Second program

__/__/__

PDF:- Portable Document format. Generally PDF file contains text, numbers, images, fonts, tables, special charactes

Pdfminer:-

→ It is used to extracting the text data from Pdf file.

→ It allow access to text position & font information.

→ It suppose various output formats like plain Text, HTML, XML

Pdfquery:-

→ It is built on top of the pdf miner.

→ It is designed for extracting structured data from Pdf file

→ it is used to extract the tabular data.

Pypdf2:- → Adding or removing pages.

→ It supports merging, splitting, rotation of PDF pages.

→ It allows text extraction & text search within PDF.

Pymupdf:-

→ It is the high performance PDF library.

→ It supports text extraction & image extraction etc.

PDF parser:-

→ It is a S/w component.

→ It is used to extract information & content from PDF

Need for parsing:-

To avoid errors and inaccurate data extraction we use parsing.

Text extraction:-

Primary task of PDF parser is to extract text from PDF document

Font Handling:-
PDF file uses different fonts for text. PDF parser is responsible for handling font information.

Image extraction:-
PDF parser is responsible for images, graphs extraction. Extract images in various format & Resolution

Metadata extraction:-
ex:- title, author, creation Date.

Content Search:-
Search Specific word with in the file.

Document Manipulation:-
Adding, or remaving pages, merging, splitting etc.

Conversion:-
Converts PDF to plain text, HTML, XML.

Program:-

```
from nltk.tokenize import RegexpTokenizer
from pdfminer.high_level import extract_text
from nltk.probability import FreqDist.
# extract the text from PDF file
text = extract_text('/mahi/DW/2010.00462.pdf')
# create an instance of tokenizer using NLTK
                              RegexpTokenzier
 tokenizer = RegexpTokenizer('\w+')
# Tokenize the text read from PDF
  tokens = tokenizer.tokenize(text)
```

\w+ pattern only word characters are included in ~~pattern~~ Tokens (like letters, digits & underscores). removes non-word charactery (,.!)

```
# find frequency Distribution
        freqdist = FreqDist(tokens)
# find words whose length is greater than 5
  and frequency greater than 20
long_frequent_words = [words for words in tokens
                       if len(words)>5  and
                       freqdist[words]>20]
long_frequent_words
FreqDist(long_frequent_words).plot()
```

Google Calab :-

```
        from google.Calab import files
           files.upload()
```