Python Assignment - 13

**1. What advantages do Excel spreadsheets have over CSV spreadsheets?**

=> Excel spreadsheets offer several advantages over CSV (Comma-Separated Values) spreadsheets:

**1. Cell Formatting**: Excel allows you to format individual cells or ranges of cells, providing options for font styles, colors, alignment, borders, and more. This allows for enhanced visual presentation and better organization of data compared to CSV, which is a plain text format without formatting capabilities.

**2. Multiple Sheets**: Excel supports multiple sheets within a single workbook, allowing you to organize related data into different sheets or tabs. This feature is not available in CSV, where each file typically represents a single sheet of data.

**3. Formulas and Functions**: Excel provides a wide range of built-in formulas and functions that enable complex calculations and data analysis. These formulas and functions can be applied to cells and ranges, allowing for dynamic and automated data processing. In CSV, you would need to perform calculations and data manipulations externally, using programming languages or other tools.

**4. Charts and Graphs**: Excel offers built-in charting and graphing capabilities, allowing you to create visual representations of data using various chart types. This makes it easier to understand and present data trends or patterns. In CSV, you would need to import the data into a separate tool or write code to generate charts or graphs.

**5. Data Validation and Protection**: Excel provides features for data validation, such as restricting cell values to specific ranges or formats. It also allows you to protect the spreadsheet with passwords to prevent unauthorized changes. These features enhance data integrity and security, which are not available in CSV.

**2.What do you pass to csv.reader() and csv.writer() to create reader and writer objects?**

=> To create reader and writer objects using the csv.reader() and csv.writer() functions in the csv module, you pass a file object as the argument to both functions.

**1. csv.reader**(): To create a reader object, you pass a file object opened in text mode ('r') as the argument to csv.reader().

Example:

import csv

with open(‘data.csv’, ‘’r’) as file:

reader = csv.reader(file)

**2. csv.writer()**: To create a writer object, you pass a file object opened in text mode ('w') as the argument to csv.writer().

Example:

Import csv

With open(‘output.csv’, ‘w’) as file:

Writer = csv.writer(file)

**3. What modes do File objects for reader and writer objects need to be opened in?**

=> For reader and writer objects in the csv module, the corresponding file objects should be opened in different modes:

**1. Reader Objects**: To create a reader object using csv.reader(), the file object should be opened in text mode ('r') with newline handling. It is recommended to use the 'r' mode with the newline='' parameter. This ensures consistent handling of line breaks in the CSV file across different platforms.

**2. Writer Objects**: To create a writer object using csv.writer(), the file object should be opened in text mode ('w') with newline handling. Similar to the reader, it is recommended to use the 'w' mode with newline='' to ensure consistent line breaks in the generated CSV file.

**4. What method takes a list argument and writes it to a CSV file?**

=> The writerow() method is used to write a list of values as a single row in a CSV file using the csv module in Python. To write multiple rows, you can call writerow() method multiple times with different lists.

**5. What do the keyword arguments delimiter and line terminator do?**

=>

**1. delimiter**: The delimiter argument specifies the character or string used to separate fields in the CSV file. By default, the delimiter is a comma (,). You can provide a different character or string as the delimiter value to customize the field separator.

**2. line\_terminator**: The line\_terminator argument specifies the character or string used to terminate each row in the CSV file. By default, the line terminator is the newline character ('\n'). You can provide a different character or string as the line\_terminator value to customize the row terminator.

**6. What function takes a string of JSON data and returns a Python data structure?**

=> The ‘json.loads()’ function is used to parse a string of JSON data and convert it into a Python data structure.

**7. What function takes a Python data structure and returns a string of JSON data?**

=> The ‘json.dumps()’ function is used to serialize a Python data structure into a string of JSON data.