Data Extraction:

```
Step 1: Connect to database for `movies` collection.
Step 2: Fetch date, publisher for every movie from database
Step 3:  year <- last 4 characters of date.
Step 4: publishers <- top 3 publishers.
Step 5: for publisher in publishers do
            yc_pair <- (year, publisher)
Step 6: Write the pair to file.
Step 7. Repeat step 2-6 till the end of query items.
```

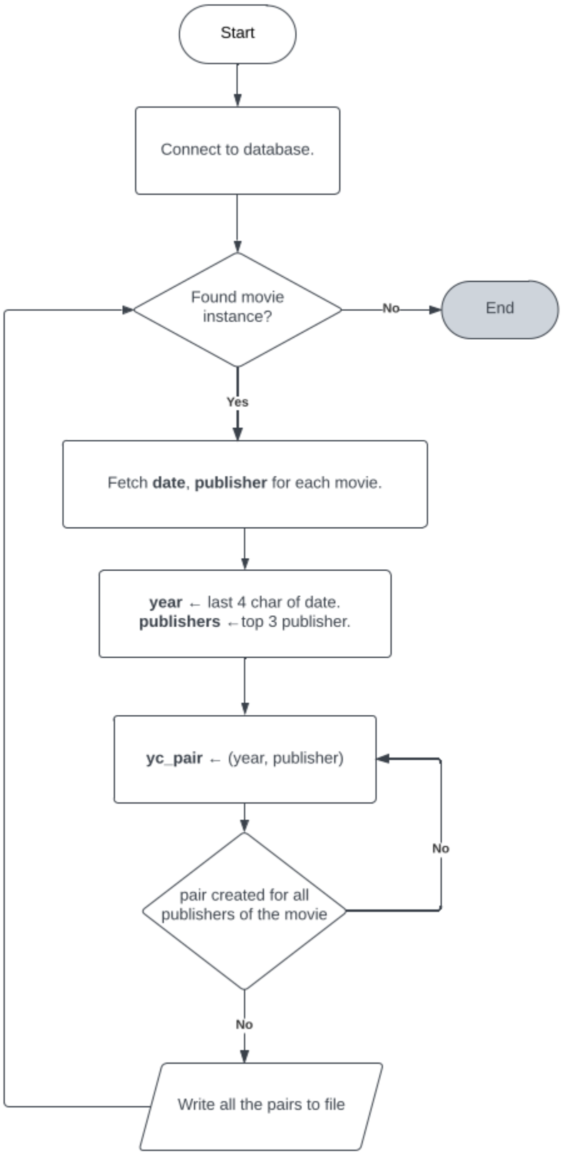
Start
Connect to database.
Found movie instance?
No
End
Yes
Fetch date, publisher for each movie.
year ← last 4 char of date.
publishers ←top 3 publisher.
yc_pair ← (year, publisher)
No
pair created for all publishers of the movie
No
Write all the pairs to file

Fig 1: Flowchart of data extraction

Data Count:

Step 1:. Read a single line from the file.
Step 2: Convert into tokens.
Step 3: Yield all tokens at once and set the count to 1.
Step 4: Join all the tokens to convert into a single string.
Step 5: Add the count to the string.
Step 6: Reducer counts the number of occurrences.
Step 7: Write every count to the file.
Step 8: Repeat step 1-7 till EOF.

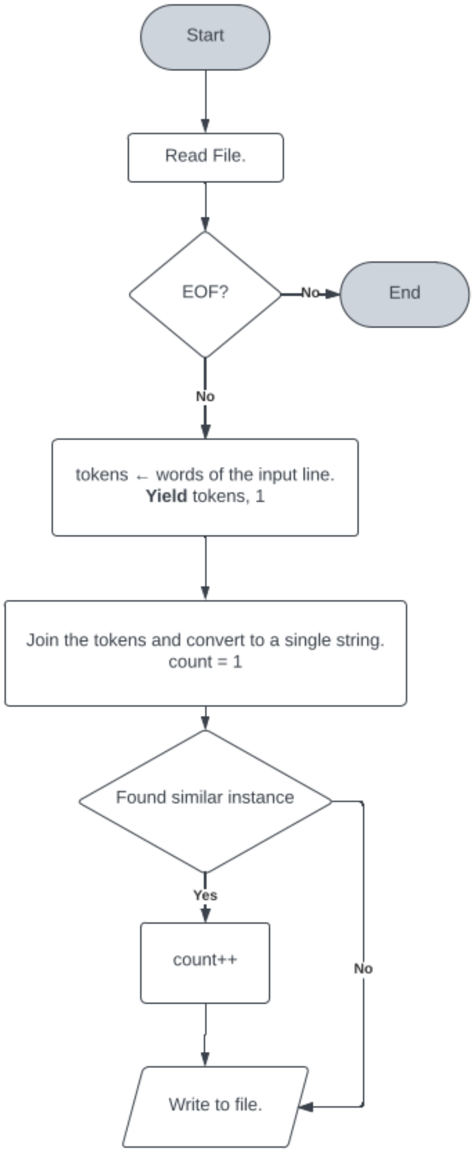

Fig 2: Flowchart of data count

MergeSort:

Step 1: Read all the data from files. Make a list of data.

Step 2: Declare two variables with 0, 0 as the count of the sorted array.

Step 3: Calculate mid using (left + right / 2). Make a left and a right array.

Step 4: Call the mergeSort function on the part (left, mid) and (mid+1, right).

Step 5: while left<right do step 4-6

Step 6: Call merge on the resulting array.

Step 7: Check left < right

Step 7.1: If true, then append the left array's item.

Step 7.2: If false, then append the right array's item.

Step 8: Step 9 continues till one or both the array is empty.

Step 9: If any one array is not empty yet, append all the items of the array to the output

Step 10: Write sorted array to file.

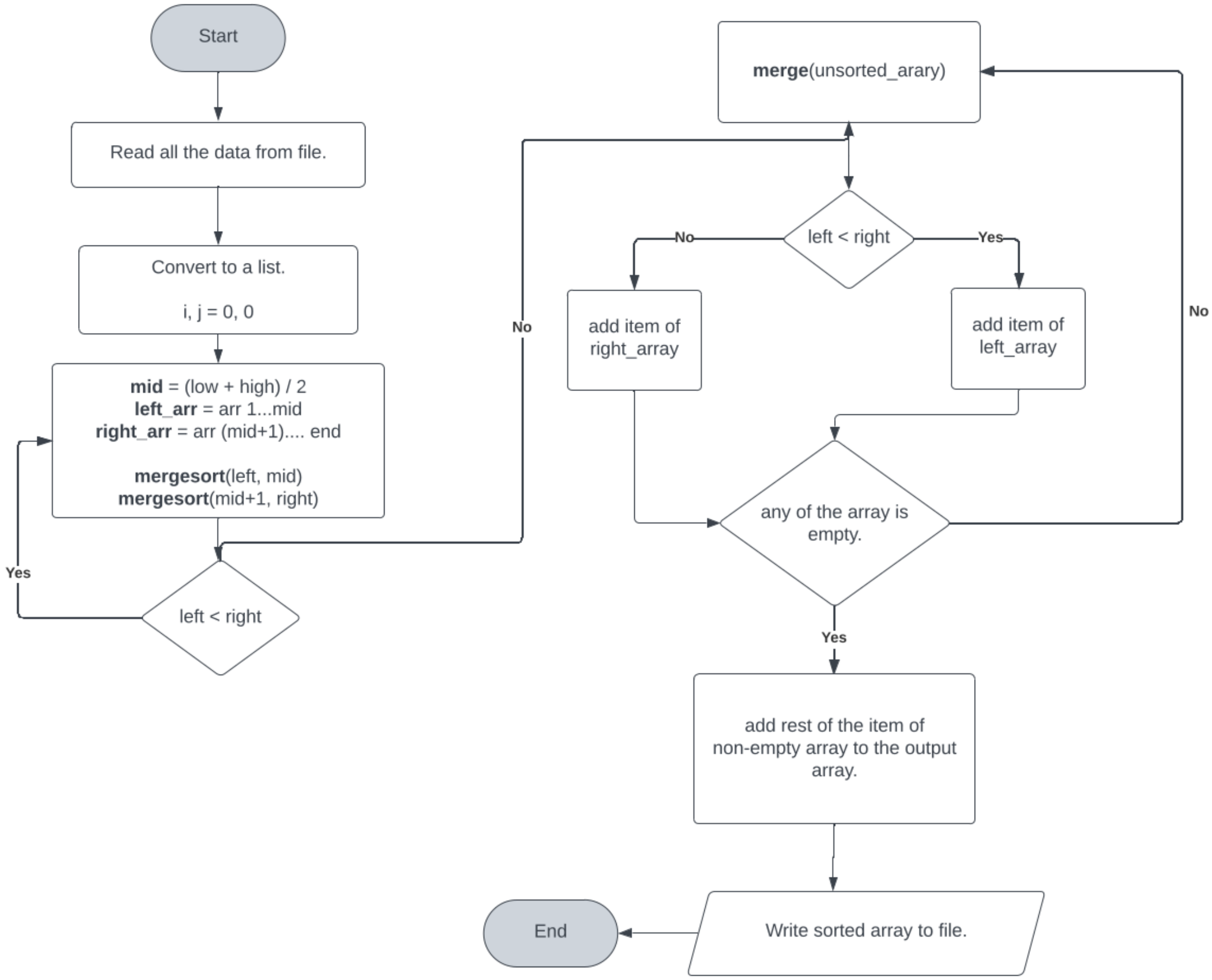

Fig 3: Flowchart of merge sort.

BucketSort:

Step 1: Read all the data from files.

Step 2: Make a list of data.

Step 3: Define an empty middle-man list named as mid_lst

Step 4: n <- total iterations required.

Step 5: Add n number of empty arrays to the array.

Step 6: Iterate over the data.

Step 6.1: Calculate val using (n * (yp_occur_count/100))

Step 6.2: Append this value to mid_lst's val index.

Step 7: Continue step-6 n times.

Step 8: for i = 1 to n do sort(buckets[i])

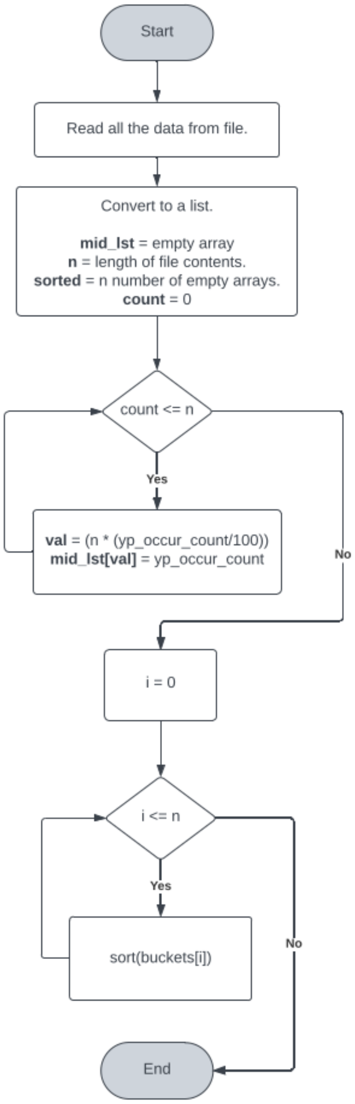
Start
Read all the data from file.
Convert to a list.
mid_lst = empty array
n = length of file contents.
sorted = n number of empty arrays.
count = 0
count <= n
Yes
val = (n * (yp_occur_count/100))
mid_lst[val] = yp_occur_count
No
i = 0
i <= n
Yes
sort(buckets[i])
No
End

Fig 4: Flowchart of bucket sort