Internship week-3

Task-1:

Merge Sort Algorithm

It is a recursion-based algorithm that is used to sort any linear data structure in ascending or descending order.

- This algorithm is based on divide and conquer technique.
- This algorithm involves two steps:
    - Divide the array into two halves
    - Merge the divided parts to create a sorted array.

Time Complexity:

- Divide step → O (log n)
- Merge step → O (n)
  Total = O (n log n)

Space Complexity:

Space complexity is O (n) because it uses an extra temporary array while merging step.

In place:

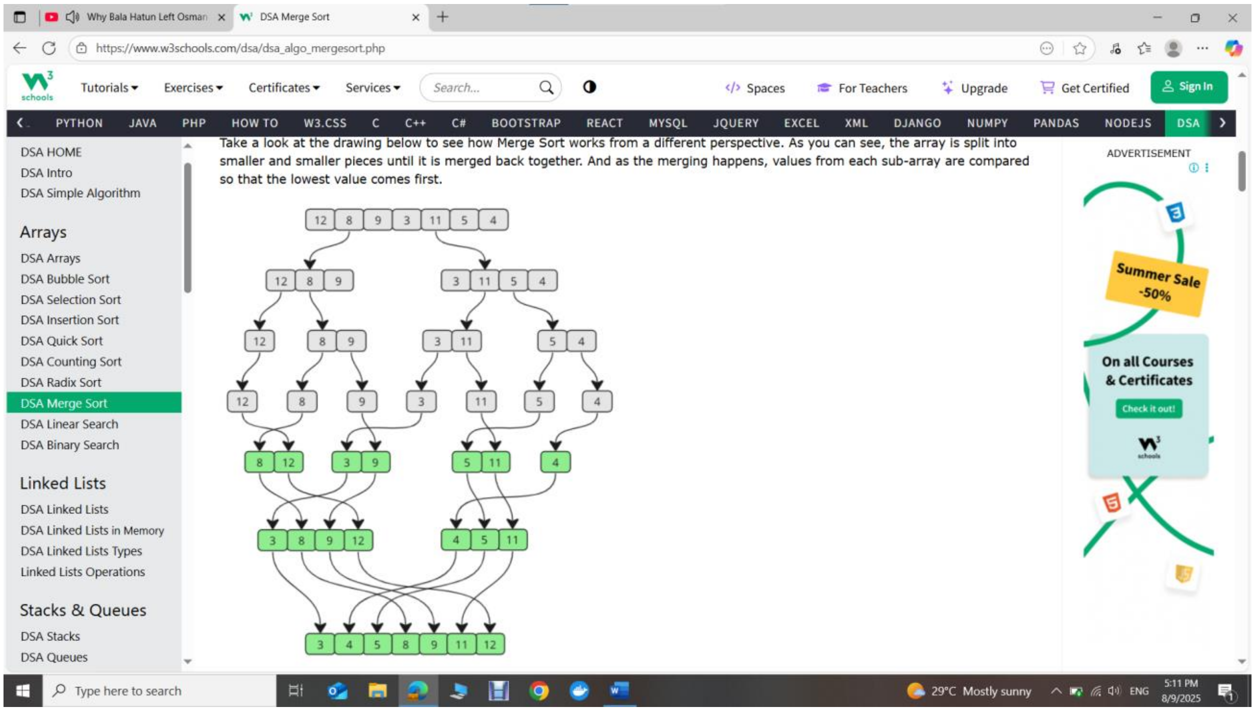
This algorithm is not in place because it uses extra space.

Stable:

Merge Sort is a stable algorithm as it preserves the original order of elements.

I learned this algorithm from following YouTube video:

Link: https://youtu.be/cQDtOBTy7_Y?si=FktjFNnFKGWXuDxL



Take a look at the drawing below to see how Merge Sort works from a different perspective. As you can see, the array is split into smaller and smaller pieces until it is merged back together. And as the merging happens, values from each sub-array are compared so that the lowest value comes first.

Task 5:

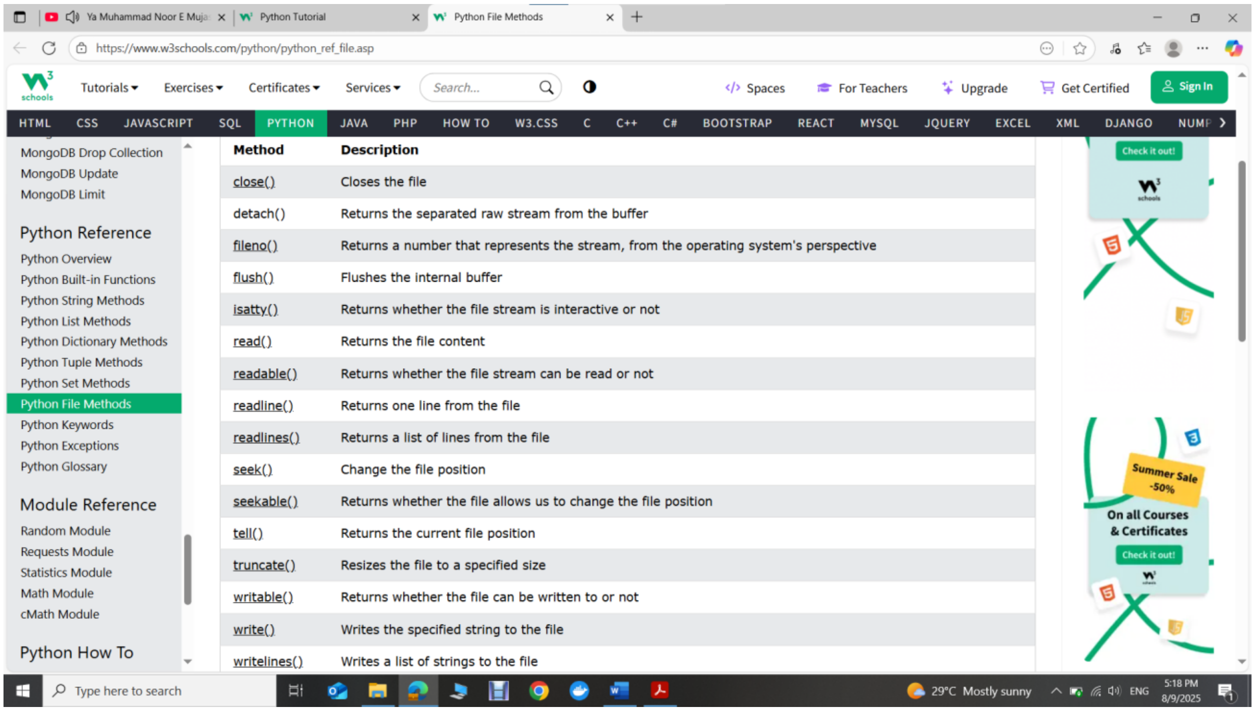
Find the most frequent word in a text file

This problem involves the concept of File Handling.

File Handling:

File Handling involves reading, writing, updating and deleting a text or a binary file.

I studied this concept from W3 Schools.



Method	Description
close()	Closes the file
detach()	Returns the separated raw stream from the buffer
fileno()	Returns a number that represents the stream, from the operating system's perspective
flush()	Flushes the internal buffer
isatty()	Returns whether the file stream is interactive or not
read()	Returns the file content
readable()	Returns whether the file stream can be read or not
readline()	Returns one line from the file
readlines()	Returns a list of lines from the file
seek()	Change the file position
seekable()	Returns whether the file allows us to change the file position
tell()	Returns the current file position
truncate()	Resizes the file to a specified size
writable()	Returns whether the file can be written to or not
write()	Writes the specified string to the file
writelines()	Writes a list of strings to the file