learnings learnt

Shell → Word Frequency

Logic = Iterate through the lines and store the words in the wordsarray. Have a resultsarray and make that empty for now. Iterate though the words in the wordsarray one by one. If word doesn't exist inside the resultsarray then add the word into it and ~~[illegible]~~ iterate though wordsarray again. Everytime the word matches that targetword then increment count. Once you come of the ~~[illegible]~~ for loop, produce the output of the element of the word and the count as well. At the end, it goes back to the first for-loop again, if that word was already in the resultsarray list, it will skip. If not, it will go though same process of going into the second for loop.

After coding my logic :

Problem here is that this will lead to $O(n^2)$ so to be able to tackle this issue, I will utilise an associative array. I tried to use IFS = $' ' but the input field separator doesn't seem to be reading those input files accurately. Hence, I have done

```
IFS = '$\n' read -d '' -a -r wordsarray << (cat words.txt | tr -s ' ' '\n')
```

turns all empty space to new line

So all words will be on separate lines because IFS will separate all the words if it sees '\n' between it and then store it inside words~~[illegible]~~array.

An associative list will be made and if the word is not in the list, it will add it.
The benefit of this is that this is done in $O(n)$ time so it is much more efficient now.

Another for loop is created to iterate through the ~~wordsarray~~ resultsarray to

```
echo "$word $~~word~~{resultsarray["$word"]}"
```

"$word" ↓ the word

${resultsarray["$word"]} ↳ the number associated with the word

The Q said to sort the array based on the frequency which is in the second column of the array. It is a numerical value and we want to store the frequency in our descending order. We will do this by:

sort -k2,2nr after we are done with our for-loop