PYTHON ASSIGNMENT 13

1. **What advantages do Excel spreadsheets have over CSV spreadsheets?**
   - Excel is a binary file format that holds information for all workbooks and is use in all worksheets. On the other hand, CSV stands for Comma Separated Values. CSV is in plain text format information with a series of values in a sheet and is separated by Commas. CSV is a text which is full of information and is less compatible with Excel
   - CSV files are formatted and exported from all databases; it is just a source of information; a CSV file can generate from any database, i.e., Oracle, SQL, Server, MySQL, Sybase, informatics, access, excel, database, etc
   - Excel is not only a measure of a store of information but also a tool for a financial analyst to perform a deep analysis of data and numbers. An analyst can build profit and loss statements, do ratio analysis, summarize huge data, and perform functions and formulas in Excel format; it is not just a store of information. On the other hand, we compare Excel with CSV; CSV is only a plain text form of the data set which does not enable the user to perform any deep-dive analysis on the data set and apply functions.
   - Excel is superior to the CSV file format; thus, CSV consumes less file size when the user is importing data; it is a much faster format compared to Excel. CSV does not manipulate data and stores it as-is. Excel also allows the user the add-in feature.

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

   Reading CSV files Using csv.reader()

   To read a CSV file in Python, we can use the csv.reader() function. The csv module defines the following functions:

   csv.reader(csvfile, dialect='excel', **fmtparams)

   Return a reader object which will iterate over lines in the given csvfile. csvfile can be any object which supports the iterator protocol and returns a string each time its __next__() method is called — file objects and list objects are both suitable. If csvfile is a file object, it should be opened with newline=''. 1 An optional dialect parameter can be given which is used to define a set of parameters specific to a particular CSV dialect. It may be an instance of a subclass of the Dialect class or one of the strings returned by the list_dialects() function. The other optional fmtparams keyword arguments can be given to override individual formatting parameters in the current dialect.

   Writing CSV files Using csv.writer()

   csv.writer(csvfile, dialect='excel', **fmtparams)

Return a writer object responsible for converting the user's data into delimited strings on the given file-like object. csvfile can be any object with a write() method. If csvfile is a file object, it should be opened with newline=" 1. An optional dialect parameter can be given which is used to define a set of parameters specific to a particular CSV dialect. It may be an instance of a subclass of the Dialect class or one of the strings returned by the list_dialects() function. The other optional fmtparams keyword arguments can be given to override individual formatting parameters in the current dialect.

3. **What modes do file objects for reader and writer objects need to be opened in?**
   File objects need to be opened in read-binary ('rb') for Reader objects and write-binary ('wb') for Writer objects.

4. **What method takes a list argument and writes it to a CSV file?**
   The writerow() method is used to write a single row at a time into a CSV file. We can write a field row using this method.
   Syntax: writerow(fields)

   The writerows() method is used to write multiple rows at a time i.e., it can be used to write the contents of a 2-dimensional list into a csv file. Row lists can be written using this method.

5. **What do the keyword arguments delimiter and line terminator do?**
   The delimiter is the character that appears between cells on a row. By default, the delimiter for a CSV file is a comma. The line terminator is the character that comes at the end of a row. By default, the line terminator is a newline. You can change characters to different values by using the delimiter and lineterminator keyword arguments with csv.writer().
   Passing delimiter='\t' and lineterminator='\n\n' changes the character between cells to a tab and the character between rows to two newlines. We then call writerow() three times to give us three rows.

6. **What function takes a string of JSON data and returns a Python data structure?**
   If you have a JSON string, you can parse it by using the json.loads() method. The result will be a Python dictionary.

7. **What function takes a Python data structure and returns a string of JSON data?**
   If you have a Python object, you can convert it into a JSON string by using the json.dumps() method.