Map Reduce Examples using Hadoop Streaming

Task 1: Compute the average friends by age from the given dataset (fakefriends.csv in github). The data has 4 columns (id, name, age, number_of_friends)

```
0,Will,33,385
1,Jean-Luc,26,2
2,Hugh,55,221
3,Deanna,40,465
4,Quark,68,21
5,Weyoun,59,318
6,Gowron,37,220
7,Will,54,307
8,Jadzia,38,380
```

Output:

```
"32"    207
"33"    325
"34"    245
"35"    211
"36"    246
"37"    249
"38"    193
"39"    169
"40"    250
```

```
def mappr(self, _, line):
        (id, name, age, number) = line.split(",")
        yield (age, int(number))

def reducer(self, key, values):
        result = 0
        count = 0
        for i in values:
            result += i
            count += 1
        yield (key, (result/count))
```

Task 2: Compute the Min and Max temperatures by location from the given dataset 1080.csv in github. It has 8 columns out of which 3 are empty. (location, timestamp, feature, value, , ,E,)

```
GM000010962,18001227,PRCP,0,,,E,
EZE00100082,18001227,TMAX,47,,,E,
EZE00100082,18001227,TMIN,20,,,E,
ITE00100554,18001228,TMAX,35,,,E,
ITE00100554,18001228,TMIN,25,,,E,
GM000010962,18001228,PRCP,7,,,E,
EZE00100082,18001228,TMAX,43,,,E,
EZE00100082,18001228,TMIN,7,,,E,
```

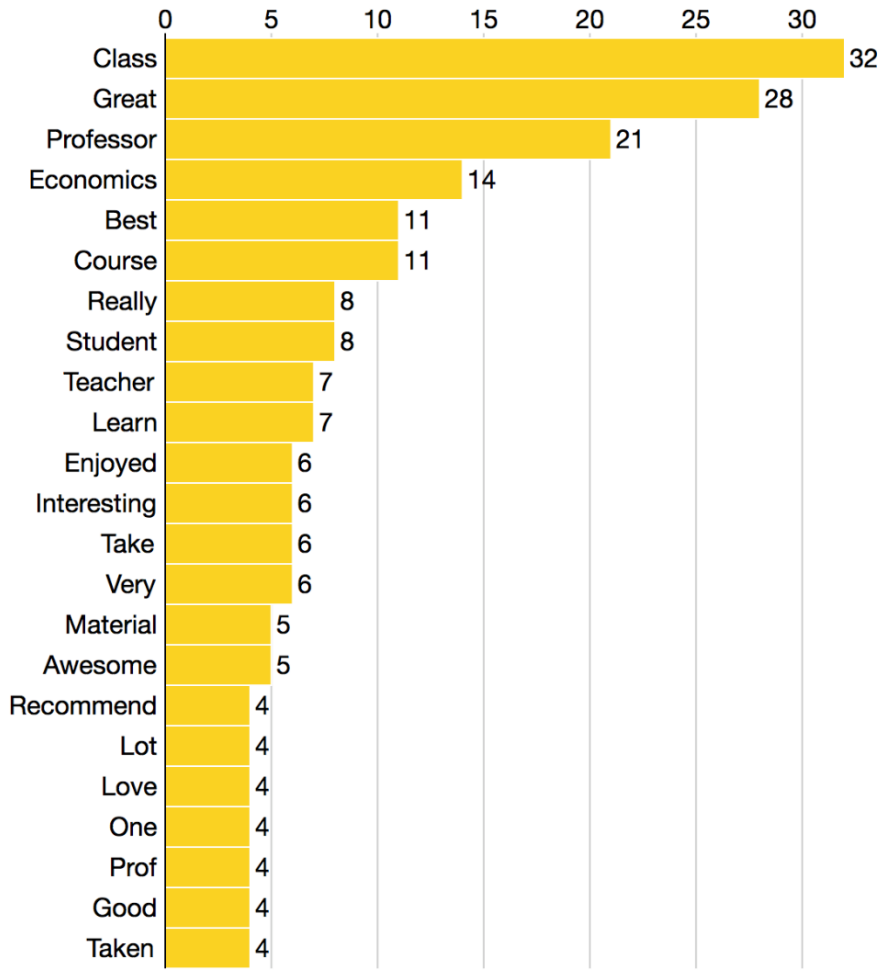
Output:

```
"EZE00100082"    "99"
"ITE00100554"    "98"
```

```
def mappr(self, _, line):
        (location, ts, feature, value, _, _, _, _) = line.split(",")
        if feature == 'TMAX':
            yield (location, value)
        else:
            pass


 def reducer(self, key, values):
        yield (key, max(values))
```

Task 3: Compute the Word frequency count in the sample text file (Book in github). Write the result into a new file and upload that to HDFS




```
def mappr(self, _, line):
        words = line.split()
        for i in words:
            yield (i, 1)
def reducer(self, key, values):
        yield (key, sum(values))
```

Task 4: Modify the above code using regular expressions

```
import re
reg_exp = re.compile(r"\b[A-Za-z]+\b")
def mappr(self, _, line):
```

```
        words = reg_exp.findall(line)
        for i in words:
            yield (i, 1)
```

Task 5: Sort the words by their frequency by using multiple MR steps