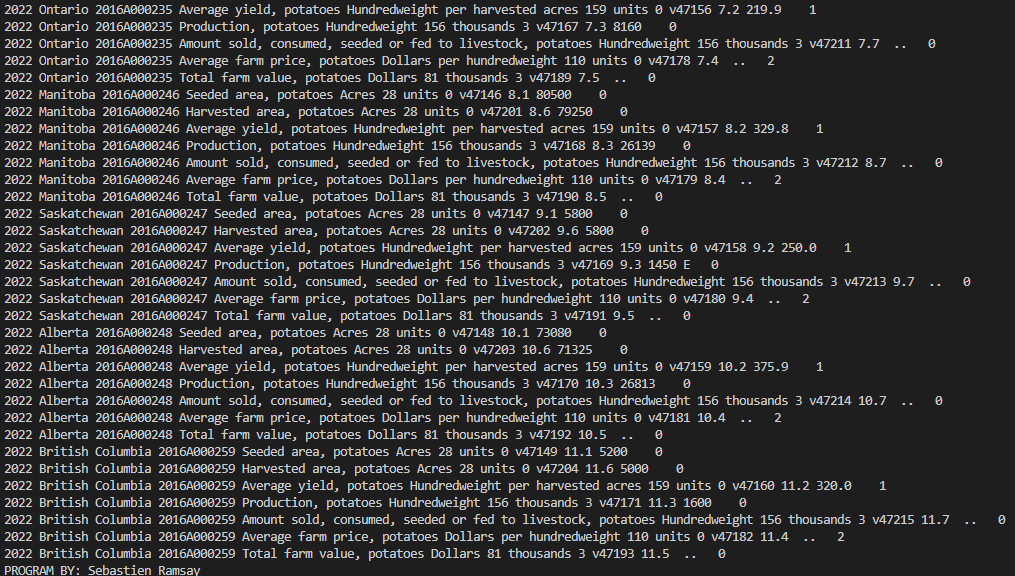


This iterates through the column names and data to print it all out to console.



This catches any exceptions that occur and will reply with file not found.

**Program Demonstration via Screen Shots**



**Source Code Commenting Example**

**import csv**

**# DTO used to store a single row in the csv file**

**class PotatoDTO:**

**'''**

**DTO for potato data from the csv**

**'''**

**def \_\_init\_\_(self, REF\_DATE, GEO, DGUID, description, UOM, UOM\_ID, SCALAR\_FACTOR, SCALAR\_ID, VECTOR, COORDINATE, VALUE, STATUS, SYMBOL, TERMINATED, DECIMALS):**

**self.REF\_DATE = REF\_DATE**

**self.GEO = GEO**

**self.DGUID = DGUID**

**self.description = description**

**self.UOM = UOM**

**self.UOM\_ID = UOM\_ID**

**self.SCALAR\_FACTOR = SCALAR\_FACTOR**

**self.SCALAR\_ID = SCALAR\_ID**

**self.VECTOR = VECTOR**

**self.COORDINATE = COORDINATE**

**self.VALUE = VALUE**

**self.STATUS = STATUS**

**self.SYMBOL = SYMBOL**

**self.TERMINATED = TERMINATED**

**self.DECIMALS = DECIMALS**

**# Try statement for exeption handling**

**try:**

**# Reading the CSV file from the project directory. First arg is file name, second arg indicates to perform a read operation. "as" indicates the name to give this file you are opening**

**with open ('32100358.csv', 'r') as csv\_file:**

**# Make a new variable called csv\_reader that uses the csv import to use the reader function passing through the file we just opened called "csv\_file"**

**csv\_reader = csv.reader(csv\_file)**

**# Skipping the first row in the csv file so they do not get read when itterating through the file**

**next(csv\_reader)**

**# Hardcoding the column names into the program as an array**

**columnNames = ["REF\_DATE","GEO","DGUID","Area, production and farm value of potatoes","UOM","UOM\_ID","SCALAR\_FACTOR","SCALAR\_ID","VECTOR","COORDINATE","VALUE","STATUS","SYMBOL","TERMINATED","DECIMALS"]**

**# Making a new array to store DTO's**

**potatos = []**

**# Itterating through the lines in the csv file called csv\_reader, each line is accessed with "line"**

**for line in csv\_reader:**

**# Make a new variable called "potato" and set it to our DTO "PotatoDTO" and populate the DTO with the information gathered from our current row "Line"**

**potato = PotatoDTO(line[0], line[1], line[2], line[3], line[4], line[5], line[6], line[7], line[8], line[9], line[10], line[11], line[12], line[13], line[14])**

**# Adding the current potato object(DTO) to the potatos array(Array of DTOS's)**

**potatos.append(potato)**

**# Itterating through the columnNames array each column name is accessed with "columnName"**

**for columnName in columnNames:**

**# Printing the columnName to console and setting end=' ' to prevent the function from printing a new line**

**print(columnName, end=' ')**

**# Printing a new line**

**print()**

**# Itterating through the potatos array(Array of DTOS's) each DTO is accessed with "potato"(DTO)**

**for potato in potatos:**

**#Printing a row of data from the potato DTO and setting end=' ' to prevent the function from printing a new line**

**print(potato.REF\_DATE, potato.GEO, potato.DGUID, potato.description, potato.UOM, potato.UOM\_ID, potato.SCALAR\_FACTOR, potato.SCALAR\_ID, potato.VECTOR, potato.COORDINATE, potato.VALUE, potato.STATUS, potato.SYMBOL, potato.TERMINATED, potato.DECIMALS, end=' ')**

**# Printing a new line**

**print()**

**# Except statment will catch any exceptions that take place**

**except:**

**print("FILE NOT FOUND!")**

**print("PROGRAM BY: Sebastien Ramsay")**