Q1) How does the map Reduce framework help in processing large datasets?

Ans) The map Reduce framework is a programing model designed to efficiently process & analyse large-scale datasets by distributing computation accross multiple machines.

In map phase, the input data is devided into small chunks, and each chunk is processed in parralled by worker nodes. Each mapper transforms the data into key-value pairs which represent intermediate results

In shuffle phase the frame work automatically groups and transfers all intermediate data based on keys, ensuring that all values associated with the same key are sent to the same reducer.

In the Reduce phase reduces combine or aggregate the values of identical keys to produce the final output.

Q2) Modify the program to count the frequency of words instead of characters. To modify:

Ans) To modify the program for word frequency counting, we replace the logic that process individual characters with one that process

whole words

- mapper function : Instead of iterating over each character, split each line into words using a function like line.split() for each word, emit a key-value pair (word, 1)

- Reducer function : Recieve all values corresponding to the same word and sum them to get the total word frequency

example.

```
def map(line):
    for word in line.split():
        emit(word.lower(), 1)

def reduce(word, counts):
    emit(word, sum(counts))
```

Q3) Explain how you can extend this program to ignore punctuation and number while counting words

Ans) To ensure the program counts only meaningful alphabetic words, we can preprocess the input text to remove unwanted characters such as punctuation marks, digits & special symbols. This can be done using regular expressions

example :-

```
import re
def map(line):

    clean_line = re.sub(r'[^a-zA-Z\s]', '', line)
    for word in clean_line.lower().split()
        emit(word, 1)
```

Q4) Why is it important to make the character count case-insensetive?

Ans:) making the count case-insensetive ensures that letters or words ~~word~~ written in different cases are treated as the same entity.
For instance the words "Apple", "apple" should all represent the same word.
If the program treats them differently. it will create inconsistent & misleading counts
By converting all the input texts to lowercase before processing, we achieve uniformity. improve data accuracy & simplify later analysis.