Python Data Processing and Encoding:
CSV (Comma Separated Values/Data) files to store data that are in tabular format into plain text & each line is treated as a data record in the file.

Define Delimiter:
It is a sequence of one or more characters used to specify the boundary between separate.

Example:
A delimiter is the comma character, or Space, or Gap or Colon etc.. a CSV file in Notepad, Microsoft Excel, OpenOffice Calc, and Google Docs.

Syntax: CSV formated text..!!
```
fname,lname,age,salary
nancy,davolio,33,$30000
erin,borakova,28,$25250
tony,raphael,35,$28700
```

In PYTHON Environment we can able to work with csv files, we can use built-in module is called CSV.
1. Open the file on required mode
2. Create the csv file represented object
3. Read or write or update the data

Syntax:
```
import csv
```

CSV Functions
1 csv.reader
2 csv.writer

EXAMPLE:Writing data in CSV file (Creating CSV File)
```
import csv
try:
    with open("MyFile.csv",mode='w',encoding='utf-8') as FP:
        a=csv.writer(FP,delimiter=',')
        data=[['STOCK','SALES','PRICE'],
              ['100','90','90$'],
              ['300','100','100$'],
              ['100','200','200$']]
        a.writerows(data)
        print("CSVFileCreatedSuccessfully")
except IOError:
    print("SorryCSVFileUnableTOCreate")
    print("ServerWriteProtected")
finally:
    print("FinallyBlockSuccess")
```

EXAMPLE:Appending data in CSV file..!!
```
import csv
try:
    with open("BigData.csv",mode='a',encoding='utf-8') as FP:
        a=csv.writer(FP,delimiter=',')
        data=[['10','9','9$'],
```

```
                ['0','0','0$'],
                ['10','2','2$']]
        a.writerows(data)
        print("CSVFileAppendSuccessfully")
except IOError:
    print("SorryCSVFileUnableTOAppend")
    print("ServerWriteProtected")
finally:
    print("FinallyBlockSuccess")

EXAMPLE: Reading CSV File Without Builtin Methods..!!
try:
    with open("BigData.csv",mode='r',encoding='utf-8') as FP:
        for data in FP:
            print(data)
        print("CSVFileReadSuccessfully")
except IOError:
    print("SorryCSVFileUnableToAppend")
    print("ServerWriteProtected")
finally:
    print("FinallyBlockSuccess")

Example:Reading CSV File
import csv
with open("MyFile.csv",'r') as FP:
    a=csv.reader(FP)
    data=[]
    for row in a:
        if len(row)!=0:
            data=data+[row]
print(data)

#without using csv module
path="MyFile.csv"
lines=[line for line in open(path)]
print(lines[0])

Python String strip() Method
It returns a copy of the string in which all chars have been stripped
from the beginning and the end of the string (default whitespace
characters).

Syntax:
str.strip([chars]);

Example:
path="MyFile.csv"
lines=[line for line in open(path)]
print(lines[0])
print(lines[1].strip())

Python String split() Method
It returns a list of all the words in the string, using str as the
separator.

Syntax
str.split(",").
```

Example:

```
path="MyFile.csv"
lines=[line for line in open(path)]
print(lines[0])
print(lines[1].strip())
print(lines[1].strip().split(','))
```

http://zetcode.com/python/csv/

PYTHON LOGGING:
Logging is a means of tracking events that happen when some software runs. Logging is important for software developing, debugging and running.

Example: Employee Log Book, System Log Files, Server Log File..!!

The main advantages of logging are:
1.We can use log files while performing debugging
2.We can provide statistics like number of requests per day etc
3.To implement logging, Python provides one inbuilt module logging.

Levels of Log Message
Debug : These are used to give Detailed information
Info : These are used to Confirm that things are working as expected
Warning : These are used an indication that something unexpected happened
Error : This tells that due to a more serious problem
Critical : Indicating that the program itself may be unable to continue running

Each built-in logging level has been assigned its numeric value.
1 NOTSET ==> 0
2 DEBUG ==> 10
3 INFO ==> 20

High Priority/Higher Level Messages
4 WARNING ==> 30
5 ERROR ==> 40
6 CRITICAL ==> 50

Example:

```
import logging
logging.basicConfig(filename='logging.txt',level=logging.CRITICAL)
print("Hey Demo for Logging")
logging.debug("This is Debug Message")
logging.info("This is Info Message")
logging.warning("This is Warning Message")
logging.error("This is Error Message")
logging.critical("This is Critical Message")
print("LoggingFileCreatedSuccessfully")
```

Example:

```
import logging
logging.basicConfig(filename='logging.txt',level=logging.ERROR)
print("Hey Demo for Logging")
logging.debug("This is Debug Message")
```

```
logging.info("This is Info Message")
logging.warning("This is Warning Message")
logging.error("This is Error Message")
logging.critical("This is Critical Message")
print("LoggingFileCreatedSuccessfully")
```

Configure Log File in different File Modes:
The above scripts by default data will be append to the log file.

Example: (Default Append)

```
#Specify Explicitly Overwriting
logging.basicConfig(filename='log.txt',level=logging.DEBUG,filemode='w
')
#Specify Explicitly
logging.basicConfig(filename='log.txt',level=logging.INFO,filemode='a'
)
#Default Mode
logging.basicConfig(filename='log.txt',level=logging.WARNING)
```

NOTE:
Default Level-Warning(30), Default File-Console, Default Mode-Append

Example:

```
import logging
logging.basicConfig(filename='logging.txt',level=logging.WARNING)
print("Hey Demo for Logging")
logging.debug("This is Debug Message")
logging.info("This is Info Message")
logging.warning("This is Warning Message")
logging.error("This is Error Message")
logging.critical("This is Critical Message")
print("LoggingFileCreatedSuccessfully")
```

Example:

```
import logging
logging.basicConfig(filename='logging.txt',level=logging.INFO)
logging.info('Hei DevTeam New Log Tracked..!!')
try:
    x=int(input('Enter First Number:'))
    y=int(input('Enter Second Number:'))
    print('The Result:',x/y)
except ZeroDivisionError as msg:
    print('Error Divide With Zero')
    logging.exception(msg)
except ValueError as msg:
    print('Integer Only')
    logging.exception(msg)
logging.info('Logging Finished Guys..!!')
```

How to format log messages:
Format keyword argument we can format messages
1. To display only level logging.basicConfig(format='%(levelname)s')

Example:

```
import logging
logging.basicConfig(filename='logging.txt',level=logging.DEBUG,format=
'%(asctime)s:%(levelname)s:%(message)s')
```

```
logging.info('Hei Dev. Team,New Log Track..!!')
try:
    x=int(input('Enter First Number:'))
    y=int(input('Enter Second Number:'))
    print('The Result:',x/y)
except ZeroDivisionError as msg:
    print('Error Divide With Zero')
    logging.exception(msg)
except ValueError as msg:
    print('Integer Only')
    logging.exception(msg)
logging.info('Logging Finished Guys..!!')
```

Example:

```
import logging
logging.basicConfig(filename='logging.txt',level=logging.INFO,format='
%(asctime)s:%(levelname)s:%(message)s', datefmt='%m/%d/%Y  %I:%M:%S
%p')
logging.info('HeyDevTeamNew Log Track..!!')
try:
    x=int(input('Enter First Number:'))
    y=int(input('Enter Second Number:'))
    print('The Result:',x/y)
except ZeroDivisionError as msg:
    print('Error Divide With Zero')
    logging.exception(msg)
except ValueError as msg:
    print('Integer Only')
    logging.exception(msg)
logging.info('Logging Finished..!!')
```

How to generate PDF using PYTHON:
We are going to learn how to generate PDF using python. Python has many supporting libraries and world's largest community. Sometimes we need to convert our text files into PDFs.

Installation:

```
$pip install fpdf
```

Example:

```
from fpdf import FPDF
pdf = FPDF()
pdf.add_page()
pdf.set_font('Arial',size=25)
pdf.cell(200,100,txt='Hello, You are in www.nareshit.com ', align='C')
pdf.output("txtpdf.pdf")
print("Say Hey PDF File Creted Successfully")
```

NOTE:
1 We import FPDF and create an object for it. Now we create a page. we set font_size, font_family. align represents the position of content in the cell, “C” -center, “R”-right, “L”-left.
2 pdf.output() method generate the output file. we need to give the output file name.

Generate pdf file from text file:
Example:

```
from fpdf import FPDF
f = open('Data.txt','r')
content = ''
for i in f:
    content =i
pdf = FPDF()
pdf.add_page()
pdf.set_font('Arial',size=18)
pdf.cell(100,10,txt=content,align='C')
pdf.output("txtpdf.pdf")
```

NOTE:
open() helps us to grab our text file from our current directory in our system. ‘r’ represents the mode of the file
Created one string empty variable "content" which helps us to store our text from the text file.
Used here a for loop to get that text from textfile and store it into "content".
pdf.cell() we reassign the txt parameter to "content" because we were storing the data in a variable content.