In [ ]:
```
import PyPDF2
```

In [ ]:
```
# importing required modules
import PyPDF2

# creating a pdf file object
pdfFileObj = open('combinedminutes.pdf', 'rb')

# creating a pdf file reader object, pdfReader
pdfReader = PyPDF2.PdfFileReader(pdfFileObj)

# printing number of pages in pdf file
# extract author name, file name, file size
print(pdfReader.numPages)

# creating an object of page object class
pageObj = pdfReader.getPage(1)

# extracting text from page
print(pageObj.extractText())

# closing the pdf file object
pdfFileObj.close()
```

In [ ]:
```
import os
os.getcwd()
```

In [ ]:
```
#Program to decrypt PDF file

import PyPDF2

# creating a pdf file reader object, pdfReader
pdfFile=open('encrypted.pdf', 'rb')
pdfReader = PyPDF2.PdfFileReader(pdfFile)

#checking whether the pdfReaderFile object is encrypted,
#if yes return True, else False
pdfReader.isEncrypted
```

In [ ]:
```
# creating an object of page object clas
pageObj=pdfReader.getPage(0)
```

In [ ]:
```
#usage of decrpyt method, it returns 1 if the password is correct,
#else 0 if incorrect
pdfReader.decrypt('rosebud')
```

In [ ]:
```
# creating an object of page object class
pageObj = pdfReader.getPage(0)
# extracting text from page
print(pageObj.extractText())
```

In [ ]:

```
#Program to create new PDF file by copying pages of existing PDF's

import PyPDF2

#opening files in read-binary mode
pdf1File = open('meetingminutes.pdf', 'rb')
pdf2File = open('meetingminutes2.pdf', 'rb')

#creating PdfFileReader objects of two pdf's
pdf1Reader = PyPDF2.PdfFileReader(pdf1File)
pdf2Reader = PyPDF2.PdfFileReader(pdf2File)

#creating object of class PdfFileWriter
pdfWriter = PyPDF2.PdfFileWriter()


for pageNum in range(pdf1Reader.numPages): # 19 pages-- 0 to 18
    pageObj = pdf1Reader.getPage(pageNum) # page index starts from 0 to 
    pdfWriter.addPage(pageObj) #adds page at the end to the pdfWriter

for pageNum in range(pdf2Reader.numPages): #21 pages --- 0 to 20
    pageObj = pdf2Reader.getPage(pageNum)
    pdfWriter.addPage(pageObj)   #19+21 = 40 pages

pdfOutputFile = open('combinedminutes.pdf', 'wb')

#use write() method
pdfWriter.write(pdfOutputFile)

pdfOutputFile.close()
pdf1File.close()
pdf2File.close()
```

In [ ]:

```
#Program to create new PDF file by rotating page of existing PDF's

import PyPDF2

minutesFile = open('meetingminutes.pdf', 'rb')
#creating object of class PdfFileReader
pdfReader = PyPDF2.PdfFileReader(minutesFile)

#extracting page number  -- 1(First Page)
page = pdfReader.getPage(0)

#rotating the page clockwise by 90 degree
page.rotateCounterClockwise(90)

#creating PdfFileWriter object
pdfWriter = PyPDF2.PdfFileWriter()

#add the rotated page to it using addPage()
pdfWriter.addPage(page)

#The resulting PDF will have one page, rotated 90 degrees clock-wise
resultPdfFile = open('rotatedPage.pdf', 'wb')
pdfWriter.write(resultPdfFile)

#close the files
resultPdfFile.close()
minutesFile.close()
```

In [ ]:
```
#Program to create new PDF file by overlaying  page of existing PDF's

import PyPDF2

#opening the file and extracting the page
minutesFile = open('meetingminutes.pdf', 'rb')
pdfReader = PyPDF2.PdfFileReader(minutesFile)
minutesFirstPage = pdfReader.getPage(0)

#opening the watermark file
pdfWatermarkReader = PyPDF2.PdfFileReader(open('watermark.pdf', 'rb'))

#adding wateramrk to the specified page
minutesFirstPage.mergePage(pdfWatermarkReader.getPage(0))

#creating the output PdfFileWriter object
pdfWriter = PyPDF2.PdfFileWriter()
#adding the watermarked first page
pdfWriter.addPage(minutesFirstPage)

#copying the remaining pages
for pageNum in range(1, pdfReader.numPages):
    pageObj = pdfReader.getPage(pageNum)
    pdfWriter.addPage(pageObj)

#creating the final PDF file
resultPdfFile = open('watermarkedCover.pdf', 'wb')
pdfWriter.write(resultPdfFile)

#closing the files
minutesFile.close()
resultPdfFile.close()
```

In [ ]:
```
#Program to create new encrypted PDF file

import PyPDF2
pdfFile = open('meetingminutes.pdf', 'rb')
pdfReader = PyPDF2.PdfFileReader(pdfFile)

pdfWriter = PyPDF2.PdfFileWriter()

for pageNum in range(pdfReader.numPages):
    pdfWriter.addPage(pdfReader.getPage(pageNum))

pdfWriter.encrypt('swordfish')
resultPdf = open('encryptedminutes.pdf', 'wb')

pdfWriter.write(resultPdf)
resultPdf.close()
```

Working with Word Documents

In [138]:
```
import docx
doc = docx.Document('demo.docx')
len(doc.paragraphs)
```

Out[138]: 7

In [139]:
```
doc.paragraphs[0].text
```

Out[139]: 'Document Title'

In [140]:
```
doc.paragraphs[1].text
```

Out[140]: 'A plain paragraph with some bold and some italic'

In [141]:
```
doc.paragraphs[2].text
```

Out[141]: 'Heading, level 1'

In [144]:
```
doc.paragraphs[3].text
```

Out[144]: 'Intense quote'

In [145]:
```
doc.paragraphs[5].text
```

Out[145]: 'first item in ordered list'

In [146]:
```
for para in doc.paragraphs:
    print(para.text)
    print("-------")
```

```
Document Title
-------
A plain paragraph with some bold and some italic
-------
Heading, level 1
-------
Intense quote
-------
first item in unordered list
-------
first item in ordered list
-------


-------
```

In [147]:
```
len(doc.paragraphs[1].runs)
```

Out[147]: 4

In [148]:
```
doc.paragraphs[1].runs[0].text
```

Out[148]: 'A plain paragraph with some '

In [149]:
```
doc.paragraphs[1].runs[1].text
```

Out[149]: 'bold'

In [150]:
```
doc.paragraphs[1].runs[2].text
```

Out[150]:
```
' and some '
```

In [151]:
```
doc.paragraphs[1].runs[3].text
```

Out[151]:
```
'italic'
```

In [152]:
```
for run in doc.paragraphs[1].runs:
    print(run.text)
    print("-------")
```

```
A plain paragraph with some
-------
bold
-------
 and some
-------
italic
-------
```

In [154]:
```
#to get the entire document as a single string
import docx

def getText(filename):
    doc = docx.Document(filename)
    fullText = []
    for para in doc.paragraphs:
        fullText.append(para.text)
    return '\n'.join(fullText)

doc_str=getText('demo.docx')
print(doc_str)
```

```
Document Title
A plain paragraph with some bold and some italic
Heading, level 1
Intense quote
first item in unordered list
first item in ordered list
```

In [ ]:
```
#writing word documents in python
import docx
doc = docx.Document() #new blank Document object
doc.add_paragraph('Hello world!') #add paragraph
```

In [ ]:
```
paraObj1 = doc.add_paragraph('This is a second paragraph.')
paraObj2 = doc.add_paragraph('This is a yet another paragraph.')
paraObj2.alignment=1 #0-left,1-center,2-right
paraObj1.add_run(' This text is being added to the second paragraph.')
doc.save('helloworld.docx')
```

In [124]:
```
#t retrieve the style info and adding new styles
doc=docx.Document("demo.docx")
doc.paragraphs[0].text
```

Out[124]:
```
'Document Title'
```

In [125]:
```
doc.paragraphs[0].style
```

Out[125]:
```
_ParagraphStyle('Title') id: 2437389766344
```

In [126]:
```
doc.paragraphs[0].style = 'Mamatha'
doc.paragraphs[0].alignment=1  #0 for left, 1 for center, 2 for right
```

In [127]:
```
doc.paragraphs[1].text
```

Out[127]:
```
'A plain paragraph with some bold and some italic'
```

In [128]:
```
(doc.paragraphs[1].runs[0].text, doc.paragraphs[1].runs[1].text, doc.
paragraphs[1].runs[2].text, doc.paragraphs[1].runs[3].text)
```

Out[128]:
```
('A plain paragraph with some ', 'bold', ' and some ', 'italic')
```

In [130]:
```
doc.paragraphs[1].runs[0].style = 'QuoteChar'
doc.paragraphs[1].runs[1].underline = True
doc.paragraphs[1].runs[3].underline = True
doc.paragraphs[1].alignment = 2
doc.add_paragraph("Heelo World","Heading 1")
doc.save('restyled.docx')
```

In [ ]:
```
#To add Headings
import docx
doc = docx.Document()
doc.add_heading('Header 0', 0)
obj=doc.add_paragraph("SAI VIDYA INSTITUTE OF TECHNOLOGY")
obj.add_run("V SEMESTER")
doc.add_heading('Header 1', 1)
doc.add_heading('Header 2', 2)
doc.add_heading('Header 3', 3)
doc.add_heading('Header 4', 4)
doc.save('headings.docx')
```

In [ ]:
```
#To add pictures
import docx
doc = docx.Document()
doc.add_picture('photo.jpg')
doc.save('headings.docx')
```

In [ ]:
```
#To add linebreak/pagebreak
import docx
doc = docx.Document()
doc.add_paragraph('This is on the first page!')
doc.paragraphs[0].runs[0].add_break(docx.enum.text.WD_BREAK.PAGE) #Inserts page break
doc.add_paragraph('This is on the second page!')
doc.save('twoPage1.docx') #no of pages=2
```

In [ ]:
```
#To add linebreak/pagebreak
import docx
doc = docx.Document("twopage.docx")
paraObj=doc.add_paragraph('This is on the first page!')
paraObj.style="Mamatha"
paraObj.add_run('This is on the second page!')
doc.save('twoPage2.docx') #no of pages=2
```

In [134]:
```
#To create invitation on each page for every guest

import docx

# Open the guests.txt,pick up the name of the guests.
guestsTxt = open('guests.txt','r')

INTRO = "It would be a pleasure to have the company of"
ADDRESS = "at 11010 Memory Lane on the Evening of"
DATE = "April 1st"
TIME = "at 7 o'clock"

docInvi = docx.Document("Doc1.docx")

for guestname in guestsTxt: #iterating over every line in the file guestsTxt
    #guestname="Mamatha\n"
    guestname=guestname.strip('\n')
    #guestname="Mamatha"
    docInvi.add_paragraph(INTRO,"Mamatha")
    docInvi.add_paragraph(guestname,"NameStyle")
    docInvi.add_paragraph(ADDRESS,"Mamatha")
    docInvi.add_paragraph(DATE,"Mamatha")
    obj=docInvi.add_paragraph(TIME,"Mamatha")
    obj.runs[0].add_break(docx.enum.text.WD_BREAK.PAGE) #page break

docInvi.save('Invitations.docx')
guestsTxt.close()
print('Done')
```

```
Done
```

In [ ]:

Working with CSV Files

In [ ]:
```
#Program to read the contents of a csv file

import csv
#open the csv file and create a file object
exampleFile = open('example.csv')
#create a reader object
exampleReader = csv.reader(exampleFile)
#passing  the reader object to list() that creates list of lists
exampleData = list(exampleReader)
print('The contents of csv file are:')
print(exampleData)
exampleFile.close()
```

In [ ]:
```
exampleData[3][2]  #first row second column
```

In [ ]:
```
#Program to read the contents of a csv file
import csv
exampleFile = open('example.csv')
exampleReader = csv.reader(exampleFile)
print("Total number of lines=",exampleReader.line_num)
for row in exampleReader:
    print('Row #',exampleReader.line_num,"::",row)
print("Total number of lines=",exampleReader.line_num)
exampleFile.close()
```

In [33]:
```
#To create a new csv file using writer objects
import csv
outputFile = open('output.csv', 'w',newline='')
#Creates writer object
outputWriter = csv.writer(outputFile)
#Writing 3 rows into output.csv
outputWriter.writerow(['spam', 'eggs', 'bacon', 'ham'])
outputWriter.writerow(['Hello', 'world!', 'eggs', 'bacon', 'ham'])
outputWriter.writerow([1, 2, 3.141592, 4])
outputFile.close()
```

In [35]:
```
#To create a new csv file using writer objects with delimiter
#and lineterminator keyword arguments
import csv
csvFile = open('newexample.csv', 'w', newline='')
csvWriter = csv.writer(csvFile, delimiter='\t', lineterminator='\n\n')
csvWriter.writerow(['apples', 'oranges', 'grapes'])
csvWriter.writerow(['eggs', 'bacon', 'ham'])
csvWriter.writerow(['spam', 'spam', 'spam', 'spam', 'spam', 'spam'])
csvFile.close()
```

In [38]:
```
import csv, os

#The os.makedirs() call will create a headerRemoved folder in current
#working directory where all the headless CSV files will be written.

os.makedirs('headerRemoved', exist_ok=True)

# Loop through every file in the current working directory.
for csvFilename in os.listdir('.'):
    if not csvFilename.endswith('.csv'):
        continue # skip non-csv files
    print('Removing header from ' , csvFilename , '...')

    # Read the CSV file in csvRows(skipping first row).
    csvRows = []
    csvFileObj = open(csvFilename)
    readerObj = csv.reader(csvFileObj)
    for row in readerObj: #iterate over the rows/lines of readerObj
        if readerObj.line_num == 1:
            continue # skip first row
        csvRows.append(row)  #append line 2 to the last
    csvFileObj.close()

    #csvRows contains all rows but the first row, the list needs to be
    #written out to a CSV file in the headerRemoved folder.
    # Write out the CSV file.
    print(os.path.join('headerRemoved',csvFilename))
    csvFileObj = open(os.path.join('headerRemoved',csvFilename),'w',newline='')
    csvWriter = csv.writer(csvFileObj)
    for row in csvRows:
        csvWriter.writerow(row)
    csvFileObj.close()
```

```
Removing header from  example.csv ...
headerRemoved\example.csv
Removing header from  fbdoc.csv ...
headerRemoved\fbdoc.csv
Removing header from  newexample.csv ...
headerRemoved\newexample.csv
Removing header from  output.csv ...
headerRemoved\output.csv
```

In [ ]:
```

```

Webbrowser Module

In [21]:
```
import webbrowser

webbrowser.open('https://www.amazon.in/')
```

Out[21]: True

Request Module

In [2]:
```
import requests
res = requests.get('http://www.gutenberg.org/cache/epub/1112/pg1112.txt')
```

In [3]:
```
type(res)
```

Out[3]: requests.models.Response

In [4]:
```
len(res.text)
```

Out[4]: 179380

In [5]:
```
print(res.text[:250])
```

```
The Project Gutenberg EBook of Romeo and Juliet, by William Shakespeare


**********************************************************************
THIS EBOOK WAS ONE OF PROJECT GUTENBERG'S EARLY FILES PRODUCED AT A
TIME WHEN PROOFING METHODS AND TOO
```

Checking for Errors

In [7]:
```
res = requests.get('http://www.gutenberg.org/cache/epub/1112/pg1112.txt')
res.raise_for_status()
```

In [8]:
```
import requests
res = requests.get('http://inventwithpython.com/page_that_does_not_exist')
try:
    res.raise_for_status()
except Exception as exc:
    print('There was a problem: %s' % (exc))
```

```
There was a problem: 404 Client Error: Not Found for url: http://inventwith
python.com/page_that_does_not_exist (http://inventwithpython.com/page_that_
does_not_exist)
```

Saving Downloaded Files to the Hard Drive

In [1]:
```
import requests
res = requests.get('http://www.gutenberg.org/cache/epub/1112/pg1112.txt')
res.raise_for_status()
playFile = open('RomeoAndJuliet.txt', 'wb')
for chunk in res.iter_content(100000):
    playFile.write(chunk)
```

In [2]:
```
playFile.close()
```

Creating Beautiful Soup Object from HTML

In [9]:
```
import requests, bs4
res = requests.get('http://nostarch.com')
res.raise_for_status()
noStarchSoup = bs4.BeautifulSoup(res.text)
type(noStarchSoup)
```

Out[9]:
```
bs4.BeautifulSoup
```

In [10]:
```
exampleFile = open('S.html', encoding="utf8")
exampleSoup = bs4.BeautifulSoup(exampleFile)
type(exampleSoup)
```

Out[10]:
```
bs4.BeautifulSoup
```

Finding an Element with the select() method

In [11]:
```
import bs4
exampleFile = open('Sa.html', encoding="utf8")
exampleSoup = bs4.BeautifulSoup(exampleFile.read())
elems = exampleSoup.select('div')
```

In [12]:
```
type(elems)   #List
```

Out[12]:
```
bs4.element.ResultSet
```

In [13]:
```
len(elems)
```

Out[13]:
```
227
```

In [14]:
```
type(elems[0])
```

Out[14]:
```
bs4.element.Tag
```

In [15]:
```
elems[0].getText()
```

Out[15]:
```
'\nSkip to main content\n'
```

In [16]:
```
str(elems[0])
```

Out[16]:
```
'<div id="skip-link">\n<a class="element-invisible element-focusable" href
="#main-content">Skip to main content</a>\n</div>'
```

In [17]:
```
elems[0].attrs
```

Out[17]:
```
{'id': 'skip-link'}
```

Getting Data from an Element's Attribute

In [18]:
```
import bs4
soup = bs4.BeautifulSoup(open('Sa.html', encoding="utf8"))
spanElem = soup.select('span')[1]
str(spanElem)
```

Out[18]:
```
'<span class="icon-bar"></span>'
```

In [19]:
```
spanElem.get('class')
```

Out[19]:
```
['icon-bar']
```

In [20]:
```
spanElem.attrs
```

Out[20]:
```
{'class': ['icon-bar']}
```

Selenium

In [22]:
```
from selenium import webdriver
chromedriver="D:\Software\chromedriver"
browser = webdriver.Chrome(chromedriver)
type(browser)
```

Out[22]:
```
selenium.webdriver.chrome.webdriver.WebDriver
```

In [23]:
```
from selenium import webdriver
chromedriver="D:\Software\chromedriver"
browser = webdriver.Chrome(chromedriver)
type(browser)
browser.get("http://inventwithpython.com/")
try:
    elem = browser.find_element_by_class_name('row')
    print('Found <%s> element with that class name!' % (elem.tag_name))
except:
    print('Was not able to find an element with that name.')
```

```
Found <div> element with that class name!
```

Clicking on the link

In [24]:
```
from selenium import webdriver
import time
chromedriver="D:\Software\chromedriver"
browser = webdriver.Chrome(chromedriver)
type(browser)
browser.get("http://inventwithpython.com/")
linkElem = browser.find_element_by_link_text('Read for Free')
time.sleep(10)
linkElem.click()
```

Filling Out & Submitting Forms

In [25]:
```
from selenium import webdriver
import time
chromedriver="D:\Software\chromedriver"
driver = webdriver.Chrome(chromedriver)
driver.get('https://demoqa.com/automation-practice-form/')
time.sleep(10)
FN = driver.find_element_by_id('firstName').send_keys("Santosh")
time.sleep(10)
SN=driver.find_element_by_id('lastName').send_keys("Reddy P")
time.sleep(10)
Email=driver.find_element_by_id('userEmail').send_keys("RamMurthyNaidu@gmail.
time.sleep(10)
Submit=driver.find_element_by_id('submit').submit()
```

Sending Special Keys

In [27]:
```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
chromedriver="D:\Software\chromedriver"
driver = webdriver.Chrome(chromedriver)
driver.get('http://nostarch.com')
htmlElem = driver.find_element_by_tag_name('html')
htmlElem.send_keys(Keys.END)
time.sleep(10)
htmlElem.send_keys(Keys.HOME)
driver.refresh()
time.sleep(10)
driver.quit()
```

In [ ]:
```

```

Reading Excel Files

In [3]:
```
import openpyxl
#creating Workbook Object for reading excel file
wb = openpyxl.load_workbook('example.xlsx')
```

In [4]:
```
type(wb)
```

Out[4]:
```
openpyxl.workbook.workbook.Workbook
```

In [5]:
```
#list of names of all the sheets in the workbook
wb.get_sheet_names()
```

Out[5]:
```
['Sheet1', 'Sheet2', 'Sheet3']
```

In [6]:
```
#Accesing sheet3, create a Worksheet object
sheet = wb.get_sheet_by_name('Sheet3')
```

In [7]:
```
type(sheet)
```

Out[7]:
```
openpyxl.worksheet.worksheet.Worksheet
```

In [ ]:
```
sheet.title
```

In [9]:
```
c=sheet.cell(row=1, column=2)
print(c)
print(c.value)
```

```
<Cell Sheet3.B1>
Apples
```

In [11]:
```
sheet = wb.get_sheet_by_name('Sheet3')
c=sheet['A1']
type(c)
print(c)
```

```
<Cell Sheet3.A1>
```

In [12]:
```
c.value
```

Out[12]:
```
datetime.datetime(2014, 4, 5, 13, 34)
```

In [13]:
```
c.row
```

Out[13]:
```
1
```

In [14]:
```
c.column
```

Out[14]:
```
'A'
```

In [15]:
```
c.coordinate
```

Out[15]:
```
'A1'
```

In [16]:
```
for i in range(1, 8, 2):
    print(i, sheet.cell(row=i, column=2).value)
```

```
1 Apples
3 Pears
5 Apples
7 Strawberries
```

In [18]:
```
import openpyxl
wb = openpyxl.load_workbook('example.xlsx')
sheet = wb.get_sheet_by_name('Sheet1')
print(sheet.get_highest_row())    #7
print(sheet.get_highest_column()) #3
```

```
7
4
```

In [19]:
```
import openpyxl
wb = openpyxl.load_workbook('example.xlsx')
sheet = wb.get_sheet_by_name('Sheet1')
print(sheet.columns) # tuple of 4 tuples consisting of column wise cells
```

```
((<Cell Sheet1.A1>, <Cell Sheet1.A2>, <Cell Sheet1.A3>, <Cell Sheet1.A4>, <Cell Sheet1.A5
l Sheet1.A6>, <Cell Sheet1.A7>), (<Cell Sheet1.B1>, <Cell Sheet1.B2>, <Cell Sheet1.B3>, <
eet1.B4>, <Cell Sheet1.B5>, <Cell Sheet1.B6>, <Cell Sheet1.B7>), (<Cell Sheet1.C1>, <Cell
1.C2>, <Cell Sheet1.C3>, <Cell Sheet1.C4>, <Cell Sheet1.C5>, <Cell Sheet1.C6>, <Cell Shee
>), (<Cell Sheet1.D1>, <Cell Sheet1.D2>, <Cell Sheet1.D3>, <Cell Sheet1.D4>, <Cell Sheet1
Cell Sheet1.D6>, <Cell Sheet1.D7>))
```

In [21]:
```
import openpyxl
wb = openpyxl.load_workbook('example.xlsx')
sheet = wb['Sheet1']
for cellObj in sheet.columns[1]: #Refers to second column
    print(cellObj.value)
```

```
Apples
Cherries
Pears
Oranges
Apples
Bananas
Strawberries
```

In [23]:
```
import openpyxl
wb = openpyxl.load_workbook('example.xlsx')
sheet = wb.get_sheet_by_name('Sheet1')
#print(sheet.rows)  #tuple of 7 tuples consisting elements of row wise
for cellObj in sheet.rows[1]: #refers to second row
    print(cellObj.value)
```

```
2014-04-05 03:41:00
Cherries
85
None
```

In [30]:
```
import openpyxl
wb = openpyxl.load_workbook('example.xlsx')
sheet = wb.get_sheet_by_name('Sheet1')
#print(sheet['A1':'C3'])
for rowOfCellObjects in sheet['B3':'C6']:#((A1,B1,C1),(A2,B2,C2),(A3,B3,C3))
    for cellObj in rowOfCellObjects:
        print(cellObj.coordinate, cellObj.value)
    print('--- END OF ROW ---')
```

```
B3 Pears
C3 14
--- END OF ROW ---
B4 Oranges
C4 52
--- END OF ROW ---
B5 Apples
C5 152
--- END OF ROW ---
B6 Bananas
C6 23
--- END OF ROW ---
```

Writing Excel Files

In [31]:
```
import openpyxl
#new workbook object contaning only one sheet with 'Sheet'
wb = openpyxl.Workbook()
print(wb.get_sheet_names())
```

```
['Sheet']
```

In [32]:
```
sheet = wb.get_active_sheet()
sheet.title = 'V ADP'
print(wb.get_sheet_names())
```

```
['V ADP']
```

In [33]:
```
import openpyxl
wb = openpyxl.Workbook()
print("The sheets are:")
print(wb.get_sheet_names())
sheet = wb.get_active_sheet()
sheet.title = 'V ADP'
print("The sheets are:")
print(wb.get_sheet_names())
wb.save('example_copy.xlsx')
```

```
The sheets are:
['Sheet']
The sheets are:
['V ADP']
```

In [36]:
```
import openpyxl
wb = openpyxl.Workbook()
print(wb.get_sheet_names())
wb.create_sheet()
wb.create_sheet(index=0, title='First Sheet')
wb.create_sheet(index=2, title='Middle Sheet')
print(wb.get_sheet_names())
wb.remove_sheet(wb.get_sheet_by_name('Middle Sheet'))
wb.remove_sheet(wb.get_sheet_by_name('Sheet1'))
print(wb.get_sheet_names())
wb.save('example_copy.xlsx')
```

```
['Sheet']
['First Sheet', 'Sheet', 'Middle Sheet', 'Sheet1']
['First Sheet', 'Sheet']
```

In [ ]:
```
wb.remove_sheet(wb.get_sheet_by_name('Middle Sheet'))
wb.remove_sheet(wb.get_sheet_by_name('Sheet1'))
print(wb.get_sheet_names())
wb.save('example_copy.xlsx')
```

In [37]:
```
import openpyxl
wb = openpyxl.Workbook()
sheet = wb.get_sheet_by_name('Sheet')
sheet['A1'] = 'Hello world!'
sheet['B1'] = 435
wb.save('example_copy.xlsx')
```

Adding Font styles to Cells

In [13]:
```
import openpyxl
from openpyxl.styles import Font, Style

wb = openpyxl.Workbook() #workbook object
sheet = wb.get_sheet_by_name('Sheet') #worksheet object

#create Font Object
italic24Font = Font(size=24, italic=True)

#create Style object using Font object
styleObj = Style(font=italic24Font)

sheet['A1'].style=styleObj
sheet['A1'] = 'Hello world!'
wb.save('styled.xlsx')
```

In [14]:
```
import openpyxl
from openpyxl.styles import Font, Style

wb = openpyxl.Workbook()
sheet = wb.get_sheet_by_name('Sheet')

fontObj1 = Font(name='Times New Roman', bold=True)
styleObj1 = Style(font=fontObj1)
sheet['A1'].style=styleObj1
sheet['A1'] = 'Bold Times New Roman'

fontObj2 = Font(size=24, italic=True)
styleObj2 = Style(font=fontObj2)
sheet['B3'].style=styleObj2
sheet['B3'] = '24 pt Italic'

wb.save('styles.xlsx')
```

Adding Formulas

In [33]:
```
import openpyxl
wb = openpyxl.Workbook()
sheet = wb.get_active_sheet()
sheet['A1'] = 200
sheet['A2'] = 300
sheet['A3'] = '=SUM(A1:A2)'
wb.save('writeFormula.xlsx')
```

In [34]:
```
sheet['A1'].value
```

Out[34]: 200

In [35]:
```
sheet['A2'].value
```

Out[35]: 300

In [36]:
```
sheet['A3'].value
```

Out[36]: '=SUM(A1:A2)'

In [44]:
```
wbFormulas = openpyxl.load_workbook('writeFormula.xlsx')
wbDataOnly=openpyxl.load_workbook('writeFormula.xlsx', data_only=True)
sheet = wbDataOnly.get_active_sheet()
print(sheet['A3'].value)  #500
```

500

Merging and Unmerge Cells

In [38]:
```
import openpyxl
wb = openpyxl.Workbook()
sheet = wb.get_active_sheet()
sheet.merge_cells('A1:D3')
sheet['A1'] = 'Twelve cells merged together.'
sheet.merge_cells('C5:D5')
sheet['C5'] = 'Two merged cells.'
wb.save('merged.xlsx')
```

In [39]:
```
import openpyxl
wb = openpyxl.load_workbook('merged.xlsx')
sheet = wb.get_active_sheet()
sheet.unmerge_cells('A1:D3')
sheet.unmerge_cells('C5:D5')
wb.save('merged.xlsx')
```

Adjusting row and column

In [1]:
```
import openpyxl
wb = openpyxl.Workbook()
sheet = wb.get_active_sheet()
sheet['A1'] = 'Tall row'
sheet['B2'] = 'Wide column'
sheet.row_dimensions[1].height = 70
sheet.column_dimensions['B'].width = 20
wb.save('dimensions.xlsx')
```

Freezing Panes

In [2]:
```
import openpyxl
wb = openpyxl.load_workbook('produceSales.xlsx')
sheet = wb.get_active_sheet()
sheet.freeze_panes = 'A2'
wb.save('freezeExample.xlsx')
```

Inserting Charts in the Excel Sheet

In [5]:

```
import openpyxl
wb = openpyxl.Workbook()
sheet = wb.get_active_sheet()

for i in range(1, 11): # create some data in column A
    sheet['A' + str(i)] = i

#create Reference object
refObj = openpyxl.charts.Reference(sheet, (1, 1), (10, 1))

#create Series object
seriesObj = openpyxl.charts.Series(refObj, title='First series')

#create Chart object
chartObj = openpyxl.charts.BarChart()

#Append the Series object to the Chart object.
chartObj.append(seriesObj)

#set the drawing.top, drawing.left, drawing.width, and
#drawing.height variables of the Chart object.

chartObj.drawing.top = 50 # set the position
chartObj.drawing.left = 100
chartObj.drawing.width = 300 # set the size
chartObj.drawing.height = 500

#Add the Chart object to the Worksheet object.
sheet.add_chart(chartObj)

#save the excel file
wb.save('sampleChart.xlsx')
```

Reading JSON with the loads() Function

In [ ]:
```
import json
stringOfJsonData = '{"name": "Zophie", "isCat": true, "miceCaught": 0, "felineIQ": null}'
jsonDataAsPythonValue = json.loads(stringOfJsonData)
print(jsonDataAsPythonValue)
print(type(jsonDataAsPythonValue))
```

In [10]:
```
import json
json_string = """
{
    "researcher": {
        "name": "Ford Prefect",
        "species": "Betelgeusian",
        "relatives": [
            {
                "name": "Zaphod Beeblebrox",
                "species": "Betelgeusian"
            }
        ]
    }
}
"""
data = json.loads(json_string)
#print(data)
#print(type(data))
print(type(data["researcher"]['relatives']))
```

```
<class 'list'>
```

Writing JSON with the dumps() Function

In [12]:
```
import json
pythonValue = {'isCat': True, 'miceCaught': 0, 'name': 'Zophie','felineIQ': None}
stringOfJsonData = json.dumps(pythonValue)
print(type(stringOfJsonData))
```

```
<class 'str'>
```

Project: Fetching Current Weather Data

In [ ]:

```
#import statements
import json, requests, sys
import pprint


# Compute location from command line arguments.
if len(sys.argv) < 2:
    print('Usage: quickWeather.py location')
    sys.exit()
location = ' '.join(sys.argv[1:])
#sys.argv=['tamil','nadu']
#location='tamil nadu'

# TODO: Download the JSON data from OpenWeatherMap.org's API.
# Enter your API key here
BASE_URL = "https://api.openweathermap.org/data/2.5/weather?"
API_KEY = "7f563da3a0374030e48f80d27d0240f6"
# upadting the URL
URL = BASE_URL + "q=" + location + "&appid=" + API_KEY

#create response object
response = requests.get(URL)
response.raise_for_status()


# TODO: Load JSON data into a Python variable.
data = json.loads(response.text)
print(data)


# getting the main dict block
#main = data['main']
# getting temperature
temperature = data['main']['temp']
# getting the humidity
humidity = data['main']['humidity']
# getting the pressure
pressure = data['main']['pressure']
# weather report
report = data['weather']


print("City:",location)
print("Temperature:",temperature)
print("Humidity:" ,humidity)
print("Pressure:",pressure)
print("Weather Report:", report[0]['description'])
```