Name: Angad Sandhu
Registration Number: 190905494
Section: CSE-A
Roll Number: 60
Subject: DS Lab 5

1) Try the given wordcount program for heart disease dataset, covid 19 dataset, example dataset and german credit dataset

 # mapper.py

```
import sys

for line in sys.stdin:
    words = line.strip().split(',')

    for word in words:
        print("%s\t%d"%(word, 1))
```

#reduce.py

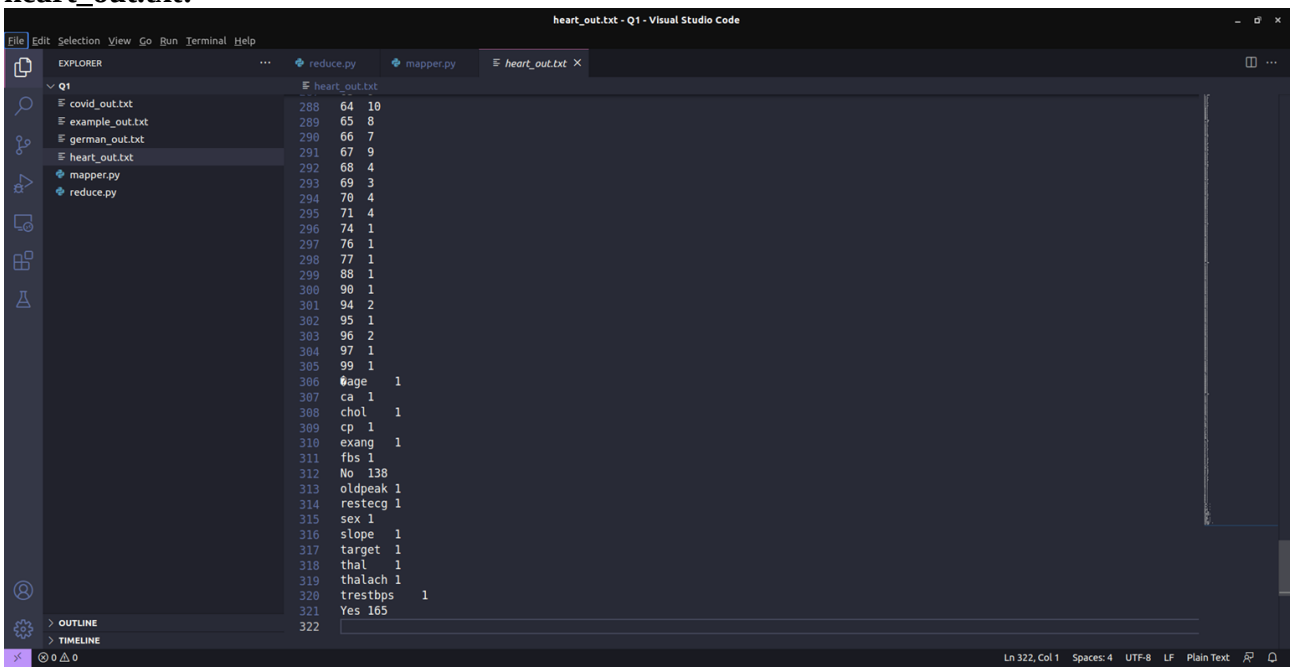
```
from operator import itemgetter
import sys

current_word = None
current_count = 0
word = None

for line in sys.stdin:
    try:
        word, count = line.strip().split('\t', 1)
        count = int(count)
    except ValueError:
        continue

    if current_word == word:
        current_count += count
    else:
        if current_word:
            print("%s\t%d"%(current_word, current_count))

        current_count = count
        current_word = word

if current_word == word:
    print("%s\t%d"%(current_word, current_count))
```

Output:

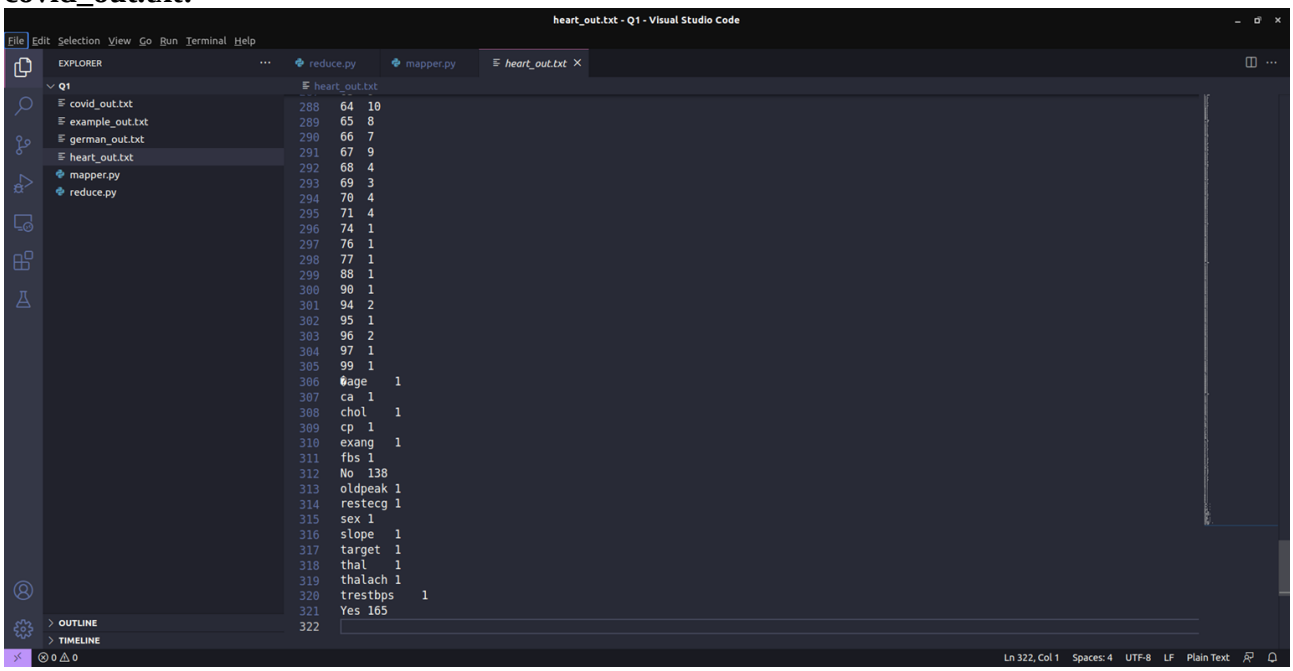
```
PS D:\Code\CSE\Year 3\Sem 6\DSL\Week 5> `
>> cat data/heart_disease_data.csv | python Q1/mapper.py | sort | python Q1/reduce.py > Q1/heart_out.txt
PS D:\Code\CSE\Year 3\Sem 6\DSL\Week 5> `
>> cat data/covid_19_data.csv | python Q1/mapper.py | sort | python Q1/reduce.py > Q1/covid_out.txt
PS D:\Code\CSE\Year 3\Sem 6\DSL\Week 5> `
>> cat data/example.csv | python Q1/mapper.py | sort | python Q1/reduce.py > Q1/example_out.txt
PS D:\Code\CSE\Year 3\Sem 6\DSL\Week 5> `
>> cat data/german_credit.csv | python Q1/mapper.py | sort | python Q1/reduce.py > Q1/german_out.txt
PS D:\Code\CSE\Year 3\Sem 6\DSL\Week 5>
```

Output files (trucated)

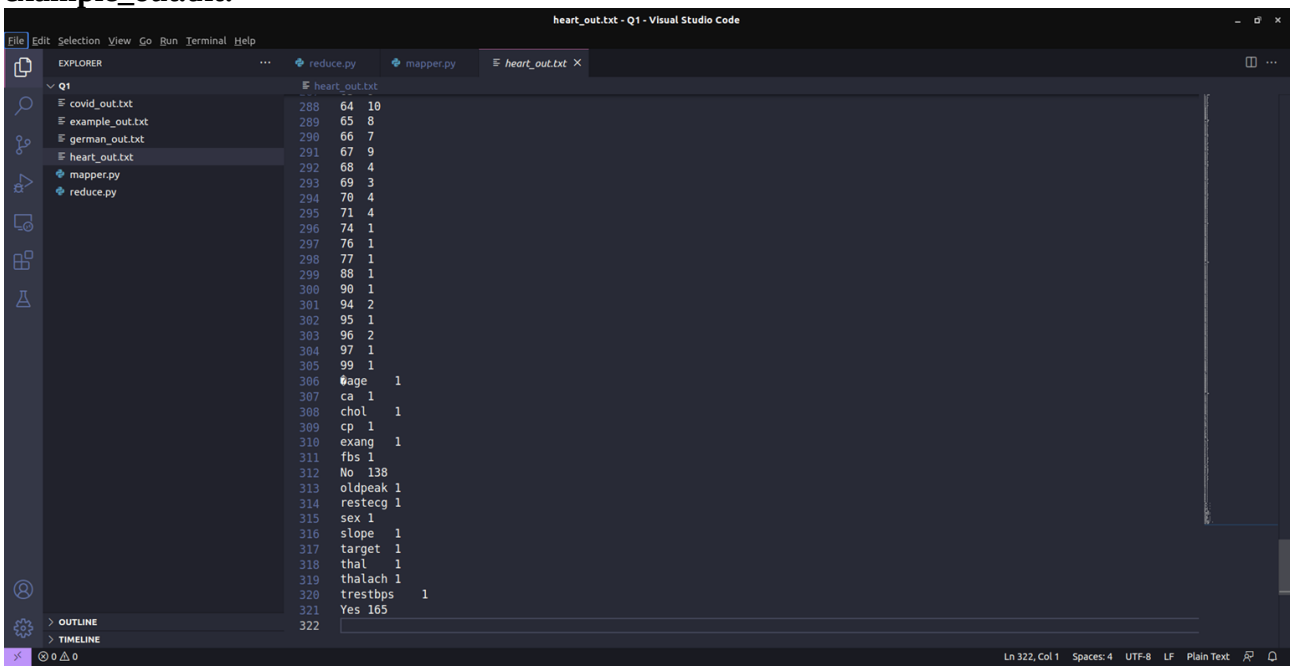
heart_out.txt:



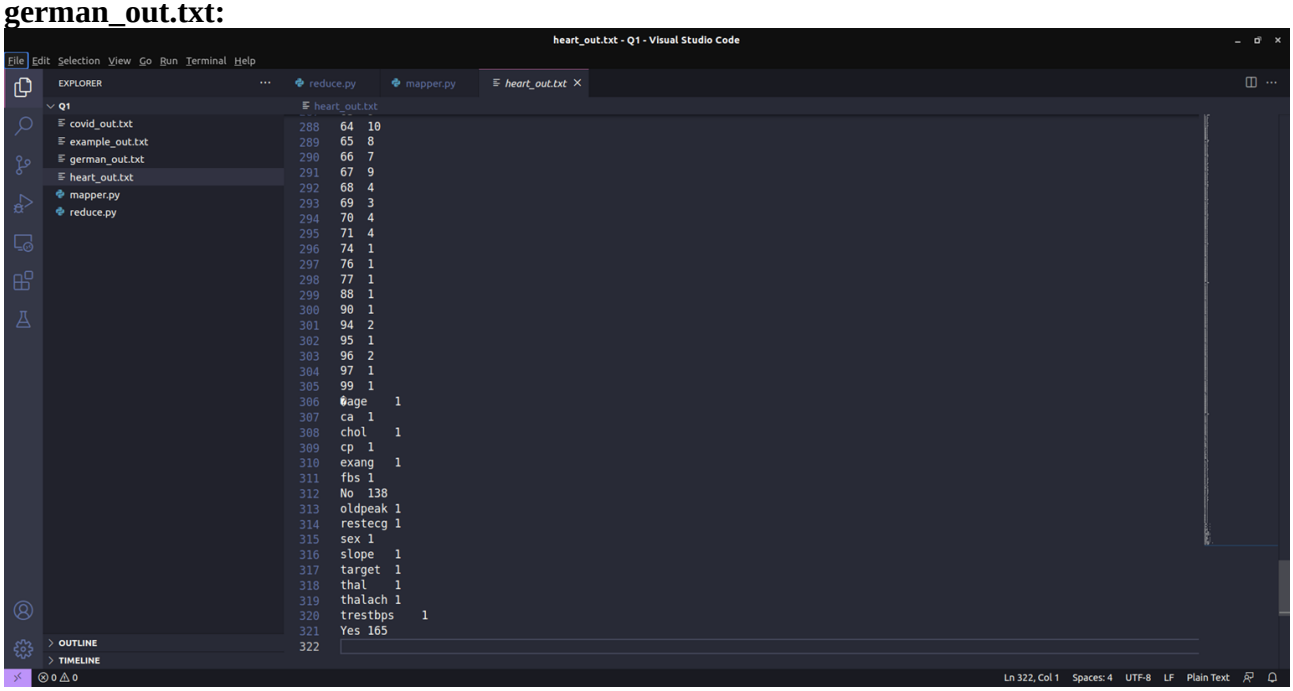
covid_out.txt:



example_out.txt:



german_out.txt:



2) Map Reduce Program to find frequent words
Try the given frequent word count program for heart disease dataset, covid 19 dataset, example dataset, and german credit dataset

#freqmap1.py:
```
import sys

for line in sys.stdin:
    L = [ (word.strip().lower(), 1) for word in line.strip().split(',') ]

    for word, n in L:
        print("{}\t{}".format(word, n))
```

#freqred1.py
```
import sys

lastWord = None
sum = 0

for line in sys.stdin:
    try:
        word, count = line.strip().split('\t', 1)
        count = int(count)
    except ValueError:
        continue

    if lastWord == None:
        lastWord = word
        sum = count
        continue

    if word == lastWord:
        sum += count

    else:
        print("{}\t{}".format(lastWord, sum))
        sum = count
        lastWord = word

# output last word
if lastWord == word:
    print("{}\t{}".format(lastWord, sum))
```

#freqmap2.py:
```
import sys

for line in sys.stdin:
    word, count = line.strip().split('\t', 1)
    count = int(count)
    print("{}\t{}".format(word, count))
```

#freqred2.py:

```
import sys

mostFreq = []
currentMax = -1

for line in sys.stdin:
    word, count = line.strip().split('\t', 1)
    count = int(count)

    if count > currentMax:
        currentMax = count
        mostFreq = [ word ]

    elif count == currentMax:
        mostFreq.append(word)

for word in mostFreq:
    print("{}\t{}".format(word, currentMax))
```

Output:

```
PS D:\Code\CSE\Year 3\Sem 6\DSL\Week 5> `
>> cat data/heart_disease_data.csv | python Q2/freqmap1.py | sort | python Q2/freqred1.py | python Q2/freqmap2.py | sort | python Q2/freqred2.py > Q2/heart_
out.txt
>> cat data/covid_19_data.csv | python Q2/freqmap1.py | sort | python Q2/freqred1.py | python Q2/freqmap2.py | sort | python Q2/freqred2.py > Q2/covid_out.t
xt
>> cat data/example.csv | python Q2/freqmap1.py | sort | python Q2/freqred1.py | python Q2/freqmap2.py | sort | python Q2/freqred2.py > Q2/example_out.txt
>> cat data/german_credit.csv | python Q2/freqmap1.py | sort | python Q2/freqred1.py | python Q2/freqmap2.py | sort | python Q2/freqred2.py > Q2/german_out.
txt
>>
```

heart_out.txt:

0 1145

covid_out.txt:

0 45012

example_out.txt:

amex 13

german_out.txt:

1 700

3) MapReduce Program to explore the dataset and perform filtering (typically creating key value pairs) by mapper and perform count and summary operations on instances.

#itemmap.py (for heart disease dataset)

```
import fileinput

for line in fileinput.input():
    data = line.strip().split(",")

    if len(data) == 14:
        age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang, oldpeak, slope, ca, thal, target = data
        print("{}\t{}".format(age, trestbps))
```

#itemmap.py (for covid 19 dataset)

```
import fileinput

for line in fileinput.input():
    data = line.strip().split(",")

    if len(data) == 8:
        sno, observationdate, province, country, lastupdate, confirmed, deaths, recovered = data
        print("{}\t{}".format(country, confirmed))
```

#itemmap.py (for example dataset)

```
import fileinput

for line in fileinput.input():
    data = line.strip().split(",")

    if len(data) == 6:
        date, time, location, itemtype, amount, cardtype = data
        print("{}\t{}".format(itemtype, amount))
```

#itemmap.py (for german credit dataset)

```
import fileinput

for line in fileinput.input():
    data = line.strip().split(",")

    if len(data) == 3:
        credibility, creditamount, durationofcredit = data
        print("{}\t{}".format(credibility, creditamount))
```

#itemred.py:

```
import fileinput

transaction_count = 0
sales_total = 0

for line in fileinput.input():

    try:
```

```
        data = line.strip().split("\t")
        if len(data) != 2:
            continue
    except ValueError:
        continue

    current_key, current_value = data
    try:
        sales_total += float(current_value)
        transaction_count += 1
    except ValueError:
        continue

print("{}\t{}".format(transaction_count, sales_total))
```

Output:

```
PS D:\Code\CSE\Year 3\Sem 6\DSL\Week 5> `
>> cat data/heart_disease_data.csv | python Q3/itemmap.py | sort | python Q3/itemred.py > Q3/heart_out.txt
>> cat data/covid_19_data.csv | python Q3/itemmap.py | sort | python Q3/itemred.py > Q3/covid_out.txt
>> cat data/example.csv | python Q3/itemmap.py | sort | python Q3/itemred.py > Q3/example_out.txt
>> cat data/german_credit.csv | python Q3/itemmap.py | sort | python Q3/itemred.py > Q3/german_out.txt
PS D:\Code\CSE\Year 3\Sem 6\DSL\Week 5>
```

4) Write a mapper and reducer program for word count by defining a seprator instead of using "\t"

#sepmap.py

```
import sys

def read_input(file):
    for line in file:
        yield line.strip().split(',')

def main(separator="\t"):
    data = read_input(sys.stdin)
    for words in data:
        for word in words:
            print("%s%s%d"%(word, separator, 1))

if __name__ == '__main__':
    sep = sys.argv[1]
    main(separator=sep)
```

#sepred.py:

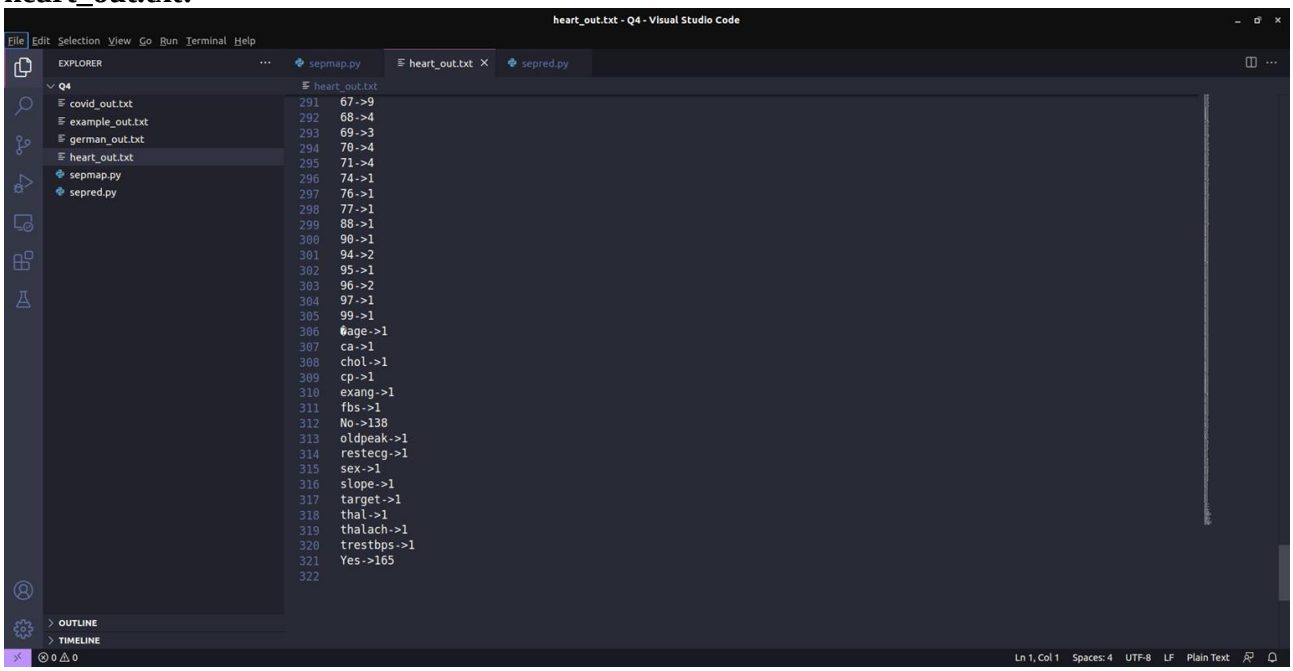
```
import sys
from itertools import groupby
from operator import itemgetter

def read_mapper_output(file, separator='\t'):
    for line in file:
        yield line.rstrip().split(separator, 1)

def main(separator="\t"):
    data = read_mapper_output(sys.stdin, separator=separator)
    for current_word, group in groupby(data, itemgetter(0)):
        try:
            total_count = sum(int(count) for current_word, count in group)
            print("%s%s%d"%(current_word, separator, total_count))
        except ValueError:
            pass

if __name__ == '__main__':
    sep = sys.argv[1]
    main(separator=sep)
```

Output:

```
PS D:\Code\CSE\Year 3\Sem 6\DSL\Week 5> `
>> cat data/heart_disease_data.csv | python Q4/sepmap.py "->" | sort | python Q4/sepred.py "->" > Q4/heart_out.txt
>> cat data/covid_19_data.csv | python Q4/sepmap.py "->" | sort | python Q4/sepred.py "->" > Q4/covid_out.txt
>> cat data/example.csv | python Q4/sepmap.py "->" | sort | python Q4/sepred.py "->" > Q4/example_out.txt
>> cat data/german_credit.csv | python Q4/sepmap.py "->" | sort | python Q4/sepred.py "->" > Q4/german_out.txt
```

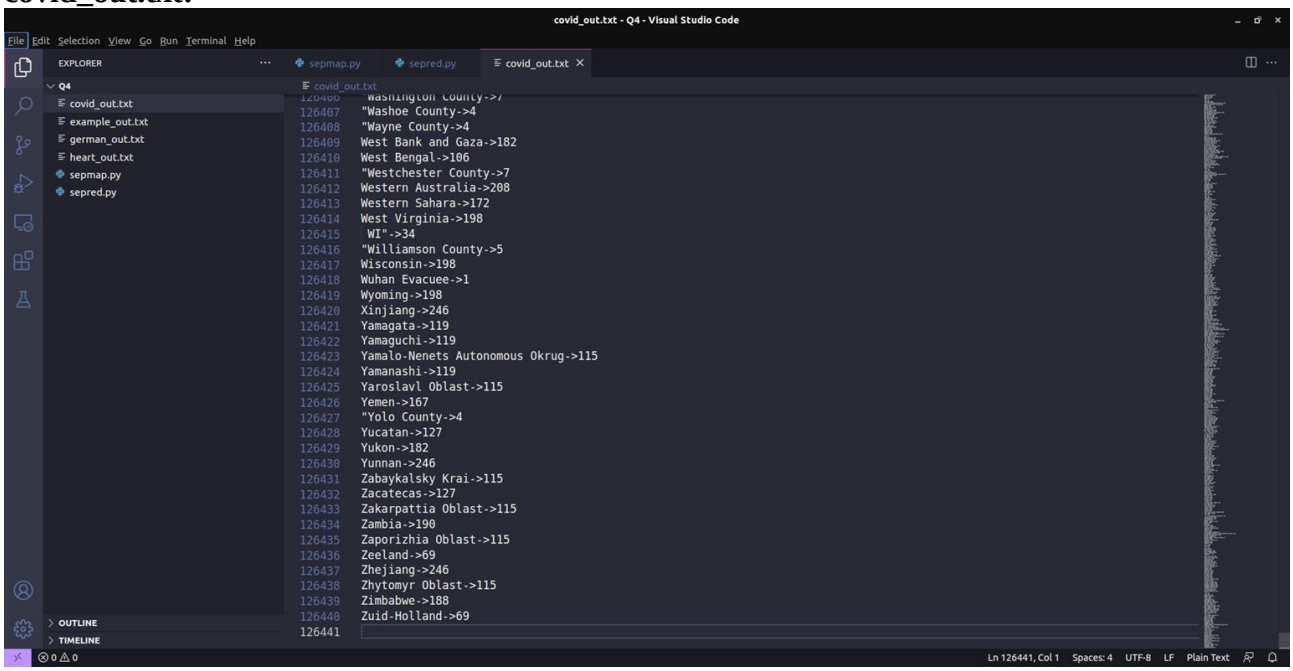
Output files (truncated):

heart_out.txt:



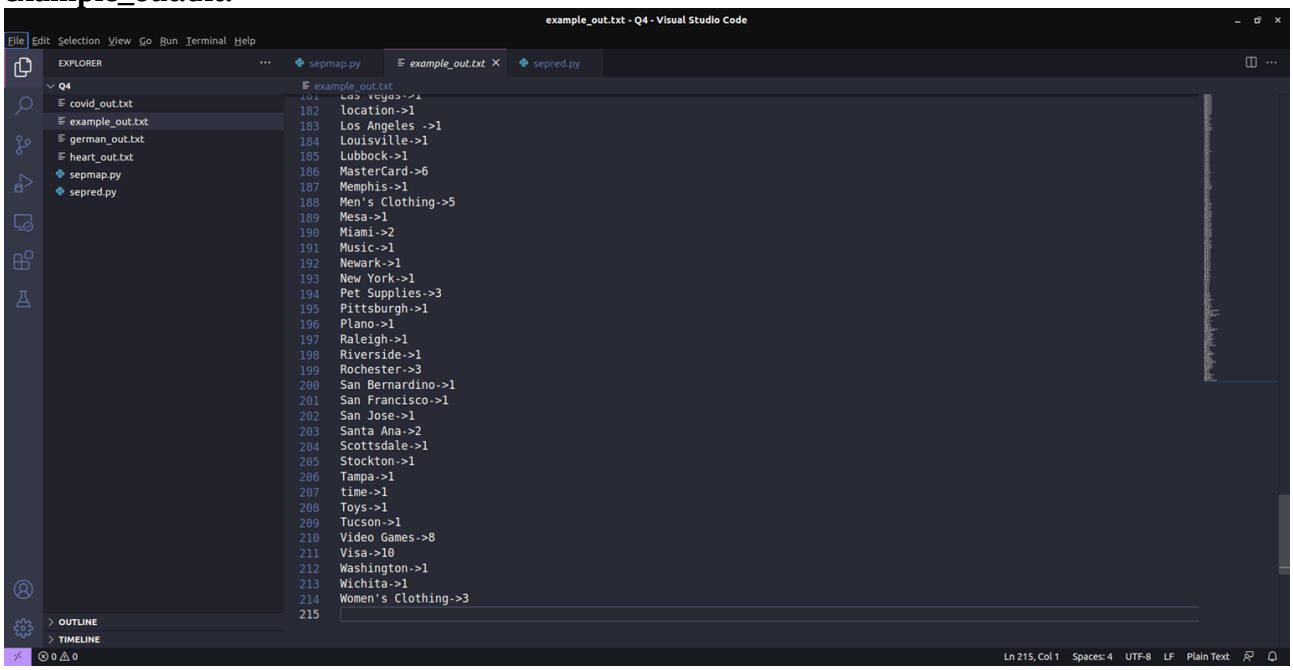

```
67->9
68->4
69->3
70->4
71->4
74->1
76->1
77->1
88->1
90->1
94->2
95->1
96->2
97->1
99->1
age->1
ca->1
chol->1
cp->1
exang->1
fbs->1
No->138
oldpeak->1
restecg->1
sex->1
slope->1
target->1
thal->1
thalach->1
trestbps->1
Yes->165
```

covid_out.txt:

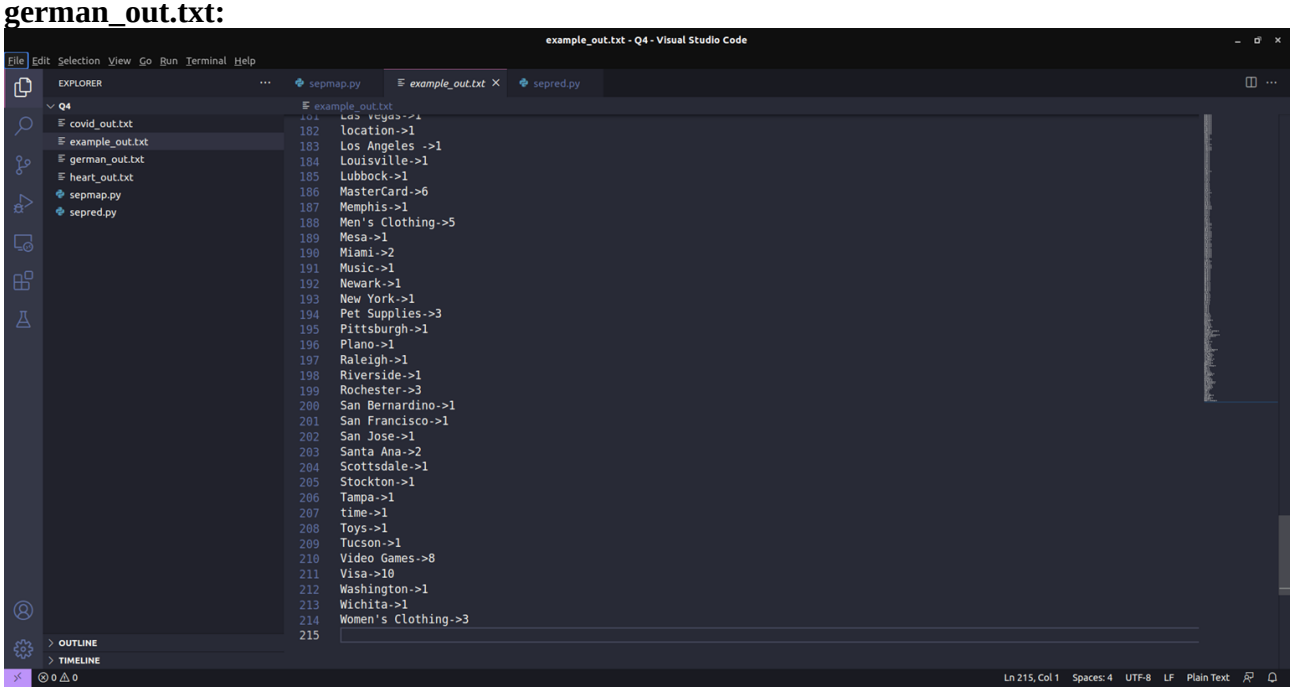



```
"Washington County->7
"Washoe County->4
"Wayne County->4
West Bank and Gaza->182
West Bengal->106
"Westchester County->7
Western Australia->208
Western Sahara->172
West Virginia->198
 WI"->34
"Williamson County->5
Wisconsin->198
Wuhan Evacuee->1
Wyoming->198
Xinjiang->246
Yamagata->119
Yamaguchi->119
Yamalo-Nenets Autonomous Okrug->115
Yamanashi->119
Yaroslavl Oblast->115
Yemen->167
"Yolo County->4
Yucatan->127
Yukon->182
Yunnan->246
Zabaykalsky Krai->115
Zacatecas->127
Zakarpattia Oblast->115
Zambia->190
Zaporizhia Oblast->115
Zeeland->69
Zhejiang->246
Zhytomyr Oblast->115
Zimbabwe->188
Zuid-Holland->69
```

example_out.txt:



german_out.txt:



5) Try to apply finding max value using map redeuce concept for the output of heart disease dataset, covid 19 dataset, example dataset, german credit dataset

#costmap.py (for heart disease dataset):

```
import fileinput

for line in fileinput.input():
    data = line.strip().split(",")

    if len(data) == 14:
        age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang, oldpeak, slope, ca, thal, target = data
        print("{}\t{}".format(sex, chol))
```

#costmap.py (for covid 19 dataset):

```
import fileinput

for line in fileinput.input():
    data = line.strip().split(",")

    if len(data) == 8:
        sno, observationdate, province, country, lastupdate, confirmed, deaths, recovered = data
        print("{}\t{}".format(observationdate, confirmed))
```

#costmap.py (for example dataset)

```
import fileinput

for line in fileinput.input():
    data = line.strip().split(",")

    if len(data) == 6:
        date, time, location, itemtype, amount, cardtype = data
        print("{}\t{}".format(itemtype, amount))
```

#costmap.py (for german credit dataset)

```
import fileinput

for line in fileinput.input():
    data = line.strip().split(",")

    if len(data) == 3:
        credibility, creditamount, durationofcredit = data
        print("{}\t{}".format(credibility, creditamount))
```

#costred.py:

```
import fileinput

max_val = 0
old_key = None

for line in fileinput.input():

    data = line.strip().split("\t")
```

```
    if len(data) != 2:
        continue

    current_key, current_value = data
    try:
        v = float(current_value)
    except ValueError:
        continue

    if old_key and (old_key != current_key):
        print("{}\t{}".format(old_key, max_val))
        old_key = current_key

        max_val = 0

    old_key = current_key
    if float(current_value) > float(max_val):
        max_val = float(current_value)

if old_key != None:
    print("{}\t{}".format(old_key, max_val))
```

Output:

```
PS D:\Code\CSE\Year 3\Sem 6\DSL\Week 5> `
>> cat data/heart_disease_data.csv | python Q5/costmap.py | sort | python Q5/costred.py > Q5/heart_out.txt
>> cat data/covid_19_data.csv | python Q5/costmap.py | sort | python Q5/costred.py > Q5/covid_out.txt
>> cat data/example.csv | python Q5/costmap.py | sort | python Q5/costred.py > Q5/example_out.txt
>> cat data/german_credit.csv | python Q5/costmap.py | sort | python Q5/costred.py > Q5/german_out.txt
PS D:\Code\CSE\Year 3\Sem 6\DSL\Week 5>
```

Heart_out.txt

```
0	564.0
1	353.0
```

covid_out.txt

```
1/22/2020	444.0
1/23/2020	444.0
1/24/2020	549.0
1/25/2020	761.0
1/26/2020	1058.0
```

German_out.txt

```
0	18424.0
1	15857.0
```

example_out.txt

```
Books	498.29
Cameras	485.71
Children's Clothing	364.53
Computers	288.32
Consumer Electronics	410.37
Crafts	489.93
DVDs	492.8
Garden	386.56
Health and Beauty	464.36
Men's Clothing	443.78
Music	213.64
Pet Supplies	431.73
Toys	13.79
Video Games	460.39
Women's Clothing	481.31
```

6) (Instructed to not do)
7) Write a map reduce program to count even or odd numbers in randomly generated natural numbers

#mapper.py:
```
import sys

for line in sys.stdin:
    words = line.strip().split()

    for word in words:
        num = int(word)
        if num % 2 == 0:
            print("%s\t%d"%("even", 1))
        else:
            print("%s\t%d"%("odd", 1))
```

#reduce.py
```
from operator import itemgetter
import sys

current_word = None
current_count = 0
word = None

for line in sys.stdin:
    try:
        word, count = line.strip().split('\t', 1)
        count = int(count)
    except ValueError:
        continue

    if current_word == word:
        current_count += count
    else:
        if current_word:
            print("%s\t%d"%(current_word, current_count))

        current_count = count
        current_word = word

if current_word == word:
    print("%s\t%d"%(current_word, current_count))
```

Output:

```
└──╼ $ cat Q7/rand_nums.txt | python Q7/mapper.py | sort | python Q7/reduce.py
even    48
odd     52
```