Word analysis project

The project will use some modules which have not been covered in the class. Please google and try them out. Here is the requirement:

1). Read the PDF file, SampleCh7.pdf, which is also in the folder, and extract all the text from the pdf file. You will need to use the PyPDF2 module. The PyPDF2 module is not in the Anaconda's default installation, and you need to use the following command to install the module:

```
conda install -c conda-forge pypdf2
```
(execute the command in the Anaconda command window)

Or you can install in the google colab environment:
```
!pip install PyPDF2
```

2). Build a WordAnalysis Class to do the analysis work. The class should include the following methods (please define the attribute yourself):

```
gordAnalysis:
        def __init__(self):
        def __str__(self):
        def AddRemovedWords(self, word):
        def DelRemovedWords(self, word):
        def ExtractWords(self, text):
        def Show(self, freq):
```

For **ExtractWords(self, text)**:
It should:
1). Strip off all punctuations, numbers, and symbols, only words are kept, and try to remove non-words as much as you can, for example, single letter words.
2). Convert the plurals to singular words, and convert the capitalized words to lowercase words. You can use the pattern module to convert the plurals to singular words. Again, you need to install the pattern module as follows:

```
#conda install -c conda-forge pattern
```
Conda install packages have compatibility issues. So I switch to
```
Pip install pattern
```

3). Count the frequency of the words, and save them in a dictionary (internal attribute).

For **AddRemovedWords(self, word),**

it will add a word to be filtered out, so that the dictionary will not save the frequency of the world, for example, ‘the’.

For **DelRemovedWords(self, word):**
It can remove a word from the filter list, and add the word back to the dictionary for frequency count.

For **Show(self, freq):**
It will show the words with a frequency higher than the specified frequency, freq, and show them in descending order.

in **__str__(self)**:
It prints out all the words in the dictionary. It should print out the words and frequency 4 pairs in a row.

In **__init__(self)**
It should construct all the internal attributes of the class.

Here are the sample testing code:

```
# you can find find the pdf file with complete code in below
pdfFileObj = open('Samplech7.pdf', 'rb')
# pdf reader object
pdfReader = PyPDF2.PdfFileReader(pdfFileObj)
# number of pages in pdf
pages=pdfReader.numPages
#remove the words that are not meaningful. You can add back any word. Please test yourself.
wordAnalysis = WordAnalysis()
wordAnalysis.AddRemovedWords("and")
wordAnalysis.AddRemovedWords("a")
wordAnalysis.AddRemovedWords("the")
# analys 4 pages. max pages is pages calculated above
for i in range(4):
   pageObj = pdfReader.getPage(i)
   # extracting the words from page.into database
   wordAnalysis.ExtractWords(pageObj.extractText())
# show words with frequency > 10
wordAnalysis.Show(10)
#print out all the words in the dictionary
print (wordAnalysis)
#close the file.
pdfFileObj.close()
```

The Output should as follows (the output should align nicely as shown in the output file, project-output.txt, also in the folder):

```
word --     number      -- frequency: 32
word --       is        -- frequency: 25
word --       to        -- frequency: 24
word --      phone      -- frequency: 24
word --       you       -- frequency: 22
word --       in        -- frequency: 22
word --     regular     -- frequency: 19
word --   expression    -- frequency: 19
word --      text       -- frequency: 19
word --       of        -- frequency: 19
word --       for       -- frequency: 18
word --     pattern     -- frequency: 17
word --  isphonenumber  -- frequency: 15
word --       if        -- frequency: 14
word --     string      -- frequency: 13
word --      that       -- frequency: 12
word --      chunk      -- frequency: 12
word --      with       -- frequency: 11
word --     return      -- frequency: 11

pattern        17 matching       3  with           11 regular        19
expression     19 you            22 may            2  be             6
familiar       1  searching      1  for            18 text           19
by             4  pressing       1  ctrl           1  entering       1
word           4  re             4  looking        1  go             2
one            2  step           3  further        1  they           1
allow          1  to             24 specify        1  of             19
search         2  not            8  know           5  busines        1
exact          1  phone          24 number         31 but            9
if             14 live           1  in             22 united         1
state          1  or             4  canada         1  it             10
will           5  three          9  digit          5  followed       1
hyphen         8  then           6  fmy            5  more           9
optionally     1  area           3  code           10 at             9
start          2  thi            9  is             25 how            3
as             4  human          1  when           4  see            4
we             4  also           4  recognize      1  all            4
sort           1  other          4  every          1  day            2
email          2  address        2  have           6  symbol         1
middle         1  social         2  security       1  nine           1
```

two	1	website	1	url	1	often	1
period	1	forward	1	slash	1	news	1
headline	1	use	4	title	1	case	1
medium	1	hashtag	1	begin	1	contain	1
no	1	space	1	chapter	4	are	3
helpful	1	few	1	non	1	programmer	2
about	2	them	2	even	3	though	1
most	1	modern	1	editor	2	processor	1
such	2	microsoft	1	openof	1	ce	1
nd	9	replace	2	feature	2	that	12
can	6	based	1	on	8	huge	1
time	2	saver	1	just	1	software	1
user	1	fact	1	tech	1	writer	1
cory	2	doctorow	2	argue	1	should	2
teaching	1	before	1	programming	1	knowing	1
mean	1	difference	1	between	1	solving	2
problem	2	nerd	1	forget	1	solve	1
cou	1	ple	1	keystroke	1	take	2
person	1	tediou	1	error	1	prone	1
work	2	slog	1	through	4	ll	4
writing	1	program	9	out	1	using	2
make	2	much	2	les	3	bloated	1
show	1	basic	1	move	1	some	1
powerful	1	string	13	substitution	1	creating	1
ymy	1	own	1	character	10	class	1
finally	1	end	1	write	2	automatically	1
extract	1	from	3	block	1	finding	2
without	1	say	1	want	1	an	4
american	2	here	1	example	5	let	1
function	8	named	1	isphonenumber	15	check	7
whether	3	match	5	returning	1	either	1
true	3	false	9	open	1	new	2
le	2	tab	1	enter	1	following	3
save	1	py	2	def	1	len	2
return	11	range	4	isdecimal	3	fihere	1
what	3	ict	2	really	1	teach	2
kid	2	do	3	guardian	1	december	1
http	1	www	1	theguardian	1	com	1
technology	1	dec	1	trueprint	1	print	8
moshi	10	run	4	output	2	look	2
like	5	truei	1	falsethe	1	ha	1
several	1	valid	1	any	4	these	3
fail	3	first	1	exactly	1	rst	4

consist	1	only	2	numeric	2	rest	1
follow	1	must	1	after	2	another	2
nally	1	execution	1	manage	1	get	1
past	1	calling	2	argument	1	test	1
because	1	long	2	wanted	1	within	1
larger	1	would	5	add	2	last	1
call	4	message	9	me	7	tomorrow	1
my	1	office	1	chunk	12	found	3
done	4	each	3	iteration	3	loop	4
assigned	3	variable	1	itera	1	tion	1
next	1	value	1	so	3	pas	1
continue	1	eventually	1	entire	1	testing	1
piece	1	printing	1	sati	1	es	1
once	1	going	1	while	1	short	2
could	2	million	1	still	1	than	2
second	2	similar	1	quicker	1	previou	2
numberœ	1	nding	1	lot	1	something	1
limited	1	line	1	formatted	1	num	1
ber	1	had	1	extension	1	validate	1
yet	1	addi	1	tional	1	there	1
easier	1	way	1	called	1	regex	5
description	1	pat	1	tern	1	stand	1
characteršthat	1	single	1	numeral	1	used	1
python	1	same	1	did	1	sophisticated	1
adding	1	brace	1	saying	1	fimatch	1
slightly	1	shorter	1	correct	1	format	1