1.

Yes. I posted questions and got answers from Piazza about ReduceToList() and algorithm details.

No.

2.

I will use "Group…retaining…reducingTo=ReduceToList()" to group the data with the use of unigram. So the data structure will change into: (unigram, [(decade1, occurence1), (decade1, occurence1), …]. And then I will use ReplaceEach (by=getResult) function to get the final output. The getResult function here is a self-defined function, which uses (unigram, [(decade1, occurence1), (decade1, occurence1), …] as parameter. The function is like:

```
def getResult(line):
    unigram = line[0]
    eventlist = line[1]
    fg = 0
    bg = 0
    for event in eventlist:
        if event[0] == 1960:
            fg = event[1]
        else:
            bg = bg + event[1]
    return unigram, "fg="fg, "bg="bg
```

3.

Map:

Convert each line in dataset 2

Input: id, p1, p2, p3

Output: p1, id

P2, id

P3, id

Sort:

Sort the output of map phase by the product number

Reduce:

Aggregate the sorted data by product number, and only keep those product numbers with only one id

Input: p1, id
P2, id
P3, id

Output: p1, id1
P5, id9

Sort:

Sort the output from reduce phase by id

Reduce:

Aggregate the sorted data by id, and count the number of product number following the same id.

Input: p1, id1
P5, id9
P23, id9
P7, id13
P34, id13

Output: id1, 1
Id9, 2
Id13, 2

Combine:

Combine the output from reduce phase with the dataset1 by id. For those id numbers do not exist in the output file from reduce phase add 0.

Input1: id1, 1
Id9, 2
Id13, 2

Input2: id1, Lucy
Id2, Leo
Id3, Marlin

Output: id1, Lucy, 1
Id2, Leo, 0
Id3, Marlin, 0

…

Id9, Joe, 2

…

Id13, Jenny, 2

5.

The lines of python split the data by space, and convert each token into line. And then separate lines in odd number and lines in even number. And then match the lines in odd number and lines in even number with the same content, and also yield the length of the matched line.