Задание 3

ХУАН ЦЗИНЬЯНЬ

1. Write a program to find the longest word.

(1) Install Ray

(2) Replace the file path with the actual path

(3) Run the program

```python
import ray
import re

# initialization Ray
ray.init()

# Define a Ray task to find the longest word of each line
1 usage
@ray.remote
def find_longest_word(line):
    # Using regular expressions to extract words
    words = re.findall( pattern: r'\b\w+\b', line)
    # Find the longest word and return
    if words:
        return max(words, key=len)
    return ""

# Read the file and process each line
1 usage
def process_file(filepath):
    longest_words = []
    with open(filepath, 'r', encoding='utf-8') as file:
        # Submission of tasks and collection of results
        future_results = [find_longest_word.remote(line) for line in file]
        results = ray.get(future_results)
        # Find the longest word from all the returned longest words
        longest_word = max(results, key=len)
        return longest_word

# Specify your file path
file_path = 'D:\Задание 3\wiki.txt'
# Process the file and print the longest word
longest_word = process_file(file_path)
print("The longest word is:", longest_word)
```

2.Write a program to find the average length of words.

(1) Install Ray

(2) Replace the file path with the actual path

(3) Run the program

```python
import ray
import os

# initialization Ray
ray.init()

# Define functions to process a single row of data
1 usage
@ray.remote
def process_line(line):
    # Separate components in each row
    parts = line.split('<tabs>')
    if len(parts) < 3:
        return [], 0  # If the format is incorrect, returns an empty list and a length of 0

    # Extract the body part, split the words
    content = parts[2].strip()
    words = content.split()
    lengths = [len(word) for word in words]
    return lengths, len(words)

# Reading and processing files
1 usage
def process_file(file_path):
    with open(file_path, 'r', encoding='utf-8') as file:
        # Create a list of tasks, with each row of data corresponding to a task
        futures = [process_line.remote(line) for line in file]

    # Collection of processing results
    results = ray.get(futures)

    # Calculate total length and total number of words
    total_length = sum(sum(lengths) for lengths, count in results)
    total_count = sum(count for lengths, count in results)

    # Calculate average word length
    if total_count > 0:
        average_length = total_length / total_count
        return average_length
    else:
        return 0

# Replace the following path with your file path
file_path = 'D:\Задание 3\wiki.txt'
average_length = process_file(file_path)
print(f"Average word length: {average_length:.2f}")
```

3. Write a program to find the most commonly used words made up of the Latin alphabet.

(1) Install Ray

(2) Replace the file path with the actual path

(3) Run the program

```python
import ray
import re
from collections import Counter

# initialization Ray
ray.init()


# Define a Ray's remote function to count words
1 usage
@ray.remote
def count_words(text):
    # Extract only words containing Latin letters
    words = re.findall( pattern: r'\b[a-zA-Z]+\b', text)
    # Count the number of occurrences of each word
    return Counter(words)


1 usage
def main():
    file_path = 'D:\Задание 3\wiki.txt'  # Replace with your file path
    results = []

    # Reads the file line by line and creates a Ray task for each line
    with open(file_path, 'r', encoding='utf-8') as file:
        for line in file:
            if '<tabs>' in line:
                parts = line.split('<tabs>')
                if len(parts) >= 3:
                    url, title, content = parts[0], parts[1], parts[2]
                    # Send title and content to remote function
                    results.append(count_words.remote(title + ' ' + content))

    # Merge all results
    total_counts = Counter()
    for result in results:
        total_counts.update(ray.get(result))

    # Find the most commonly used words
    most_common_word = total_counts.most_common(1)
    print("The most commonly used words are: ", most_common_word)


if __name__ == "__main__":
    main()
```

4. Find all words that begin with a capital letter more than half the time and occur more than 10 times.

(1) Install Ray

(2) Replace the file path with the actual path

(3) Run the program

```python
import ray
import re

# initialization Ray
ray.init()

1 usage  新 *
@ray.remote
def count_words(line):
    # Use regular expressions to remove the tags, leaving only the body part
    text = re.split( pattern: r' <tabs> ', line)[-1]
    # Match all words using regular expressions
    words = re.findall( pattern: r'\b[A-Z][a-z]*\b', text)
    return words

1 usage  新 *
def main():
    # Replace with your file path
    file_path = 'D:\Задание 3\wiki.txt'
    # Read file
    with open(file_path, 'r', encoding='utf-8') as file:
        lines = file.readlines()

    # Distributing tasks using Ray
    futures = [count_words.remote(line) for line in lines]
    results = ray.get(futures)

    # Flat results list
    all_words = [word for sublist in results for word in sublist]

    # Counting word occurrences
    word_count = {}
    for word in all_words:
        if word in word_count:
            word_count[word] += 1
        else:
            word_count[word] = 1

    # Filter words with at least 10 occurrences
    frequent_words = {word: count for word, count in word_count.items() if count >= 10}

    # output result
    for word, count in frequent_words.items():
        print(f"{word}: {count}")

if __name__ == '__main__':
    main()
```

5. Write a program that uses statistics to identify pr., dr., ... forms of stable abbreviations.

(1) Install Ray

(2) Replace the file path with the actual path

(3) Run the program

```python
import ray
import re

# 初始化 Ray
ray.init()


# 定义一个 Ray 任务来处理文件的每一行
1 usage  新 *
@ray.remote
def find_abbreviations(line):
    # 使用正则表达式匹配缩写，如 pr., dr. 等
    abbreviations = re.findall( pattern: r'\b[a-zA-Z]+\.', line)
    return abbreviations


1 usage  新 *
def process_file(file_path):
    results = []
    with open(file_path, 'r', encoding='utf-8') as file:
        for line in file:
            # 为每一行文本启动一个 Ray 任务
            result = find_abbreviations.remote(line)
            results.append(result)

    # 收集所有结果
    results = ray.get(results)

    # 打印所有识别的缩写
    for result in results:
        if result:
            print(result)


# 替换为你的文件路径
file_path = 'your_file_path'
process_file(file_path)

```

6. write a program to find out t.p., n.e., ... using statistical data stable abbreviations of the form.

(1) Install Ray

(2) Replace the file path with the actual path

(3) Run the program

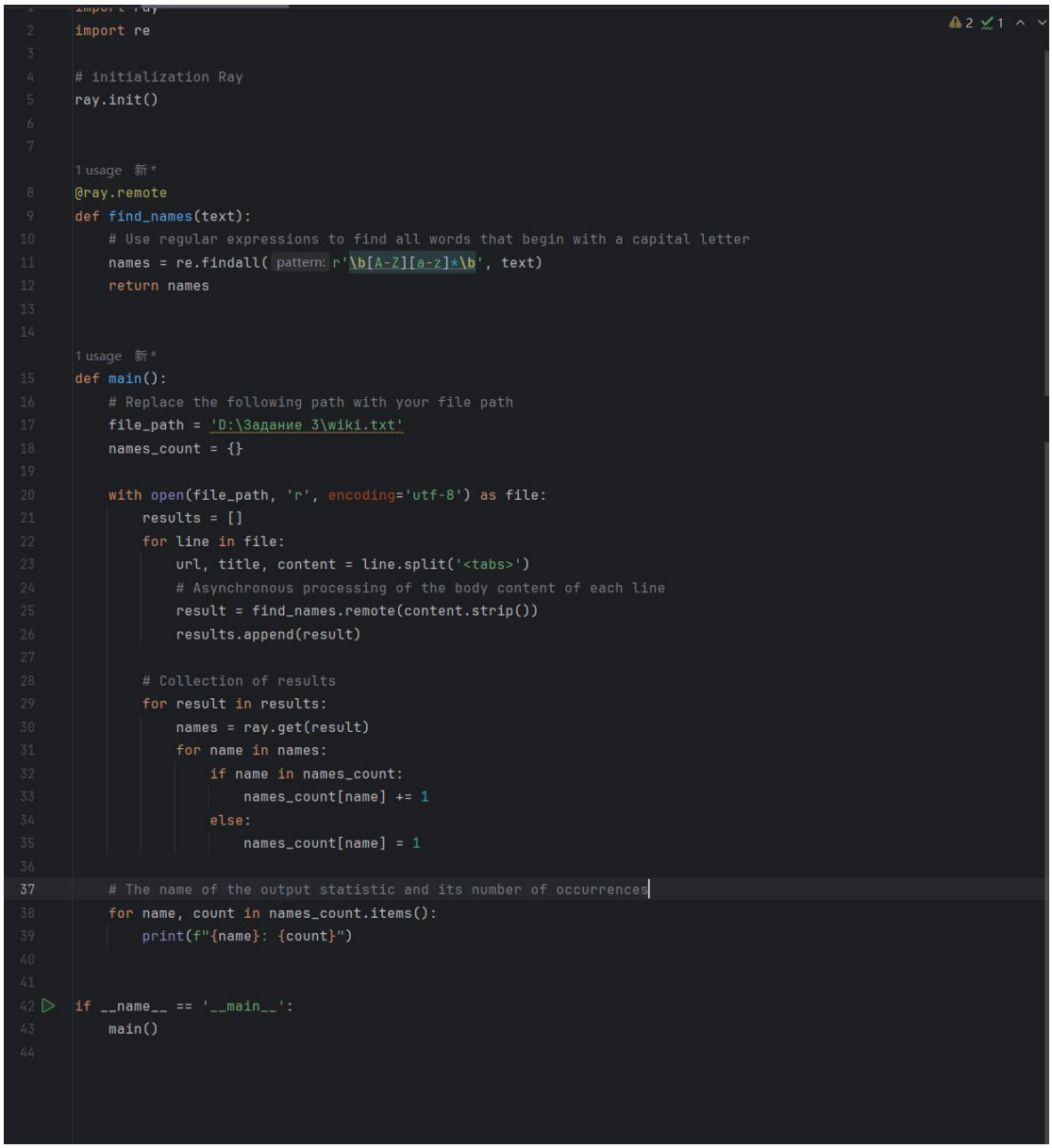
```python
import ray
import re

# initialization Ray
ray.init()

# Define a Ray remote function that finds abbreviations in text
1 usage  新 *
@ray.remote
def find_abbreviations(text):
    # Regular expressions match abbreviations like t.p., n.e.
    abbreviations = re.findall( pattern: r'\b\w+\.\w+\.', text)
    return abbreviations

1 usage  新 *
def main():
    # Specify your file path
    file_path = 'D:\Задание 3\wiki.txt'

    # Create a list of all future objects.
    futures = []

    # Open and read files
    with open(file_path, 'r', encoding='utf-8') as file:
        for line in file:
            # Assuming that the format of each line is: URL <tag> title <tag> full text
            parts = line.split('<tabs>')
            if len(parts) >= 3:
                content = parts[2]  # Access to the body of the text
                # Calls the remote function and adds the resulting object to the futures list
                future = find_abbreviations.remote(content)
                futures.append(future)

    # Collect all results
    results = ray.get(futures)

    # Output all found abbreviations
    all_abbreviations = set()
    for result in results:
        all_abbreviations.update(result)

    print(all_abbreviations)

if __name__ == '__main__':
    main()
```

7. Write a program that uses statistics to find names used in articles.

(1) Install Ray

(2) Replace the file path with the actual path

(3) Run the program



```python
import ray
import re

# initialization Ray
ray.init()


1 usage  新*
@ray.remote
def find_names(text):
    # Use regular expressions to find all words that begin with a capital letter
    names = re.findall( pattern: r'\b[A-Z][a-z]*\b', text)
    return names


1 usage  新*
def main():
    # Replace the following path with your file path
    file_path = 'D:\Задание 3\wiki.txt'
    names_count = {}

    with open(file_path, 'r', encoding='utf-8') as file:
        results = []
        for line in file:
            url, title, content = line.split('<tabs>')
            # Asynchronous processing of the body content of each line
            result = find_names.remote(content.strip())
            results.append(result)

        # Collection of results
        for result in results:
            names = ray.get(result)
            for name in names:
                if name in names_count:
                    names_count[name] += 1
                else:
                    names_count[name] = 1

    # The name of the output statistic and its number of occurrences
    for name, count in names_count.items():
        print(f"{name}: {count}")


if __name__ == '__main__':
    main()
```