Neural Network and Deep Learning

ICP-2

Name – Prem Sai Chigullapalli

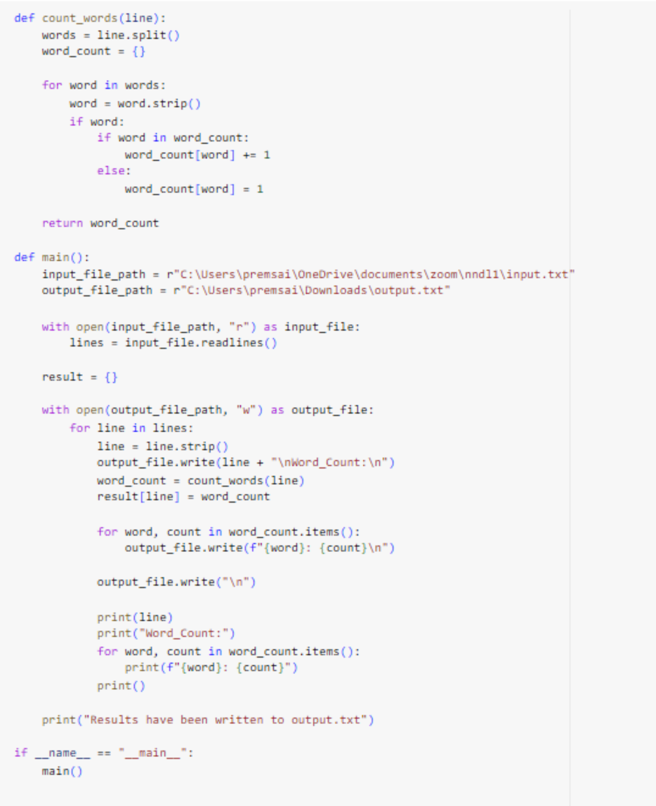
700 Number- 700757659

CRN- 30828

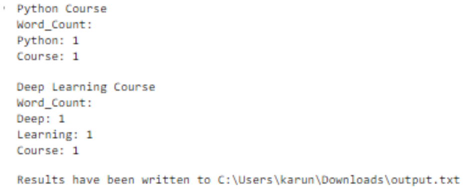
Github link- https://github.com/premsaichigullapalli/NNDL-2.git

Drive Link - https://drive.google.com/file/d/1ndJXRZeqa5-2Nvwr3JvHoGJ_7T_yEsIn/view?usp=drive_link

1) Here I took two strings as input- first_name, last_name. Pass these variables as input and return the full_name. A function named "string_alternative" that returns every other char in the full_name string.

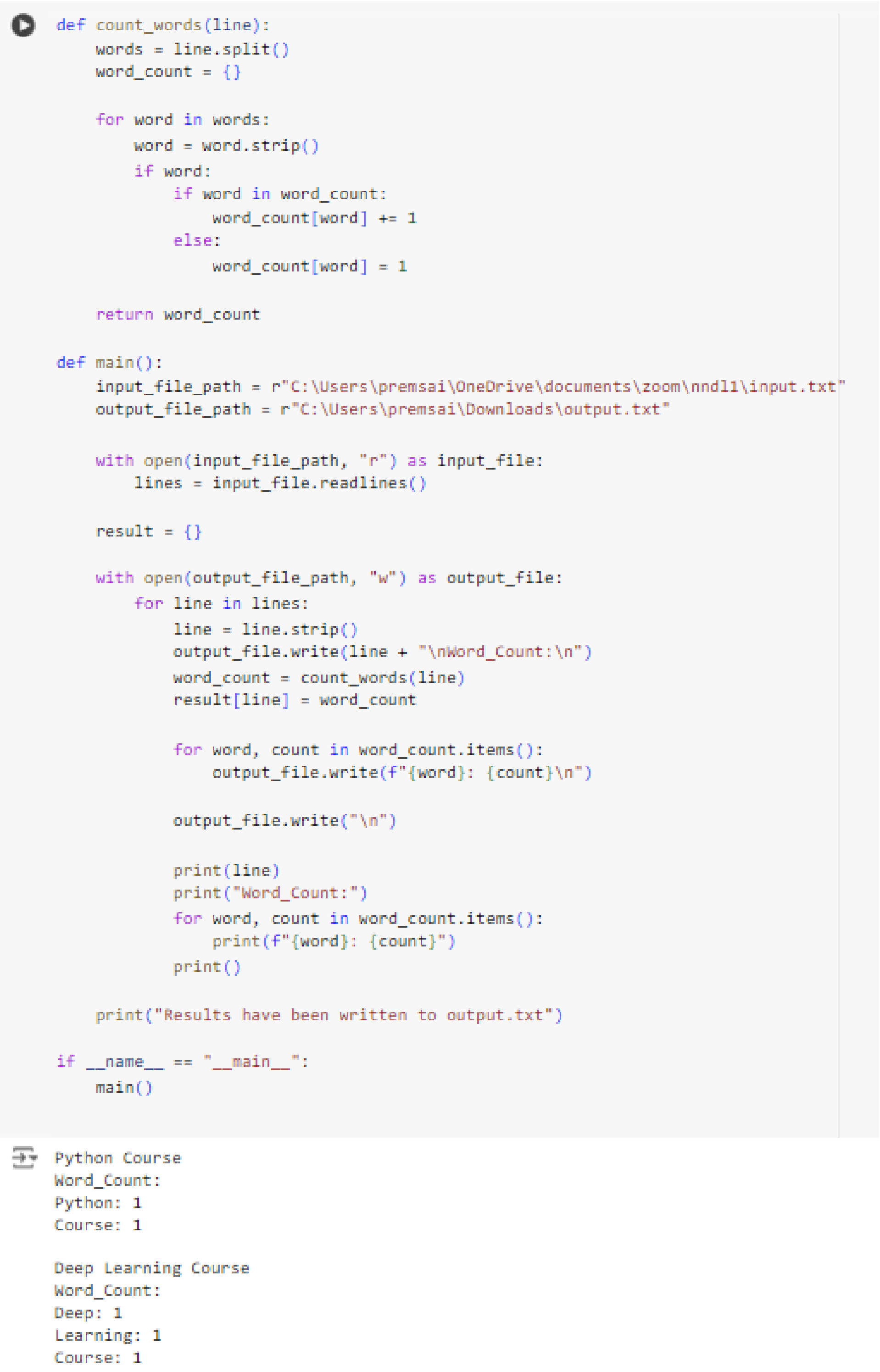


```python
def count_words(line):
    words = line.split()
    word_count = {}

    for word in words:
        word = word.strip()
        if word:
            if word in word_count:
                word_count[word] += 1
            else:
                word_count[word] = 1

    return word_count

def main():
    input_file_path = r"C:\Users\premsai\OneDrive\documents\zoom\nndl1\input.txt"
    output_file_path = r"C:\Users\premsai\Downloads\output.txt"

    with open(input_file_path, "r") as input_file:
        lines = input_file.readlines()

    result = {}

    with open(output_file_path, "w") as output_file:
        for line in lines:
            line = line.strip()
            output_file.write(line + "\nWord_Count:\n")
            word_count = count_words(line)
            result[line] = word_count

            for word, count in word_count.items():
                output_file.write(f"{word}: {count}\n")

            output_file.write("\n")

            print(line)
            print("Word_Count:")
            for word, count in word_count.items():
                print(f"{word}: {count}")
            print()

    print("Results have been written to output.txt")

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



```
Python Course
Word_Count:
Python: 1
Course: 1

Deep Learning Course
Word_Count:
Deep: 1
Learning: 1
Course: 1

Results have been written to C:\Users\karun\Downloads\output.txt
```

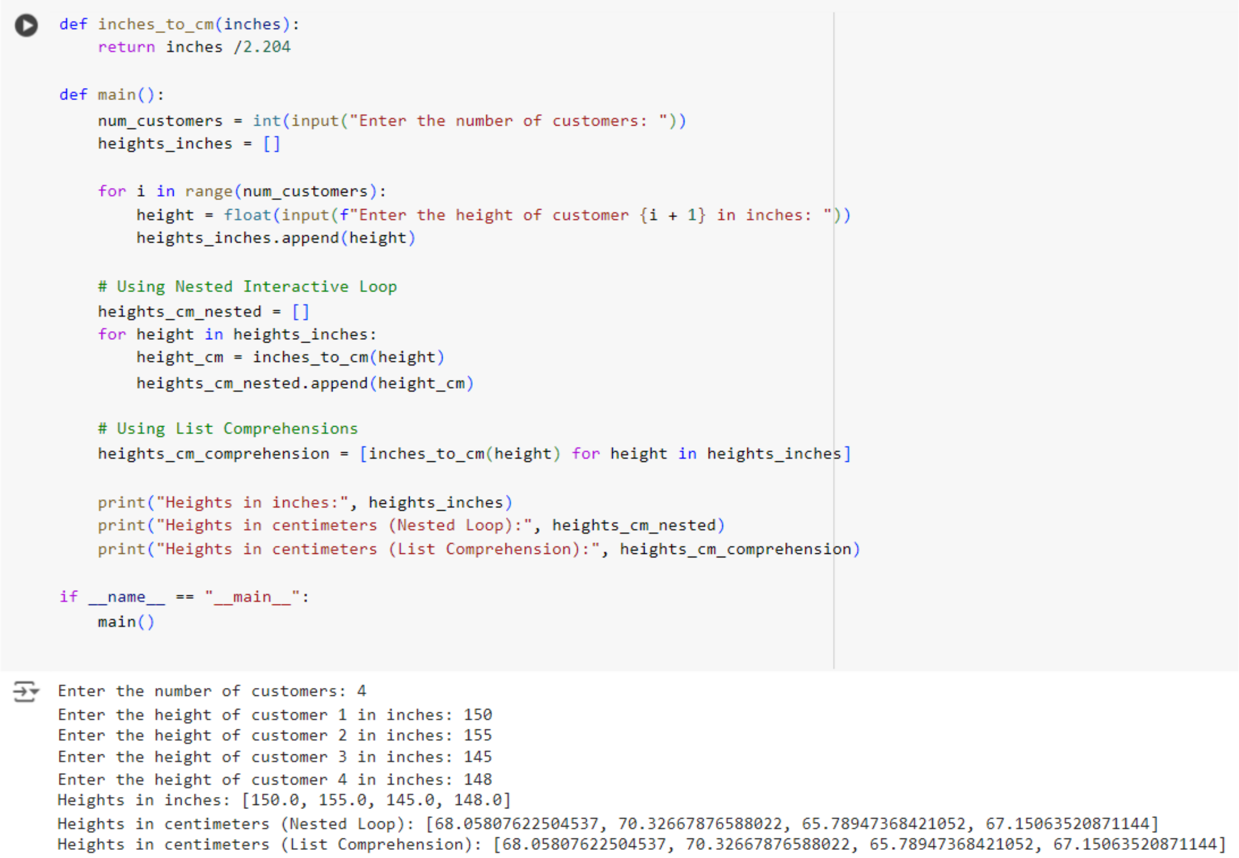
2) An input file includes two lines(input.txt). Function counts_words will take each line as input and give the word count. I stored the output in output.txt.



```
def count_words(line):
    words = line.split()
    word_count = {}

    for word in words:
        word = word.strip()
        if word:
            if word in word_count:
                word_count[word] += 1
            else:
                word_count[word] = 1

    return word_count

def main():
    input_file_path = r"C:\Users\premsai\OneDrive\documents\zoom\nndl1\input.txt"
    output_file_path = r"C:\Users\premsai\Downloads\output.txt"

    with open(input_file_path, "r") as input_file:
        lines = input_file.readlines()

    result = {}

    with open(output_file_path, "w") as output_file:
        for line in lines:
            line = line.strip()
            output_file.write(line + "\nWord_Count:\n")
            word_count = count_words(line)
            result[line] = word_count

            for word, count in word_count.items():
                output_file.write(f"{word}: {count}\n")

            output_file.write("\n")

            print(line)
            print("Word_Count:")
            for word, count in word_count.items():
                print(f"{word}: {count}")
            print()

    print("Results have been written to output.txt")

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

```
Python Course
Word_Count:
Python: 1
Course: 1

Deep Learning Course
Word_Count:
Deep: 1
Learning: 1
Course: 1
```

3) I took heights in inches of customers into a list as input. I wrote a function inches_to_centimeters) to convert the list into centimeters.



```python
def inches_to_cm(inches):
    return inches /2.204

def main():
    num_customers = int(input("Enter the number of customers: "))
    heights_inches = []

    for i in range(num_customers):
        height = float(input(f"Enter the height of customer {i + 1} in inches: "))
        heights_inches.append(height)

    # Using Nested Interactive Loop
    heights_cm_nested = []
    for height in heights_inches:
        height_cm = inches_to_cm(height)
        heights_cm_nested.append(height_cm)

    # Using List Comprehensions
    heights_cm_comprehension = [inches_to_cm(height) for height in heights_inches]

    print("Heights in inches:", heights_inches)
    print("Heights in centimeters (Nested Loop):", heights_cm_nested)
    print("Heights in centimeters (List Comprehension):", heights_cm_comprehension)

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

```
Enter the number of customers: 4
Enter the height of customer 1 in inches: 150
Enter the height of customer 2 in inches: 155
Enter the height of customer 3 in inches: 145
Enter the height of customer 4 in inches: 148
Heights in inches: [150.0, 155.0, 145.0, 148.0]
Heights in centimeters (Nested Loop): [68.05807622504537, 70.32667876588022, 65.78947368421052, 67.15063520871144]
Heights in centimeters (List Comprehension): [68.05807622504537, 70.32667876588022, 65.78947368421052, 67.15063520871144]
```