Unit IV

CHAPTER 13

PYTHON AND CSV FILES

Learning Objectives

After the completion of this chapter, the student will be able to

- Understand what is CSV?
- Able to import CSV files in python programs
- Execute and debug python programs

KH9FZ

13.1 Introduction

Python has a vast library of modules that are included with its distribution. One among the module is the **CSV module** which gives the Python programmer the ability to parse **CSV (Comma Separated Values)** files. **A CSV file is a human readable text file where each line has a number of fields, separated by commas or some other delimiter.** You can assume each line as a row and each field as a column. The CSV module will be able to read and write the vast majority of CSV files.

13.2 Difference between CSV and XLS file formats

The difference between **Comma-Separated** Values (CSV) and e**X**ce**L** **S**heets(XLS) file formats is

Excel	CSV
Excel is a binary file that holds information about all the worksheets in a file, including both content and formatting	CSV format is a plain text format with a series of values separated by commas.
XLS files can only be read by applications that have been especially written to read their format, and can only be written in the same way.	CSV can be opened with any text editor in Windows like notepad, MS Excel, OpenOffice, etc.
Excel is a spreadsheet that saves files into its own proprietary format viz. xls or xlsx	CSV is a format for saving tabular information into a delimited text file with extension .csv
Excel consumes more memory while importing data	Importing CSV files can be much faster, and it also consumes less memory

Files saved in excel cannot be opened or edited by text editors.

13.3 Purpose Of CSV File

CSV is a simple **file format** used to store tabular data, such as a spreadsheet or database. Since they're plain text, they're easier to import into a spreadsheet or another storage database, regardless of the specific software you're using.

You can open CSV files in a spreadsheet program like Microsoft Excel or in a text editor or through a database which make them easier to read.

Note

CSV File cannot store charts or graphs. It stores data but does not contain formatting, formulas, macros, etc.

A CSV file is also known as a Flat File. Files in the CSV format can be imported to and exported from programs that store data in tables, such as ***Microsoft Excel*** or ***OpenOfficeCalc***

13.4 Creating a CSV file using Notepad (or any text editor)

A CSV file is a text file, so it can be created and edited using any text editor, But more frequently a CSV file is created by exporting a spreadsheet or database in the program that created it.

13.4.1 Creating CSV Normal File

To create a CSV file in Notepad, First open a new file using

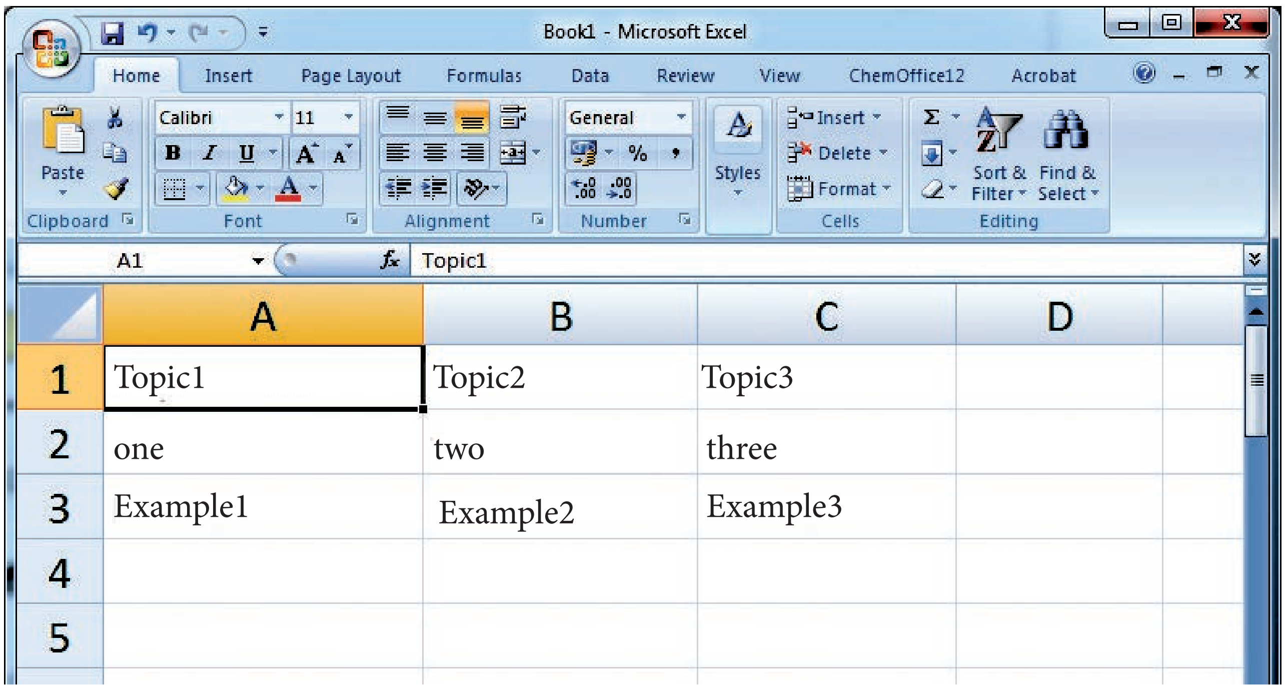
File →New or ctrl +N.

Then enter the data you want the file to contain, separating each value with a comma and each row with a new line.

For example consider the following details

```
Topic1,Topic2,Topic3
one,two,three
Example1,Example2,Example3
```

Save this content in a file with the extension .csv . You can then open the same using Microsoft Excel or any other spreadsheet program. Here we have opened using Microsoft Excel. It would create a table of data similar to the following:



	A	B	C	D
1	Topic1	Topic2	Topic3	
2	one	two	three	
3	Example1	Example2	Example3	
4				
5				

Fig. 13.4.1 CSV file when opened in MS-Excel

In the above CSV file, you can observe the fields of data were separated by commas. But what happens if the data itself contains commas in it?

If the fields of data in your CSV file contain commas, you can protect them by enclosing those data fields in double-quotes ("). ***The commas that are part of your data will then be kept separate from the commas which delimit the fields themselves.***

13.4.2 Creating CSV File That contains Comma With Data

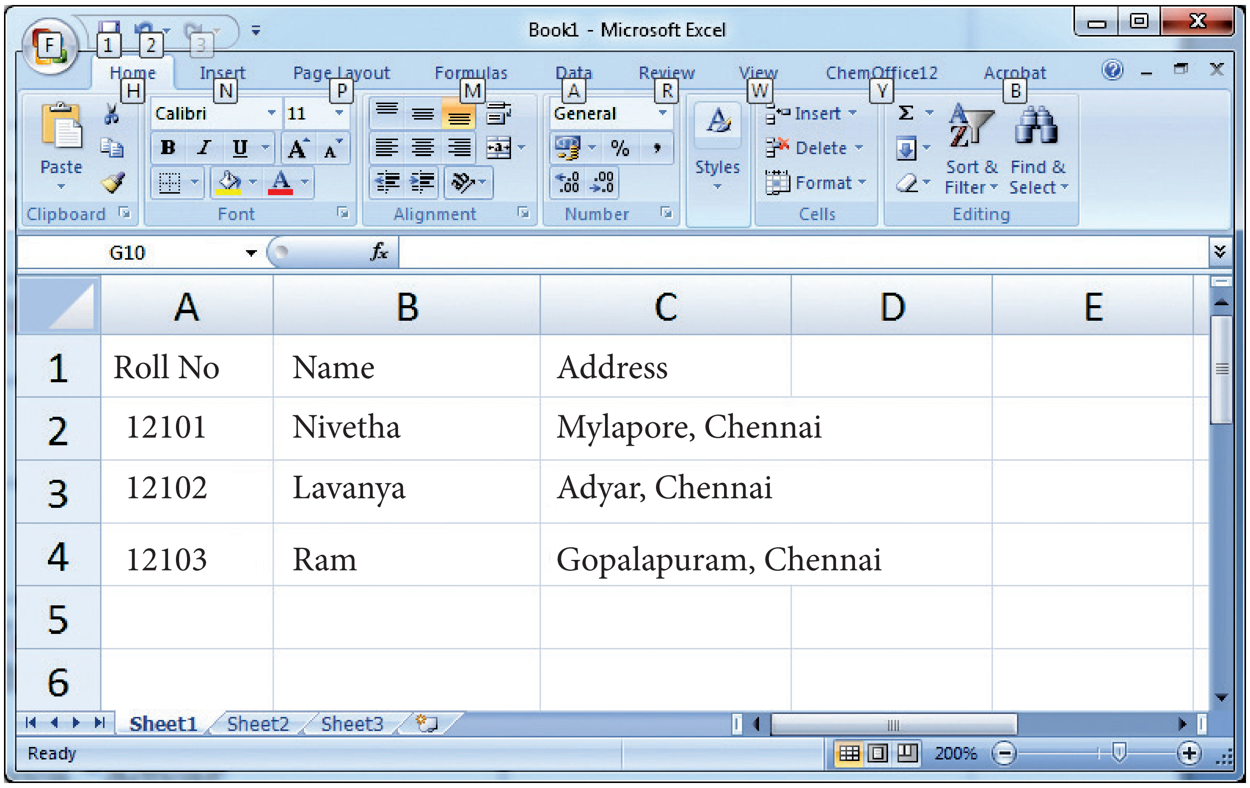
For example, let's say that one of our fields contain commas in the description. If our data looked like the below example:

RollNo	Name	Address
12101	Nivetha	Mylapore, Chennai
12102	Lavanya	Adyar, Chennai
12103	Ram	Gopalapuram, Chennai

To retain the commas in "Address" column, you can enclose the fields in quotation marks. For example:

```
RollNo, Name, Address
12101, Nivetha, "Mylapore, Chennai"
12102, Lavanya, "Adyar, Chennai"
12103, Ram, "Gopalapuram, Chennai"
```

As you can see, only the fields that contain commas are enclosed in quotes. If you open this in MS Excel, It looks like as follows



	A	B	C	D	E
1	Roll No	Name	Address		
2	12101	Nivetha	Mylapore, Chennai		
3	12102	Lavanya	Adyar, Chennai		
4	12103	Ram	Gopalapuram, Chennai		
5					
6					

Fig 13.4.2 (a) CSV Field data with comma in Excel

The same goes for newlines (display data in more than one line example Address column) which may be part of your field data. Any fields containing a newline as part of its data need to be enclosed in double-quotes.

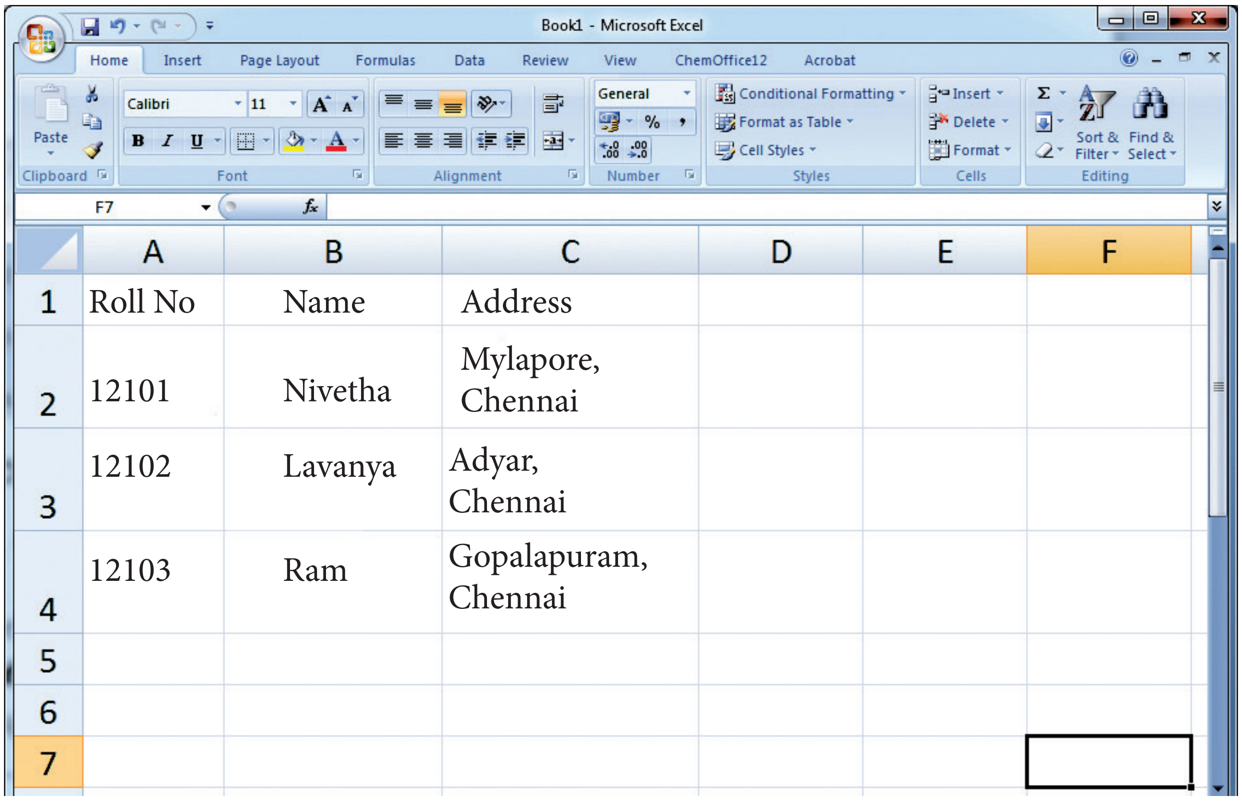
For Example

RollNo	Name	Address
12101	Nivetha	Mylapore, Chennai
12102	Lavanya	Adyar, Chennai
12103	Ram	Gopalapuram, Chennai

It should be written in CSV file as

```
RollNo, Name, Address
12101, Nivetha, “Mylapore
Chennai”
12102, Lavanya, “Adyar
Chennai”
12103, Ram, “Gopalapuram
Chennai”
```

The Result will look like this



	A	B	C	D	E	F
1	Roll No	Name	Address			
2	12101	Nivetha	Mylapore, Chennai			
3	12102	Lavanya	Adyar, Chennai			
4	12103	Ram	Gopalapuram, Chennai			
5						
6						
7						

Fig 13.4.2 (b) CSV Field Data with newline in Excel

13.4.3 Creating CSV File That contains Double Quotes With Data

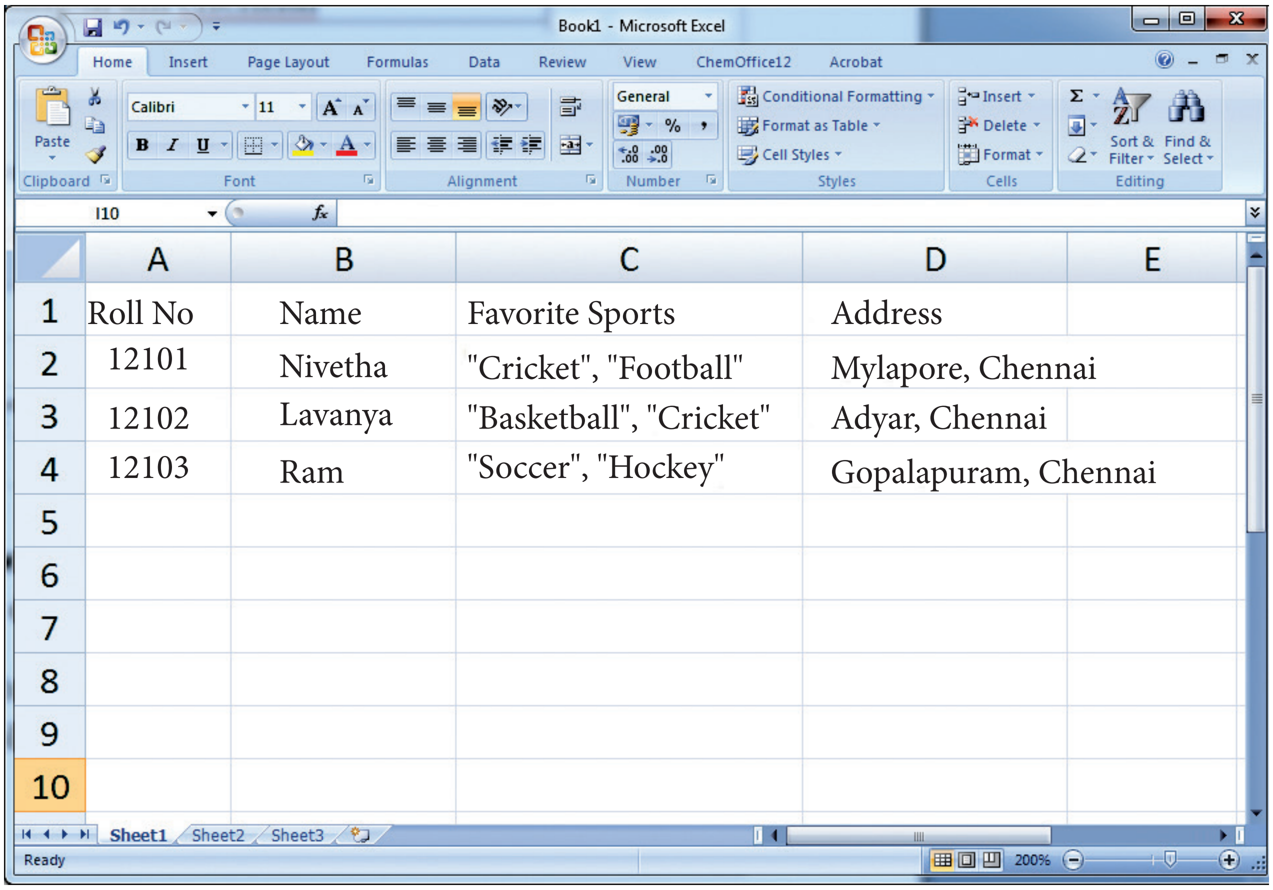
If your fields contain double-quotes as part of their data, **the internal quotation marks need to be doubled so that they can be interpreted correctly.** For Example, given the following data:

Roll No	Name	Favorite Sports	Address
12101	Nivetha	“Cricket”, “Football”	Mylapore Chennai
12102	Lavanya	“Basketball”, “Cricket”	Adyar Chennai
12103	Ram	“Soccer”, “Hockey”	Gopala puram Chennai

It should be written in csv file as

```
RollNo, Name, FavoriteSports, Address
12101, Nivetha,””” Cricket ”””,”” Football ”””, Mylapore chennai
12102, Lavanya,””” Basketball ”””,”” Cricket ”””, Adyar chennai
12103, Ram,””” Soccer”””,”” Hockey”””, Gopalapuram chennai
```

The output will be



	A	B	C	D	E
1	Roll No	Name	Favorite Sports	Address	
2	12101	Nivetha	"Cricket", "Football"	Mylapore, Chennai	
3	12102	Lavanya	"Basketball", "Cricket"	Adyar, Chennai	
4	12103	Ram	"Soccer", "Hockey"	Gopalapuram, Chennai	
5					
6					
7					
8					
9					
10					

Fig 13.4.3 CSV Field Data with Double quotes in Excel

13.4.4 Rules to be followed to format data in a CSV file

1. Each record (row of data) is to be located on a separate line, delimited by a line break by pressing enter key. For example:↵

   xxx,yyy↵

   ↵ denotes enter Key to be pressed

2. The last record in the file may or may not have an ending line break. For example:

   ppp, qqq ↵

   yyy, xxx

3. There may be an optional header line appearing as the first line of the file with the same format as normal record lines. The header will contain names corresponding to the fields in the file and should contain the same number of fields as the records in the rest of the file. For example:

   field_name1,field_name2,field_name3 ↵
   aaa,bbb,ccc ↵
   zzz,yyy,xxx CRLF(**Carriage Return and Line Feed**)

4. Within the header and each record, there may be one or more fields, separated by commas. Spaces are considered part of a field and should not be ignored. The last field in the record must not be followed by a comma. For example: Red , Blue

5. Each field may or may not be enclosed in double quotes. If fields are not enclosed with double quotes, then double quotes may not appear inside the fields. For example:

```
"Red","Blue","Green"↵      #Field data with doule quotes
Black,White,Yellow         #Field data without doule quotes
```

6. Fields containing line breaks (CRLF), double quotes, and commas should be enclosed in double-quotes. For example:

```
Red, “,”, Blue CRLF   # comma itself is a field value.so it is enclosed with double  quotes
Red, Blue , Green
```

7. If double-quotes are used to enclose fields, then a double-quote appearing inside a field must be preceded with another double quote. For example:

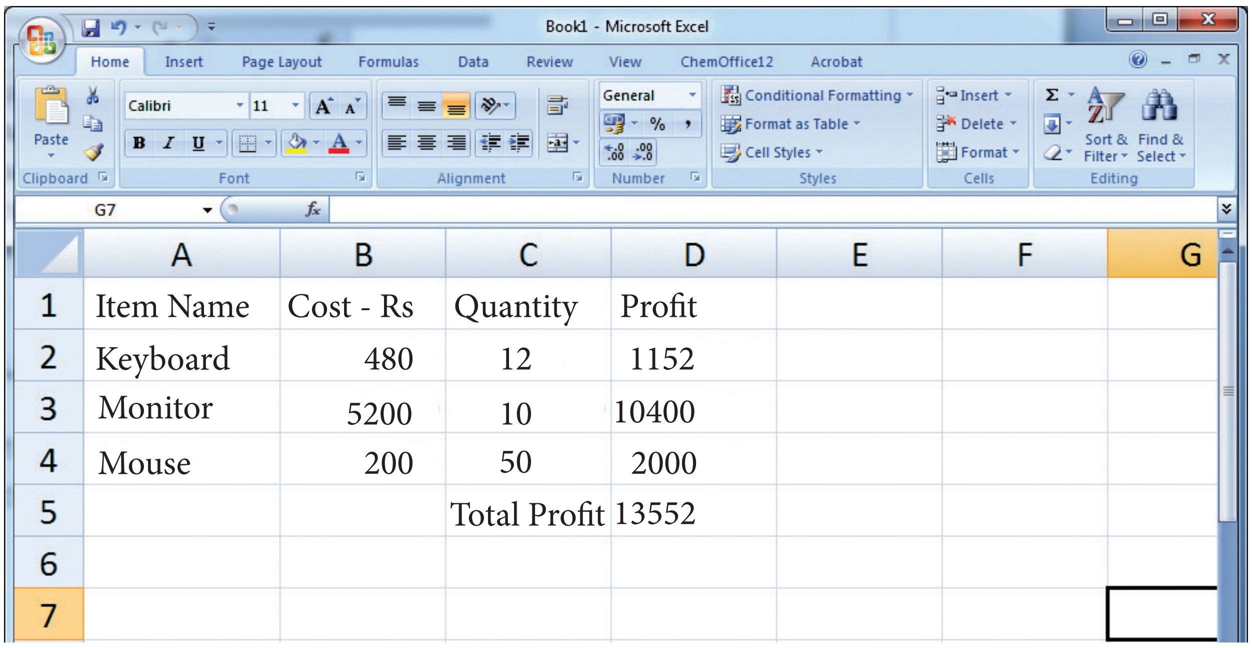
```
“““Red”””,  “““Blue”””, “““Green””” CRLF   # since double quotes is a field value it is enclosed with another
, ,  White                                    double  quotes
```

Note

The last row in the above example (, , White) begins with two commas because the first two fields of that row were empty in our spreadsheet. Don't delete them — the two commas are required so that the fields correspond from row to row. They cannot be omitted.

13.5 Create A CSV File Using Microsoft Excel

To create a CSV file using Microsoft Excel, launch Excel and then open the file you want to save in CSV format. For example, below is the data contained in our sample Excel worksheet:

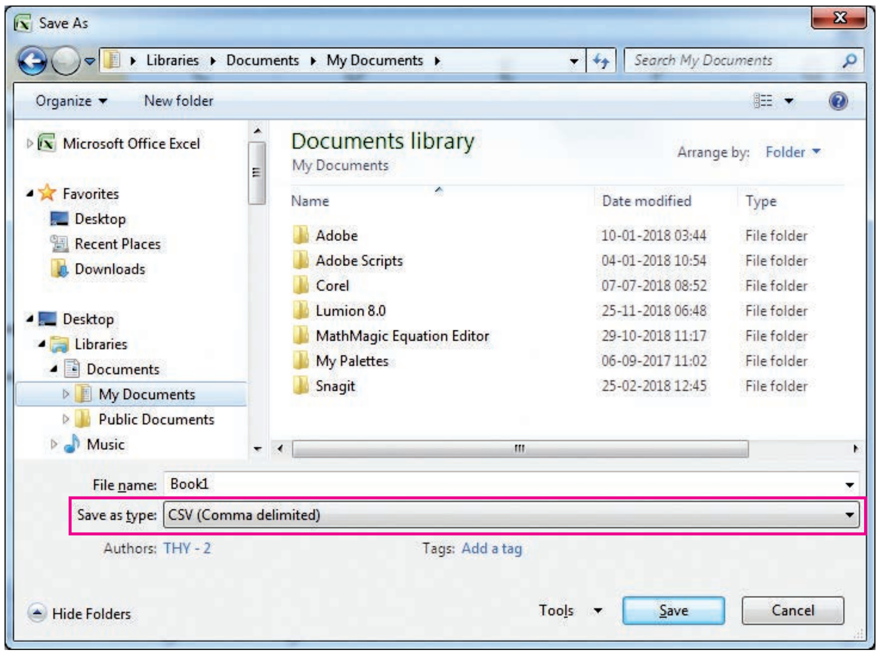


	A	B	C	D	E	F	G
1	Item Name	Cost - Rs	Quantity	Profit			
2	Keyboard	480	12	1152			
3	Monitor	5200	10	10400			
4	Mouse	200	50	2000			
5			Total Profit	13552			
6							
7							

Fig 13.5 Sample Worksheet Data

Once the data is entered in the worksheet, select **File** → **Save** As option, and for the "Save as type option", select CSV (Comma delimited) or type the file name along with extension .csv.

Saving excel file as CSV



Name	Date modified	Type
Adobe	10-01-2018 03:44	File folder
Adobe Scripts	04-01-2018 10:54	File folder
Corel	07-07-2018 08:52	File folder
Lumion 8.0	25-11-2018 06:48	File folder
MathMagic Equation Editor	29-10-2018 11:17	File folder
My Palettes	06-09-2017 11:02	File folder
Snagit	25-02-2018 12:45	File folder

File name: Book1

Save as type: CSV (Comma delimited)

Authors: THY - 2 Tags: Add a tag

Fig 13.6 Save As dialog box

After you save the file, you are free to open it up in a text editor to view it or to edit it manually. Its contents will resemble the following:

```
Item Name, Cost-Rs, Quantity, Profit
Keyboard, 480, 12, 1152
Monitor, 5200, 10, 10400
Mouse, 200, 50, 2000
,,Total Profit =,13552
```

13.5.1 Microsoft Excel to open a CSV file

If Microsoft Excel has been installed on the computer, **by default CSV files should open automatically in Excel when the file is double-clicked.** If you are getting an **Open With** prompt when opening the CSV file, choose Microsoft Excel from the available programs to open the file.

Alternatively, you can open Microsoft Excel and in the menu bar, select **File → Open,** and select the CSV file. If the file is not listed, make sure to change the file type to be opened to Text Files (*.prn, *.txt, *.csv).

If both MS Excel and Open Office calc is installed in the computer, by default the CSV file will be opened in MS Excel.

13.6 Read and write a CSV file Using Python

Python provides a module named **CSV,** using this you can do several operations on the CSV files. The CSV library contains objects and other code to **read, write, and process** data from and to CSV files.

CSV files have been used extensively in e-commerce applications because they are considered very easy to process.

13.6.1 Read a CSV File Using Python

There are two ways to read a CSV file.

1. Use the csv module's reader function
2. Use the DictReader class.

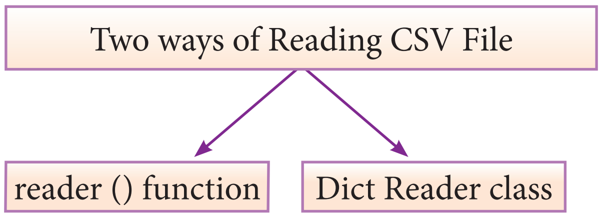


Fig 13.7 Ways to read CSV file

When you want to read from or write to a file ,you need to open it. Once the reading is over it needs to be closed. So that, resources that are tied with the file are freed. Hence, in Python, a file operation takes place in the following order

Step 1 Open a file

Step 2 Perform Read or write operation

Step 3 Close the file

Note

File name or the complete path name can be represented either with in " " or in ' ' in the open command.

Python has a built-in function open() to open a file. This function returns a **file object,** also called a **handle,** as it is used to read or modify the file accordingly.

For Example

```
>>> f = open("sample.txt")  # open file in current directory and f is file object
>>> f = open('c:\pyprg\ch13sample5.csv') # specifying full path
```

You can specify the mode while opening a file. In mode, you can specify whether you want to read **'r',** write **'w'** or append **'a'** to the file. you can also specify "text or binary" in which the file is to be opened.

The default is reading in text mode. In this mode, while reading from the file the data would be in the format of **strings.**

On the other hand, **binary mode returns bytes and this is the mode to be used when dealing with non-text files like image or exe files.**

Python File Modes

Mode	Description
'r'	Open a file for reading. (default)
'w'	Open a file for writing. Creates a new file if it does not exist or truncates the file if it exists.
'x'	Open a file for exclusive creation. If the file already exists, the operation fails.

'a'	Open for appending at the end of the file without truncating it. Creates a new file if it does not exist.
't'	Open in text mode. (default)
'b'	Open in binary mode.
'+'	Open a file for updating (reading and writing)

f=**open("sample.txt")**

#equivalent to 'r' or 'rt'

f = open("sample.txt",'w') *# write in text mode*

f = open("image1.bmp",'r+b') *# read and write in binary mode*

Python has a garbage collector to clean up unreferenced objects but, one must not rely on it to close the file.

```
f = open("test.txt")        #  since no mode is specified the default mode rt is used
# perform file operations
f.close()
```

The above method is **not entirely safe.** If an **exception** occurs when you are performing some operation with the file, the code exits without closing the file. The best way to do this is using the **"with"** statement. This ensures that the file is closed when the block inside **with** is exited. You need not to explicitly call the close() method. It is done internally.

```
with open("test.txt",'r') as f:
# f is file object to perform file operations
```

Closing a file will free up the resources that were tied with the file and is done using Python close() method.

```
f = open("sample.txt")
# perform file operations
f.close()
```

13.6.1.1 CSV Module's Reader Function

You can read the contents of CSV file with the help of **csv.reader()** function. **The reader function is designed to take each line of the file and make a list of all columns.** Then, you just choose the column you want the variable data for. Using this function one can read data from csv files of different formats like quotes (" "), pipe (|) and comma (,).

The syntax for csv.reader() is

```
csv.reader(fileobject,delimiter,fmtparams)
```

where

file object :-	passes the path and the mode of the file
delimiter :-	an optional parameter containing the standard dilects like , | etc can be omitted
fmtparams:	optional parameter which help to override the default values of the dialects like skipinitialspace,quoting etc. Can be omitted

1. CSV file - data with default delimiter comma (,)
2. CSV file - data with Space at the beginning
3. CSV file - data with quotes
4. CSV file - data with custom Delimiters

13.6.1.1.1 CSV file with default delimiter comma (,)

The following program read a file called "sample1.csv" with default delimiter comma (,) and print row by row.

```
#importing csv
import csv
#opening the csv file which is in different location with read mode
with open('c:\pyprg\sample1.csv', 'r') as F:
#other way to open the file is f= ('c:\pyprg\sample1.csv', 'r')
        reader = csv.reader(F)
# printing each line of the Data row by row
for row in reader:
       print(row)
F.close()
```

OUTPUT

```
       ['SNO', 'NAME', 'CITY']
       ['12101', 'RAM', 'CHENNAI']
       ['12102', 'LAVANYA', 'TIRUCHY']
       ['12103', 'LAKSHMAN', 'MADURAI']
```

13.6.1.1.2. CSV files- data with Spaces at the beginning

Consider the following file "sample2.csv" containing the following data when opened through notepad

```
Topic1, Topic2, Topic3,
one, two, three
Example1, Example2, Example3
```

The following program read the file through Python using "csv.reader()".

```
import csv
csv.register_dialect('myDialect',delimiter = ',',skipinitialspace=True)
F=open('c:\pyprg\sample2.csv','r')
reader = csv.reader(F, dialect='myDialect')
for row in reader:
        print(row)
F.close()
```

OUTPUT

```
['Topic1', 'Topic2', 'Topic3']
['one', 'two', 'three']
['Example1', 'Example2', 'Example3']
```

As you can see in "sample2.csv" there are spaces after the delimiter due to which the output is also displayed with spaces.

These whitespaces can be removed, by registering new dialects using **csv.register_dialect()** class of csv module. **A dialect describes the format of the csv file that is to be read.** In dialects the parameter **"skipinitialspace"** is used for removing whitespaces after the delimiter.

Note

By default "skipinitialspace" has a value false

The following program reads "sample2.csv" file, which contains spaces after the delimiter.

```
import csv
csv.register_dialect('myDialect',delimiter = ',',skipinitialspace=True)
F=open('c:\pyprg\sample2.csv','r')
reader = csv.reader(F, dialect='myDialect')
for row in reader:
        print(row)
F.close()
```

OUTPUT

```
['Topic1', 'Topic2', 'Topic3']
['one', 'two', 'three']
['Example1', 'Example2', 'Example3']
```

Note

A dialect is a class of csv module which helps to define parameters for reading and writing CSV. It allows you to create, store, and re-use various formatting parameters for your data.

13.6.1.1.3 CSV File-Data With Quotes

You can read the csv file with quotes, by registering new dialects using csv.register_dialect() class of csv module.

Here, we have quotes.csv file with following data.

```
SNO,Quotes
1, "The secret to getting ahead is getting started."
2, "Excellence is a continuous process and not an accident."
3, "Work hard dream big never give up and believe yourself."
4, "Failure is the opportunity to begin again more intelligently."
5, "The successful warrior is the average man, with laser-like focus."
```

The following Program read "quotes.csv" file, where delimiter is comma (,) but the quotes are within quotes (" ").

```
import csv
csv.register_dialect('myDialect',delimiter = ',',skipinitialspace=True)
f=open('c:\pyprg\quotes.csv','r')
reader = csv.reader(f, dialect='myDialect')
for row in reader:
    print(row)
```

OUTPUT

```
['SNO', 'Quotes']
['1', 'The secret to getting ahead is getting started.']
['2', 'Excellence is a continuous process and not an accident.']
['3', 'Work hard dream big never give up and believe yourself.']
['4', 'Failure is the opportunity to begin again more intelligently.']
['5', 'The successful warrior is the average man, with laser-like focus. ']
```

In the above program, register a dialect with name myDialect. Then, we used **csv. QUOTE_ALL to display all the characters** after double quotes.

13.6.1.1.4 CSV files with Custom Delimiters

You can read CSV file having custom delimiter by registering a new dialect with the help of csv.register_dialect().

In the following file called "sample4.csv",each column is separated with | (Pipe symbol)

```
Roll No | Name | City
12101 | Arun | Chennai
12102 | Meena | Kovai
12103 | Ram | Nellai
```

The following program read the file "sample4.csv" with user defined delimiter "|"

```
import csv
csv.register_dialect('myDialect', delimiter = '|')
with open('c:\pyprg\sample4.csv', 'r') as f:
    reader = csv.reader(f, dialect='myDialect')
    for row in reader:
        print(row)
f.close()
```

OUTPUT

```
['RollNo', 'Name', 'City']
['12101', 'Arun', 'Chennai']
['12102', 'Meena', 'Kovai']
['12103', 'Ram', 'Nellai']
```

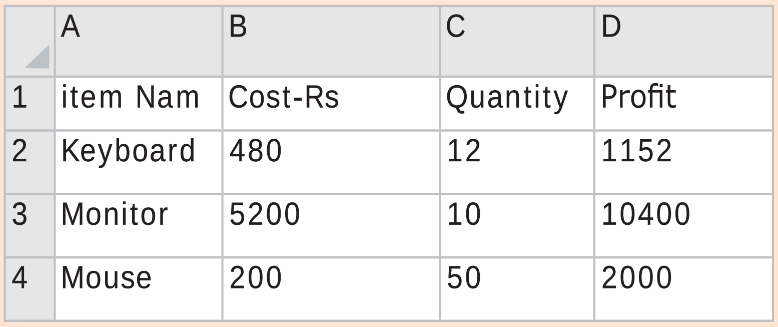
In the above program, a new dialects called myDialect is registered. Use the delimiter=| where a pipe (|) is considered as column separator.

13.6.2 Read a specific column In a File

To get the specific columns like only Item Name and profit for the "sample5.csv" file . Then you have to do the following:

```
import csv
#opening the csv file which is in different location with read mode
f=open("c:\pyprg\ch13sample5.csv",'r')
#reading the File with the help of csv.reader()
readFile=csv.reader(f)
#printing the selected column
for col in readFile :
    print (col[0],col[3])
f.close()
```

sample5.csv File in Excel



	A	B	C	D
1	item Nam	Cost-Rs	Quantity	Profit
2	Keyboard	480	12	1152
3	Monitor	5200	10	10400
4	Mouse	200	50	2000

sample5.csv File with selected col

OUTPUT

```
Item Name Profit
Keyboard 1152
Monitor 10400
Mouse 2000
```

13.6.3 Read A CSV File And Store It In A List

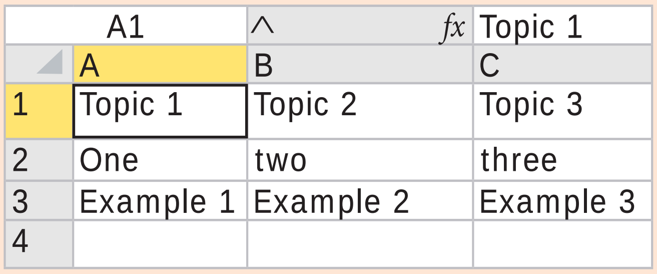
In this topic you are going to read a CSV file and the contents of the file will be stored as a list. The syntax for storing in the List is

```
list = []                    # Start as the empty list
list.append(element)         # Use append() to add elements
```

For example all the row values of “sample.csv” file is stored in a list using the following program

```
import csv
# other way of declaring the filename
inFile= 'c:\pyprg\sample.csv'
F=open(inFile,'r')
reader = csv.reader(F)
# declaring array
arrayValue = []
# displaying the content of the list
for row in reader:
    arrayValue.append(row)
    print(row)
F.close()
```

sample.csv opened in MS-Excel



A1	^ *fx*	Topic 1

	A	B	C
1	Topic 1	Topic 2	Topic 3
2	One	two	three
3	Example 1	Example 2	Example 3
4			

OUTPUT

```
['Topic1', 'Topic2', 'Topic3']
[' one', 'two', 'three']
['Example1', 'Example2', 'Example3']
```

Note

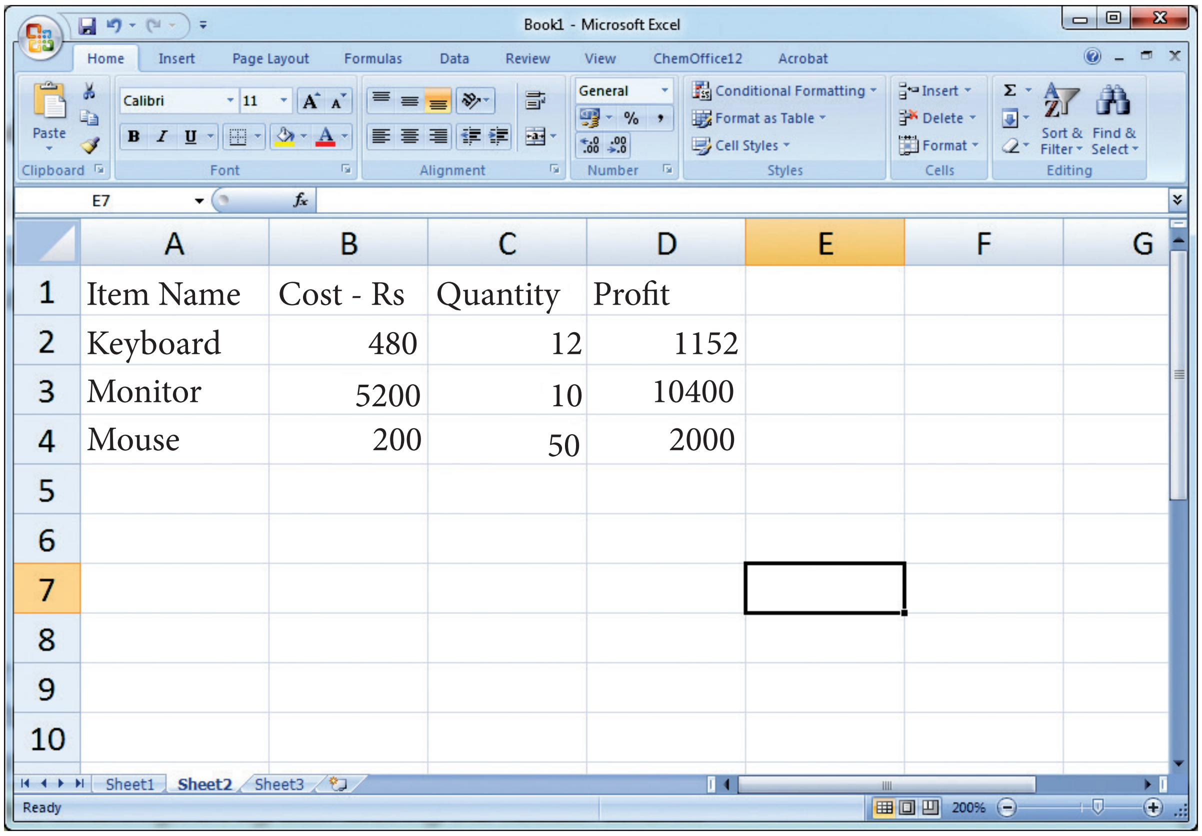
A list is a data structure in Python that is a mutable, or changeable, ordered sequence of elements.

DO YOU KNOW?

List literals are written within square brackets [ ]. Lists work similarly to strings

13.6.4 Read A CSV File And Store A Column Value In A List For Sorting

In this program you are going to read a selected column from the “sample6.csv” file by getting from the user the column number and store the content in a list.



	A	B	C	D
1	Item Name	Cost - Rs	Quantity	Profit
2	Keyboard	480	12	1152
3	Monitor	5200	10	10400
4	Mouse	200	50	2000

Fig 13.6.4 CSV file Data for a selected column for sorting

Since the row heading is also get sorted, to avoid that the first row should be skipped. This is can be done by using the command "next()". The list is sorted and displayed.

```
#  sort a selected column given by user leaving the header column in
# descending order of value
import csv
#  other way of declaring the filename
inFile= 'c:\pyprg\sample6.csv'
#  opening the csv file which is in the same location of where the current #
python file
```

```
F=open(inFile,'r')
#  reading the File with the help of csv.reader()
reader = csv.reader(F)
#  skipping the first row(heading)
next(reader)
#  declaring a list
arrayValue = []
a = int(input ("Enter the column number between 0 to 3:-"))
#  sorting a particular column-cost
for row in reader:
        arrayValue.append(row[a])
arrayValue.sort(reverse=True)
for row in arrayValue:
        print (row)
F.close()
```

OUTPUT

```
Enter the column number between 0 to 3:- 2
50
12
10
```

Read a specific column in a csv file and display its result in Ascending order

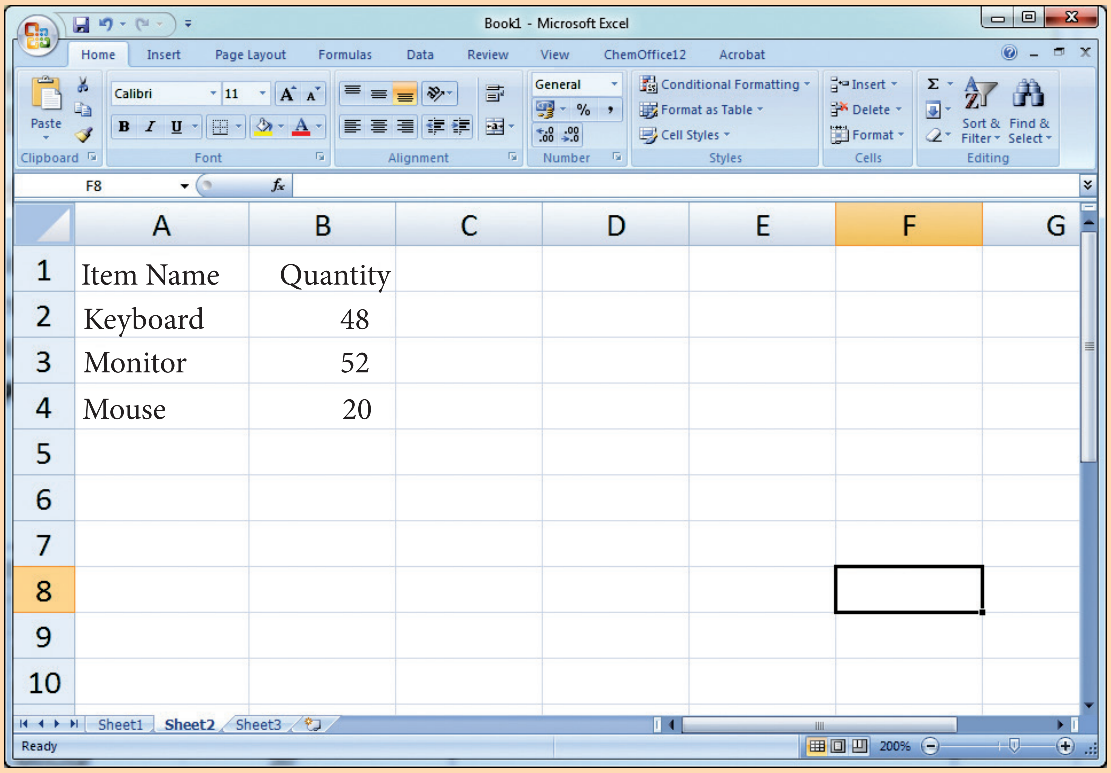
DO YOU KNOW?

list_name.sort() command arranges a list value in ascending order. list_name.sort(reverse=True) is used to arrange a list in descending order

13.6.5 Sorting A CSV File With A Specified Column

In this program you are going to see the "sample8.csv" file's entire content is transferred to a list. Then the list of rows is sorted and displayed in ascending order of quantity. To sort by more than one column you can use itemgetter with multiple indices: operator .itemgetter (1,2), The content of "sample8.csv" is

sample8.csv in Excel screen



	A	B	C	D	E	F	G
1	Item Name	Quantity					
2	Keyboard	48					
3	Monitor	52					
4	Mouse	20					
5							
6							
7							
8							
9							
10							

sample8.csv in Notepad

```
ItemName ,Quantity
Keyboard, 48
Monitor,52
Mouse ,20
```

Fig 13.6.5 CSV file Data into a list for sorting

The following program do the task mentioned above using **operator.itemgetter(col_no)**

```
#Program to sort the entire row by using a specified column.
# declaring multiple header files
import csv ,operator
#One more way to read the file
data = csv.reader(open('c:\pyprg\sample8.csv'))
next(data)                    #(to omit the header)
#using operator module for sorting multiple columns
sortedlist = sorted (data, key=operator.itemgetter(1))   # 1 specifies we want to sort
                                                          # according to  second column

for row in sortedlist:
     print(row)
```

OUTPUT

```
     ['Mouse ', '20']
     ['Keyboard ', '48']
     ['Monitor', '52']
```

Note

The sorted() method sorts the elements of a given item in a specific order – Ascending or Descending. Sort() method which performs the same way as sorted(). Only difference, sort() method doesn't return any value and changes the original list itself.

Add one more column "cost" in "sample8.csv" and sort it in descending order of cost by using the syntax
sortedlist = sorted(data, key=operator.itemgetter(Col_number),reverse=True)

13.6.6 Reading CSV File Into A Dictionary

To read a CSV file into a dictionary can be done by using **DictReader** method of csv module which works similar to the reader() class but creates an object which maps data to a dictionary. The keys are given by the fieldnames as parameter. **DictReader** works by reading the first line of the CSV and using each comma separated value in this line as a **dictionary key.** The columns in each subsequent row then behave like dictionary values and can be accessed with the appropriate key (i.e. fieldname).

If the first row of your CSV does not contain your column names, you can pass a fieldnames parameter into the DictReader's constructor to assign the dictionary keys manually.

The main difference between the csv.reader() and DictReader() is in simple terms csv.reader and csv.writer work with **list/tuple,** while csv.DictReader and csv.DictWriter work with dictionary. csv.DictReader and csv.DictWriter take additional argument fieldnames that are used as dictionary keys.

For Example Reading "sample8.csv" file into a dictionary

```
import csv
filename = 'c:\pyprg\sample8.csv'
input_file =csv.DictReader(open(filename,'r'))
for row in input_file:
    print(dict(row))                    #dict() to print data
```

OUTPUT

```
    {'ItemName ': 'Keyboard ', 'Quantity': '48'}
    {'ItemName ': 'Monitor', 'Quantity': '52'}
    {'ItemName ': 'Mouse ', 'Quantity': '20'}
```

In the above program, DictReader() is used to read "sample8.csv" file and map into a dictionary. Then, the function dict() is used to print the data in dictionary format without order.

Remove the dict() function from the above program and use print(row).Check you are getting the following output

```
OrderedDict([('ItemName ', 'Keyboard '), ('Quantity', '48')])
OrderedDict([('ItemName ', 'Monitor'), ('Quantity', '52')])
OrderedDict([('ItemName ', 'Mouse '), ('Quantity', '20')])
```

13.6.7 Reading CSV File With User Defined Delimiter Into A Dictionary

You can also register new dialects and use it in the DictReader() methods. Suppose "sample8.csv" is in the following format

```
ItemName|Quantity
Keyboard|48
Monitor|52
Mouse|20
```

Then "sample8.csv" can be read into a dictionary by registering a new dialect

```
import csv
csv.register_dialect('myDialect',delimiter = '|',skipinitialspace=True)
filename = 'c:\pyprg\ch13\sample8.csv'
with open(filename, 'r') as csvfile:
      reader = csv.DictReader(csvfile, dialect='myDialect')
for row in reader:
      print(dict(row))
csvfile.close()
```

OUTPUT

```
{'ItemName':'Keyboard','Quantity': 48}
{'ItemName' :'Monitor':'Quantity':52}
{'ItemName': 'Mouse':'Quantity': 20}
```

Note

DictReader() gives OrderedDict by default in its output. An OrderedDict is a dictionary subclass which saves the order in which its contents are added. To remove the OrderedDict use dict().

13.7 Writing Data Into Different Types in Csv Files

As you know Python provides an easy way to work with CSV file and has csv module to read and write data in the csv file. In the previous topics, You have learned how to read CSV files in Python. In similar way, You can also write a new or edit an existing CSV files in Python.

1 Creating A New Normal CSV File

2 Modifying An Existing File

3 Writing On A CSV File with Quotes

4 Writing On A CSV File with Custom Delimiters

5 Writing On A CSV File with Lineterminator

6 Writing On A CSV File with Quotechars

7 Writing CSV File Into A Dictionary

8 Getting Data At Runtime And Writing In a File

13.7.1 Creating A New Normal CSV File

When you have a set of data that you would like to store inside a CSV file, it's time to do the opposite and use the writer function.

The csv.writer() method returns a writer object which converts the user's data into delimited strings on the given file-like object. The writerow() function writes a row of data into the specified file.

The syntax for csv.writer() is

```
csv.writer(fileobject,delimiter,fmtparams)
```

where

fileobject :-	passes the path and the mode of the file
delimiter :-	an optional parameter containing the standard dilects like , | etc can be omitted
fmtparams :	optional parameter which help to override the default values of the dialects like skipinitialspace,quoting etc. can be omitted

You can **create** a normal CSV file using writer() function of csv module having default delimiter comma (,)

Here's an example.

The following Python program converts a List of data to a CSV file called "Pupil.csv" that uses, (comma) as a value separator.

```
Import csv
csvData = [['Student', 'Age'], ['Dhanush', '17'], ['Kalyani', '18'], ['Ram', '15']]
with open('c:\pyprg\ch13\Pupil.csv', 'w') as CF:
      writer = csv.writer(CF)                    #  CF is the file object
      writer.writerows(csvData)                  #  csvData is the List name
CF.close()
```

When you open the "Pupil.csv" file with a text editor, it will show the content as follows.

```
Student, Age
Dhanush, 17
Kalyani, 18
Ram, 15
```

In the above program, csv.writer() function converts all the data in the list "csvData" to strings and create the content as file like object. The writerows () function writes all the data in to the new CSV file "Pupil.csv".

Note

The writerow() function writes one row at a time. If you need to write all the data at once you can use writerows() method.

13.7.2 Modifying An Existing File

Making some changes in the data of the existing file or adding more data is called modification .For example the "student.csv" file contains the following data.

```
Roll No, Name, City
1, Harshini, Chennai
2, Adhith, Mumbai
3, Dhuruv, Bangalore
4, Krishna, Tiruchy
5, Venkat, Madurai
```

The following program modify the "student.csv" file by modifying the value of an existing row in student.csv

```
import csv
row = ['3', 'Meena','Bangalore']
with open('student.csv', 'r') as readFile:
        reader = csv.reader(readFile)
        lines = list(reader)         # list()- to store each row of data as a list
         lines[3] = row
with open('student.csv', 'w') as writeFile:
# returns the writer object which converts the user data with delimiter
        writer = csv.writer(writeFile)
#writerows()method writes multiple rows to a csv file
        writer.writerows(lines)
readFile.close()
writeFile.close()
```

When we open the student.csv file with text editor, then it will show:

```
Roll No, Name, City
1, Harshini, Chennai
2, Adhith, Mumbai
3, Meena, Bangalore
4, Krishna, Tiruchy
5, Venkat, Madurai
```

In the above program,the third row of "student.csv" is modified and saved. First the "student.csv" file is read by using csv.reader() function. Then, the list() stores each row of the file. The statement "lines[3] = row", changed the third row of the file with the new content in "row". The file object writer using writerows (lines) writes the values of the list to "student.csv" file.

13.7.2.1 ADDING NEW ROW

Sometimes, you may need to add new rows in the existing CSVfile. Adding a new row at the end of the file is called appending a row.

The following program add a new row to the existing "student.csv" file.

```
import csv
row = ['6', 'Sajini ', 'Madurai']
with open('student.csv', 'a') as CF:            # append mode to add data at the end
      writer = csv.writer(CF)
      writer.writerow(row)    # writerow() method write a single row of data in file
CF.close()
```

When "student.csv" file is opened with a text editor, it displays as follows

```
Roll No, Name, City
1, Harshini, Chennai
2, Adhith, Mumbai
3, Meena, Bangalore
4, Krishna, Tiruchy
5, Venkat, Madurai
6, Sajini, Madurai
```

In the above program, a new row is appended into "student.csv". For this, purpose only the CSV file is opened in 'a' append mode. Append mode write the value of row after the last line of the "student.csv file."

DO YOU KNOW?

The 'w' write mode creates a new file. If the file is already existing 'w' mode over writs it. Where as 'a' append mode add the data at the end of the file if the file already exists otherwise creates a new one

Note

writerow() takes 1-dimensional data (one row), and writerows takes 2-dimensional data (multiple rows) to write in a file.

13.7.3 CSV Files With Quotes

You can write the csv file with quotes, by registering new dialects using **csv.register_dialect()** class of csv module. The following program explains this.

```
import csv
info = [['SNO', 'Person', 'DOB'],
['1', 'Madhu', '18/12/2001'],
['2', 'Sowmya','19/2/1998'],
['3', 'Sangeetha','20/3/1999'],
['4', 'Eshwar', '21/4/2000'],
['5', 'Anand', '22/5/2001']]
csv.register_dialect('myDialect',quoting=csv.QUOTE_ALL)
with open('c:\pyprg\ch13\person.csv', 'w') as f:
        writer = csv.writer(f, dialect='myDialect')
        for row in info:
            writer.writerow(row)
f.close()
```

When you open "person.csv" file, we get following output :

```
"SNO","Person","DOB"
"1","Madhu","18/12/2001"
"2","Sowmya","19/2/1998"
"3","Sangeetha","20/3/1999"
"4","Eshwar","21/4/2000"
"5","Anand","22/5/2001"
```

In above program, a dialect named myDialect is registered(declared). Quoting=csv.QUOTE_ALL allows to write the double quote on all the values.

13.7.4 CSV Files With Custom Delimiters

A delimiter is a string used to separate fields. The default value is comma(,). You can have custom delimiter in CSV files by registering a new dialect with the help of **csv.register_dialect()**.This example Program is written using the custom delimiter pipe(|)

```
import csv
info = [['SNO', 'Person', 'DOB'],
['1', 'Madhu', '18/12/2001'],
['2', 'Sowmya','19/2/1998'],
['3', 'Sangeetha','20/3/1999'],
['4', 'Eshwar', '21/4/2000'],
['5', 'Anand', '22/5/2001']]
csv.register_dialect('myDialect',delimiter = '|')
with open('c:\pyprg\ch13\dob.csv', 'w') as f:
        writer = csv.writer(f, dialect='myDialect')
         for row in info:
            writer.writerow(row)
f.close()
```

When we open “dob.csv” file, we get the following output:

```
SNO|Person|DOB
1|Madhu|18/12/2001
2|Sowmya|19/2/1998
3|Sangeetha|20/3/1999
4|Eshwar|21/4/2000
5|Anand|22/5/2001
```

In the above program, a dialect with delimiter as pipe(|)is registered. Then the list “info” is written into the CSV file “dob.csv”.

Note

The dialect parameter skipinitialspace when it is True, whitespace immediately following the delimiter is ignored. The default is False.

13.7.5 CSV File With A Line Terminator

A Line Terminator is a string used to terminate lines produced by writer. The default value is \r or \n. We can write csv file with a line terminator in Python by registering new dialects using csv.register_dialect() class of csv module. For Example

```
import csv
Data = [['Fruit', 'Quantity'], ['Apple', '5'], ['Banana', '7'], ['Mango', '8']]
csv.register_dialect('myDialect', delimiter = '|', lineterminator = '\n')
with open('c:\pyprg\ch13\line.csv', 'w') as f:
    writer = csv.writer(f, dialect='myDialect')
    writer.writerows(Data)
f.close()
```

When we open the line.csv file, we get following output with spacing between lines:

```
Fruit|Quantity
Apple|5
Banana|7
Mango|8
```

In the above code, the new dialect “myDialect uses the delimiter='|' where a | (pipe) is considered as column separator. The line terminator='\r\n\r\n' separates each row and displays the data after one blank line in Notepad.

Note

Python's CSV module only accepts \r\n, \n or \r as line terminator

13.7.6 CSV File with quote characters

You can write the CSV file with custom quote characters, by registering new dialects using csv.register_dialect() class of csv module.

```
import csv
csvData = [['SNO','Items'], ['1','Pen'], ['2','Book'], ['3','Pencil']]
csv.register_dialect('myDialect',delimiter = '|',quotechar = '"',
        quoting=csv.QUOTE_ALL)
with open('c:\pyprg\ch13\quote.csv', 'w') as csvFile:
        writer = csv.writer(csvFile, dialect='myDialect')
        writer.writerows(csvData)
print("writing completed")
csvFile.close()
```

When you open the "quote.csv" file in notepad, we get following output:

```
"SNO"|"Items"
"1"|"Pen"
"2"|"Book"
"3"|"Pencil"
```

In the above program, myDialect uses pipe (|) as delimiter and quotechar as doublequote '"' to write inside the file.

13.7.7 Writing CSV File Into A Dictionary

Using DictWriter() class of csv module, we can write a csv file into a dictionary. It creates an object which maps data into a dictionary. The keys are given by the fieldnames parameter. The following program helps to write the dictionary in to file.

```
import csv
data = [{'MOUNTAIN' : 'Everest', 'HEIGHT': '8848'},
        {'MOUNTAIN' : 'Anamudi ', 'HEIGHT': '2695'},
        {'MOUNTAIN' : 'Kanchenjunga', 'HEIGHT': '8586'}]
with open('c:\pyprg\ch13\peak.csv', 'w') as CF:
fields = ['MOUNTAIN', 'HEIGHT']
        w = csv.DictWriter(CF, fieldnames=fields)
        w.writeheader()
        w.writerows(data)
        print("writing completed")
CF.close()
```

When you open the “peak.csv” file in notepad, you get the following output:

```
MOUNTAIN,HEIGHT
Everest,8848
Anamudi,2695
Kanchenjunga,8586
```

In the above program, use fieldnames as headings of each column in csv file. Then, use a DictWriter() to write dictionary data into “peak.csv” file.

13.7.7.1 Writing Dictionary Into CSV File With Custom Dialects

```
import csv
csv.register_dialect(‘myDialect’, delimiter = ‘|’, quoting=csv.QUOTE_ALL)
with open(‘c:\pyprg\ch13\grade.csv’, ‘w’) as csvfile:
    fieldnames = [‘Name’, ‘Grade’]
    writer = csv.DictWriter(csvfile, fieldnames=fieldnames, dialect=”myDialect”)
    writer.writeheader()
    writer.writerows([{‘Grade’: ‘B’, ‘Name’: ‘Anu’},
                {‘Grade’: ‘A’, ‘Name’: ‘Beena’},
                {‘Grade’: ‘C’, ‘Name’: ‘Tarun’}])
print(“writing completed”)
```

When we open grade.csv file, it will contain following output:

```
“Name”|”Grade”
”Anu”|”B”
”Beena”|”A”
“Tarun”|”C”
```

In the above program, a custom dialect called myDialect with pipe (|) as delimiter uses the fieldnames as headings of each column to write in a csv file. Finally, we use a DictWriter() to write dictionary data into “grade.csv” file.

13.7.8 Getting Data At Runtime And Writing It In a CSV File

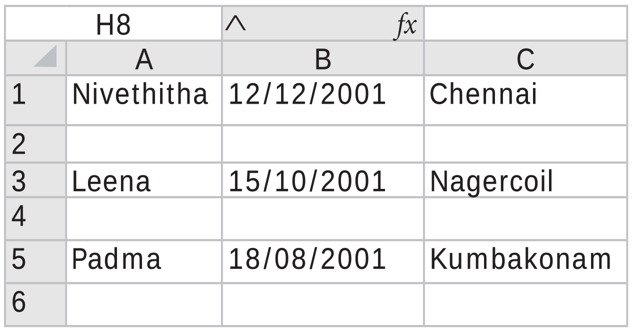
You can even accept the data through keyboard and write in to a CSV file. For example the following program accept data from the user through key board and stores it in the file called “dynamicfile.csv”. It also displays the content of the file.

```
import csv
with open('c:\pyprg\ch13\dynamicfile.csv', 'w') as f:
      w = csv.writer(f)
      ans='y'
      while (ans=='y'):
          name = input("Name?: ")
          date = input("Date of birth: ")
          place = input("Place: ")
          w.writerow([name, date, place])
          ans=input("Do you want to enter more y/n?: ")
      F=open('c:\pyprg\ch13\dynamicfile.csv','r')
      reader = csv.reader(F)
      for row in reader:
          print(row)
      F.close()
```

OUTPUT

```
Name?: Nivethitha
Date of birth: 12/12/2001
Place: Chennai
Do you want to enter more y/n?: y
Name?: Leena
Date of birth: 15/10/2001
Place: Nagercoil
Do you want to enter more y/n?: y
Name?: Padma
Date of birth: 18/08/2001
Place: Kumbakonam
Do you want to enter more y/n?: n
['Nivethitha', '12/12/2001', 'Chennai']
[]
['Leena', '15/10/2001', 'Nagercoil']
[]
['Padma', '18/08/2001', 'Kumbakonam']
```

MS Excel output



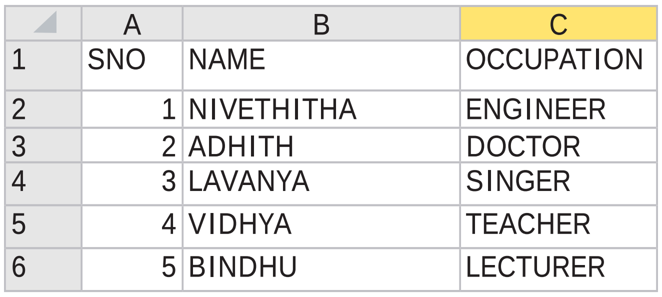
H8	^ *fx*		
	A	B	C
1	Nivethitha	12/12/2001	Chennai
2			
3	Leena	15/10/2001	Nagercoil
4			
5	Padma	18/08/2001	Kumbakonam
6			

Points to remember:

- A CSV file is a human readable text file where each line has a number of fields, separated by commas or some other delimiter
- Excel is a binary file whereas CSV format is a plain text format
- The two ways to read a CSV file are using csv.reader() function and using DictReader class.
- The default mode of csv file in reading and writing is text mode
- Binary mode can be be used when dealing with non-text files like image or exe files.
- Python has a garbage collector to clean up unreferenced objects
- close() method will free up the resources that were tied with the file
- By default CSV files should open automatically in Excel
- The CSV library contains objects and other code to read, write, and process data from and to CSV files.
- "skipinitialspace" is used for removing whitespaces after the delimiter
- To sort by more than one column operator.itemgetter() can be used
- DictReader() class of csv module creates an object which maps data to a dictionary
- CSV file having custom delimiter is read with the help of csv.register_dialect().
- To sort by more than one column itemgetter() with multiple indices is used.
- csv.reader and csv.writer work with list/tuple, while csv.DictReader and csv.DictWriter work with dictionary .
- csv.DictReader and csv.DictWriter take additional argument fieldnames that are used as dictionary keys.
- The function dict() is used to print the data in dictionary format with order.
- The csv.writer() function returns a writer object which converts the user's data into delimited strings.
- The writerow() function writes one row at a time. Writerows() method is used to write all the data at once
- Adding a new row at the end of the file is called appending a row.

Hands on Experience

1. Write a Python program to read the following Namelist.csv file and sort the data in alphabetically order of names in a list and display the output



	A	B	C
1	SNO	NAME	OCCUPATION
2	1	NIVETHITHA	ENGINEER
3	2	ADHITH	DOCTOR
4	3	LAVANYA	SINGER
5	4	VIDHYA	TEACHER
6	5	BINDHU	LECTURER

2. Write a Python program to accept the name and five subjects mark of 5 students .Find the total and store all the details of the students in a CSV file

Evaluation

Part - I

MQDPW

I Choose the best answer

1. A CSV file is also known as a ….

(A) Flat File (B) 3D File

(C) String File (D) Random File

2. The expansion of CRLF is

(A) Control Return and Line Feed

(B) Carriage Return and Form Feed

(C) Control Router and Line Feed

(D) Carriage Return and Line Feed

3. Which of the following module is provided by Python to do several operations on the CSV files?

(A) py (B) xls

(C) csv (D) os

4. Which of the following mode is used when dealing with non-text files like image or exe files?

(A) Text mode (B) Binary mode

(C) xls mode (D) csv mode

5. The command used to skip a row in a CSV file is

(A) next() (B) skip()

(C) omit() (D) bounce()

6. Which of the following is a string used to terminate lines produced by writer()method of csv module?

(A) Line Terminator (B) Enter key

(C) Form feed (D) Data Terminator

7. What is the output of the following program? import csv

```
d=csv.reader(open('c:\PYPRG\ch13\city.csv'))
next(d)
for row in d:
    print(row)
```

if the file called "city.csv" contain the following details

```
chennai,mylapore
mumbai,andheri
```

A) chennai,mylapore (B) mumbai,andheri

(C) chennai
mumba

(D) chennai,mylapore
mumbai,andheri

8. Which of the following creates an object which maps data to a dictionary?

(A) listreader() (B) reader()

(C) tuplereader() (D) DictReader ()

9. Making some changes in the data of the existing file or adding more data is called

(A)Editing (B) Appending

(C)Modification (D) Alteration

10. What will be written inside the file test.csv using the following program

```
import csv
D = [['Exam'],['Quarterly'],['Halfyearly']]
```

```
csv.register_dialect('M',lineterminator = '\n')
with open('c:\pyprg\ch13\line2.csv', 'w') as f:
    wr = csv.writer(f,dialect='M')
    wr.writerows(D)
f.close()
```

(A) Exam Quarterly Halfyearly (B) Exam Quarterly Halfyearly

(C) E
Q
H

(D) Exam,
Quarterly,
Halfyearly

Part - II

1. What is CSV File?
2. Mention the two ways to read a CSV file using Python.
3. Mention the default modes of the File.
4. What is use of next() function?
5. How will you sort more than one column from a csv file?Give an example statement.

Part - III

1. Write a note on open() function of python. What is the difference between the two methods?
2. Write a Python program to modify an existing file.
3. Write a Python program to read a CSV file with default delimiter comma (,).
4. What is the difference between the write mode and append mode.
5. What is the difference between reader()method and DictReader() class?

Part - IV

1. Differentiate Excel file and CSV file.
2. Tabulate the different mode with its meaning.
3. Write the different methods to read a File in Python.
4. Write a Python program to write a CSV File with custom quotes.
5. Write the rules to be followed to format the data in a CSV file.

REFERENCES

1. ***Python for Data Analysis, Data Wrangling with Pandas, NumPy, and IPython By William McKinney***
2. ***CSV File Reading and Writing - Python 3.7.0 documentation***
3. ***https://docs.python.org***