```
#This script is intended to extract data from a PDF.
#The following commands are needed to install PyPDF2 and fitz if they haven't been previously
installed.
#Each command only needs to be run once in Anaconda before being commented out.
#**if the packages are already installed, the following message (or something similar) will be
#displayed:
#"Requirement already satisfied: PyPDF2 in /opt/anaconda3/lib/python3.8/site-packages
#(1.26.0)”**
#!pip install PyPDF2
#!pip install PDFMiner
#!pip3 install PyMuPDF

import PyPDF2 as pypdf
from PyPDF2 import PdfFileReader, PdfFileWriter
from pathlib import Path
import numpy as np
import os
import fitz #used with PyMuPDF for extracting PDF images

#Naming the file from which data will be read
#**Note: slash amount and direction may need to be changed depending on the os**
fileName=“//Users//Ren//Documents//textfile.txt"

#Select the first (and possibly only) column in the text file, skipping any header rows
#and confirming the dtype (data type) as str; the default is float.
#**Be sure to adjust the rows to skip for different text files.**
fileColumn=np.loadtxt(fileName,dtype=str,skiprows=3,usecols=[0])

#Just in case the title is in the text file, or there are any other relevant headers:
#**May be changed into a for loop if more than one title is needed**
titleColumn=np.loadtxt(fileName,dtype=str,usecols=[0])
header=titleColumn[0]
#print(header,"header")

#Display the number of individual objects in the column
totalItems=str(len(fileColumn))
#print("There are "+totalParts+" elements in this column of the file.")

#Putting each string starting with a certain character into a different list, just in case
allParts=[] #Initializing the list for all items
fotherList=[] #Initializing the list for only the other numbers

for i in range(len(fileColumn)):
    numstr=fileColumn[i] #The item in the column currently being checked
    allItems.append(numstr)

    #Check if the first character in the number string is "S" and break if it isn't
    charCheck='S'
    for j, v in enumerate(numstr):
        if v == charCheck:
            otherList.append(numstr)
            break

totalItems=str(len(allItems))
```

```
someItems=str(len(otherList))
#print("There are "+totalItems+" elements in the list containing all items.”)
#print("There are "+someItems+" elements in the list with only items starting with 'S'.")

#name the file path on which all of the relevant folders will be located
#filePath=input('Enter a location for the file path:')
partialFP='//Users//Ren//Desktop//'

#open the pdf or docx file to be parsed; ensure the entire path to the file is included; 'rb'
#indicates 'read bytes'
#**Either a pdf or docx file can be opened for reading; just
#ensure that the correct extension is referenced in the string.
searchedFileStr=partialFP+'Sample.pdf'
searchedFile=open(searchedFileStr, 'rb') #may be changed to input if necessary

#name the file for output messages and its location
output=partialFP+'Process Messages.txt'
stdOutputFile=open(output,"w+")

#number of pages in the pdf file
docLength=pdfReader.numPages
docLengthStr=str(docLength) #Converting the number to a string
#print("This document is",docLengthStr,"pages long.")

#initialize a list containing names of the items for which files have been searched
#and copied when necessary
searchedList=[]

#initialize a list containing names of newly-made folders
newFolderList=[]

for i in range(len(allItems))
   searchPhrase=allItems[i] #Each search phrase is an item from the list

   #Make a folder for each item if that doesn't already exist
   completePath=partialFP+searchPhrase
   folderPath=Path(completePath)
   subDirPath1=completePath+"//SubFolder1"
   fullSubDirPath1=Path(subDirPath1)
   subDirPath2=completePath+"//SubFolder2"
   fullSubDirPath2=Path(subDirPath2)
   if not os.path.exists(folderPath):
      os.mkdir(completePath)
      stdOutputFile.write("A folder named '"+searchPhrase+"' was created.\n")
      newFolderList.append(searchPhrase)

      #Make subfolders
      if not os.path.exists(subDirPath1):
         os.mkdir(subDirPath1)
                     stdOutputFile.write("A subfolder named 'SubFolder1' was created in the
\'"+searchPhrase+"\' folder.\n")
      if not os.path.exists(subDirPath2):
         os.mkdir(subDirPath2)
```

```
            stdOutputFile.write("A subfolder named 'SubFolder1' was created in the
\'"+searchPhrase+"\' folder.\n")

    #If the item hasn't already been searched for, add it to the list of
    #searched words before parsing the document
    if not searchPhrase in searchedList:
        searchedList.append(searchPhrase)

        #Create a page object for each page in the document
        for curPage in range(docLength):
            #Create a PDF Reader object to read the PDF
            pdfReader=pypdf.PdfFileReader(searchedFileStr) #Only a string may be used here
            PDFWrite=PdfFileWriter()
            docPage=pdfReader.getPage(curPage)
            pageNum=str(curPage+1)

            #Extract text from the document
            pageText=docPage.extractText()

            #Determine whether the search phrase appears on the current page of the document
            if searchPhrase in pageText:
                foundStatus="' was found"
                fileCopyStatus=" has "
                endPhrase=" with the name '"+searchPhrase+"'."

                #Export the page as a PDF to a folder when the item is found on a page
                PDFWrite.addPage(docPage)
                outFile=completePath+"//Page"+pageNum+".pdf" #this may have to include an
#offset
                with open(outFile,"wb") as output:
                    PDFWrite.write(output)

            else:
                foundStatus="' was not found"
                fileCopyStatus=" has not "
                endPhrase="."

            #this may have to include an offset due to title pages
            statMessageA=str("The search phrase \'"+searchPhrase+foundStatus+" on page
"+pageNum+' of '+docLengthStr+';\n')
            stdOutputFile.write(statMessageA)
            statMessageB=str("The page"+fileCopyStatus+"been copied to the destination
folder"+endPhrase+"\n")
            stdOutputFile.write(statMessageB)

if newFolderList==[]:
    stdOutputFile.write("No new folders have been created.")

print("Page extraction complete. The parsed file will now be closed.")

#closing the PDF file and stdout files that were previously opened
searchedFile.close()
stdOutputFile.close()
```