Name: Manukashyap.u.v
Email:manukashyap.u.v@gmail.com

Aim:

Creating the summary of the piece of text that might be fetched from a website or the text the user inputs.

Note: We are not going to create a user interface of any kind to interact with this code and everything will be done on Jupyter notebook.

Tools and Libraries used:

- Jupyter Notebook (https://jupyter.org/)
  The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include data cleaning and transformation, numerical simulation, statistical modelling, data visualization, machine learning, and much more.
- Python NLTK Library (https://www.nltk.org/)
  The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical natural language processing for English written in the Python programming language.
- Beautiful Soup (https://www.crummy.com/software/BeautifulSoup/bs4/doc/)
  Beautiful Soup is a Python library for pulling data out of HTML and XML files. It works with your favourite parser to provide idiomatic ways of navigating, searching and modifying the parse tree. It commonly saves programmers hours or days of work.
- LXML(https://lxml.de/)
  lxml is the most feature-rich and easy-to-use library for processing XML and HTML in the Python language.
- Python Regex(https://docs.python.org/3/howto/regex.html)
  In Python, a regular expression is denoted as RE (REs, regexes or regex pattern) are imported through re module. Python supports regular expression through libraries. In Python regular expression supports various things like Modifiers, Identifiers, and White space characters.
- Heapq(https://docs.python.org/2/library/heapq.html)
  This module provides an implementation of the heap queue algorithm, also known as the priority queue algorithm.

Procedure:

Installing required libraries:

Installing BeautifulSoup

```
pip install beautifulsoup4
```

```
Requirement already satisfied: beautifulsoup4 in /Users/manukashyap.u.v/Desktop/Anaconda/anac
onda3/lib/python3.7/site-packages (4.7.1)
Requirement already satisfied: soupsieve>=1.2 in /Users/manukashyap.u.v/Desktop/Anaconda/anac
onda3/lib/python3.7/site-packages (from beautifulsoup4) (1.8)
Note: you may need to restart the kernel to use updated packages.
```

Installing lxml

```
pip install lxml
```

```
Requirement already satisfied: lxml in /Users/manukashyap.u.v/Desktop/Anaconda/anaconda3/lib/
python3.7/site-packages (4.3.2)
Note: you may need to restart the kernel to use updated packages.
```

Insatalling NLTK

```
import nltk
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('wordnet')
nltk.download('treebank')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     /Users/manukashyap.u.v/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to
[nltk_data]     /Users/manukashyap.u.v/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     /Users/manukashyap.u.v/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package treebank to
[nltk_data]     /Users/manukashyap.u.v/nltk_data...
[nltk_data]   Package treebank is already up-to-date!
```

```
True
```

Input:

The user can give the input in two forms. Either has the option to give the URL or the text data.

If the user wishes to input the URL of a website. Then the URL is taken and it is parsed to get the text data into a string class.

If the user wishes to directly input the text we read it as it is and store it in a string class.

```
print("Please choose your prefered way to input text.\nPress 1 to copy paste the URL of a webs
ite.\nPress 2 to input your own text.\n")
input_choice = input()
summary_size = int(input("Enter the lines you want in the summary."))
if input_choice == "1":
    url = input("\nPlease enter the URL:\n")
    #importing the necessary libraries to parse the data from the url
    import bs4 as bs
    import urllib.request
    import re

    #scraping the data from the given url
    scraped_data = urllib.request.urlopen(url)
    #reading the data from the object returned by the urlopen
    article = scraped_data.read()
    #parsing the article using BeautifulSoup
    parsed_article = bs.BeautifulSoup(article,'lxml')
    #using findall function on the object returned by BeautifulSoup to get the paragraphs
    paragraphs = parsed_article.find_all('p')
    article_text = ""
    for p in paragraphs:
        article_text += p.text
```

Parsing the data from the given URL:

- We use *urllib.request.urlopen(URL)* to get the data from the given URL.
- The *urlopen* will return an object (scraped_data) and we read it into an article using the function *.read()*
- To scrape the data we pass the object (article) and the lxml parsed to the object BeautifulSoup.
- In website, the texts are enclosed in the tag <p>. So we use *find_all('p')* on the object returned by the BeautifulSoup(parsed_article) to get the list of all the paragraphs.
- We combine all the paragraph in the list to create an article in the string class.

Processing the input:

If the user gives the URL of the website or copy-paste the text from a popular website it might contain some references. We are now trying to remove the unnecessary things from the text file we have.

```
#importing PythonRegex(Regular Expression)
import re
# Removing Square Brackets and Extra Spaces and replacing them with white spaces
article_text = re.sub(r'\[[0-9]*\]', ' ', article_text)
article_text = re.sub(r'\s+', ' ', article_text)
```

```
# Removing special characters and digits
formatted_article_text = re.sub('[^a-zA-Z]', ' ', article_text )
formatted_article_text = re.sub(r'\s+', ' ', formatted_article_text)
```

Converting the text to sentences:

After doing all the processing we have plain text. Now we convert this into a list of sentences using the prebuilt functions in NLTK

```
import nltk
sentence_list = nltk.sent_tokenize(article_text)
```

Calculating the weighted frequency of words:

Now we find the frequency of occurrence of each word in the text. For that, we first store all the stop words of English in a variable called *stopwords.* Then we create a dictionary called *word_frequencies* with words as its key and frequency as its data. Then all we have to do is to run simply for loop to get the frequency of each word. Once we have the frequency of each word we can find the weighted frequency of the words by dividing the frequency of each word by the maximum frequency.

```
stopwords = nltk.corpus.stopwords.words('english')

word_frequencies = {}
for word in nltk.word_tokenize(formatted_article_text):
    if word not in stopwords:
        if word not in word_frequencies.keys():
            word_frequencies[word] = 1
        else:
            word_frequencies[word] += 1
```

```
maximum_frequncy = max(word_frequencies.values())

for word in word_frequencies.keys():
    word_frequencies[word] = (word_frequencies[word]/maximum_frequncy)
```

Calculating the sentence scores:

Once we find the weight of each word it is easy to calculate the score of each sentence. All we have to do is to add the weights of the words in each sentence to find the score of that sentence.

```
stopwords = nltk.corpus.stopwords.words('english')

word_frequencies = {}
for word in nltk.word_tokenize(formatted_article_text):
    if word not in stopwords:
        if word not in word_frequencies.keys():
            word_frequencies[word] = 1
        else:
            word_frequencies[word] += 1
```

```
maximum_frequncy = max(word_frequencies.values())

for word in word_frequencies.keys():
    word_frequencies[word] = (word_frequencies[word]/maximum_frequncy)
```

Getting the summary:

Once we find the scores of each sentence all we have to do is retrieve n sentences with highest scores where the value of n is given by the user. Thus we can display the summary.

```
1 import heapq
2 summary_sentences = heapq.nlargest(summary_size, sentence_scores, key=sentence_scores.get)
3
4 summary = ' '.join(summary_sentences)
5 print(summary)
```