***Information Management - Assignment 4***

*Santhosh Kumar Ramkumar (SR55965)*

*Sanjhana Rangaraj (SR55737)*

*Rajkumar Rajavel (RR53572)*

**Answers -**

A diagram of fruit and vegetables

Description automatically generated with medium confidence

A screenshot of a computer application

Description automatically generated

**Code:**

from collections import defaultdict

import string

# Input data

data = """

Two roads diverged in a yellow wood,

And sorry I could not travel both

And be one traveler, long I stood

And looked down one as far as I could

To where it bent in the undergrowth;

Then took the other, as just as fair,

And having perhaps the better claim,

Because it was grassy and wanted wear;

Though as for that the passing there

Had worn them really about the same,

And both that morning equally lay

In leaves no step had trodden black.

Oh, I kept the first for another day!

Yet knowing how way leads on to way,

I doubted if I should ever come back.

I shall be telling this with a sigh

Somewhere ages and ages hence:

Two roads diverged in a wood, and I—

I took the one less traveled by,

And that has made all the difference.

"""

def mapper(input\_text):

word\_count = defaultdict(int)

words = input\_text.strip().split() # Split by spaces to get individual words

for word in words:

# Remove punctuation and convert to lowercase for consistent counting

word = word.strip(string.punctuation).lower()

word\_count[word] += 1

return word\_count

def reducer(mapper\_outputs):

final\_word\_count = defaultdict(int)

for word, counts in mapper\_outputs.items(): # Iterate over the dictionary

final\_word\_count[word] = sum(counts)

return final\_word\_count

def main():

input\_data = data.splitlines()

mapper\_outputs = []

# Map Phase

for line in input\_data:

mapper\_output = mapper(line)

mapper\_outputs.append(mapper\_output)

# Shuffle and Sort Phase

intermediate\_data = defaultdict(list)

for output in mapper\_outputs:

for word, count in output.items():

intermediate\_data[word].append(count)

# Reduce Phase

final\_word\_count = reducer(intermediate\_data)

# Sort the final word count dictionary by counts in descending order

sorted\_final\_word\_count = dict(sorted(final\_word\_count.items(), key=lambda x: x[1], reverse=True))

# Output word frequencies

for word, count in sorted\_final\_word\_count.items():

print(f"{word}: {count}")

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Output**

and: 9

i: 8

the: 8

as: 5

in: 4

a: 3

one: 3

that: 3

two: 2

roads: 2

diverged: 2

wood: 2

could: 2

both: 2

be: 2

to: 2

it: 2

took: 2

for: 2

had: 2

way: 2

ages: 2

yellow: 1

sorry: 1

not: 1

travel: 1

traveler: 1

long: 1

stood: 1

looked: 1

down: 1

far: 1

where: 1

bent: 1

undergrowth: 1

then: 1

other: 1

just: 1

fair: 1

having: 1

perhaps: 1

better: 1

claim: 1

because: 1

was: 1

grassy: 1

wanted: 1

wear: 1

though: 1

passing: 1

there: 1

worn: 1

them: 1

really: 1

about: 1

same: 1

morning: 1

equally: 1

lay: 1

leaves: 1

no: 1

step: 1

trodden: 1

black: 1

oh: 1

kept: 1

first: 1

another: 1

day: 1

yet: 1

knowing: 1

how: 1

leads: 1

on: 1

doubted: 1

if: 1

should: 1

ever: 1

come: 1

back: 1

shall: 1

telling: 1

this: 1

with: 1

sigh: 1

somewhere: 1

hence: 1

i—: 1

less: 1

traveled: 1

by: 1

has: 1

made: 1

all: 1

difference: 1