```
#The authors names are Isa grace Guthrie and Ellen Kevin and Camryn Hurley
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import string
## We have to import string in order to use any classes and constants
# open the text file:
f = open('relativity.txt')
# read the file and create a list of words (google split function in python):
##This just opens the txt file, this doesnt have any "w" attached so the code is not ruined
words = f.read().split()
##.read goes through the opened txt file and reads it line by line. The .split() splits a sring into a list
f.close()
##this closes the file and stops the computer from reading the file
# calculate how many time each word is repeated (using dictionary):
word_counts = {}
##this creates a new dictionary named word_counts
for word in words:
    ##this loops through words in value "words" which was assigned to f.read().split()
    word_counts[word] = word_counts.get(word, 0) + 1
    ##this checks the dictionary word_counts for the for looped word. If the word is in the dictionary it will add one to the frequency. This
line basically checks the frequency of the words in the txt documennt.

# create a list of all words:
word_list = list(word_counts.keys())
##this creates a list from the arguemnt, the argument is words.counts.keys(), this then takes all of the keys in the dictionary attached to .
keys(), and prints them in a list.

# sort the list of all words based on how many times they are repeated:
def dict_val(x):
    ##this is creating a defintion of a function named "dict_val" with parameter x
    return word_counts[x]
    ##This will return values from the dictionary
word_list.sort(key = dict_val, reverse = True)
#this will take the word list we created earlier, and sort it in a certian order. As we found dict_val gives us the values from the
dictionary, and we have assigned the values to now be the keys, and the reverse=True means that the list will be in decending order.
##More clearly sort() sorts the function, the parameter key helps to sort the function becuase it words as a comparison, and reverse sorts the
list in decending order.
# print the top 10 most frequently used words:
print("List of top 10 most frequently used words: ")
##This will simply print the statement in quotes
print(word_list[ : 10])
##this will print the first 10 values in the list, the list is reverse so it will print the top 10 most used words.
###this doesn't provide much information about the text because all of these are stop words


###second attempt###
#contents = open('relativity.txt').read()
#translation_table = {ord(ch) : None  for ch in string.punctuation}
#contents = contents.translate(translation_table)
#words = contents.split()

# create a list of all words in lower case:
lowercase_words = [word.lower() for word in words]
##lower() returns a string where all the characters are lowercased, the for words in words has it loop through the entire given srings.
# open the file containing all of the stop words in English language:
f = open('stopWords.txt')
##This just opens the txt file, this doesnt have any "w" attached so the code is not ruined
# read the file create the list of all stop words in English language:
stop_words = f.read().split()
##.read goes through the opened txt file and reads it line by line. The .split() splits a sring into a list

###remove stop words###
# create the list of non stop words by filtering out the stop words:
non_stop_words = [word for word in lowercase_words if word not in stop_words]
##this creates a new list and inside that list it loops through all of the words in our previously named lowercase_words. then it checks if
that word is not in the txt file stop_words and if it is not then it puts it into the list.
# now calculate the frequency of the non stop words:
word_counts = {}
## creates a new dictionary named word_counts
for word in non_stop_words:
    ## this loops through our list non_stop_words
    word_counts[word] = word_counts.get(word, 0) + 1
    ##this checks the dictionary word_counts for the for looped word. If the word is in the dictionary it will add one to the frequency. This
line basically checks the frequency of the words in the txt documennt. This will check the non_stop_words list
    ##if the word is not in this dictorionary it will add it to the dictionary with the assigned value 1, if it is in the dictionary it will
increase the frequency by 1.

word_list = list(word_counts.keys())
##this creates a list from the arguemnt, the argument is words.counts.keys(), this then takes all of the keys in the dictionary attached to .
keys(), and prints them in a list.
# sort the words again:
word_list.sort(key = dict_val, reverse = True)
##this will take the word list we created earlier, and sort it in a certian order. As we found dict_val gives us the values from the
dictionary, and we have assigned the values to now be the keys, and the reverse=True means that the list will be in decending order.
##More clearly sort() sorts the function, the parameter key helps to sort the function becuase it words as a count, and reverse sorts the list
in decending order, from the key=dict_vals.
```

```
## print the top 10 most frequently used words:
print("\nList of top 10 most frequently used non stop words: ")
##this prints the statement in quotes in the parentheses
print(word_list[ : 10])
## it prints the first 10 words in the word list without the stop words, and from the reversed order, so the top 10 words with the highest
frequeny not incldуing stop words.
```