Steps :

```
1. Put all ads in a map with key as group Id
2. For all ads in each groupId
        Compare all ads with each other, using 2-proportion z-interval test and
        store result of win count, lose count, no result count – for each ad.
3. For all ads in each groupId
        If (ad wins over all other ads)
            Add status for this ad as clear ‘Winner’
        If (ad loses over half or more ads it competes with)
            Add status for this ad as clear ‘Loser’
        else
            Add status for this ad as ‘No Result’
```

Pseudo Code:

Variables :

```
    Group_Map = Map <groupId, Array<Ad Object>>
    Ad = {adId, clicks, impressions}
    Ad_Result_Map = {adId, winCount, loseCount, noResultCount, groupSize, sufficiency}
    Result_list = {groupId, adId, resultStatus, remark}
```

Pseudo code :

```
1. Read line by line input from given file in json format
        Add Ad in array in Group_Map, against its group Id
        // We have map of all groupIds with all Ads in its value array
2. For each groupId in Group_Map {
        decide_winner_loser (groupId, ads_list);
   }
   // decide_winner_loser() function, explained below, will set values in Ad_Result_Map
   // i.e for each adId, count of how many ads it won against, lose against and was having
   // indeterminate result against.
3. For each row in input file {
        // to output the result in same sequence as input we are putting result with
        // respective adId in row file
        For the AdId in the row – get Result from ‘Ad_Result_Map’
        If (groupSize == 1) {
            Mark The Ad – “NO_RESULT”
            And – set remark – “NO_COMPETITOR”
        } else if (sufficiency ==  “insufficent”) {
            Mark The Ad – “NO_RESULT”
            And – set remark – “INSUFFICIENT_SAMPLE_DATA”
        } else if (winCount == groupSize -1) {
            // wins over all competitors
            Mark The Ad – “WINNER”
        } else if (loseCount >= ceil [groupSize/2]) {
            // loses over half or more competitors
            Mark The Ad – “LOSER”
        } else {
            Mark The Ad – “NO_RESULT”
        }
        Add the result <group Id, ad Id, status, remark> to Result_list
   }
4.  Print result list in output file
```

Function : decide_winner_loser (groupId , list_of_ads)

```
1.  // only single ad in the group – cannot be compared!, thus add status = no_result and
    // remark = no_competitor
    If  (list_of_ads.size == 1) {
            Add to - Ad_Result_Map –
            ‘status’ for this ad = ‘NO_RESULT’
            ‘remark’ = ‘NO_COMPETITOR’
    }
2.  for (i = 0 to AdsArray.size) {
            for (j = i+1 to AdsArray.size) {
                 // compare 2 ads using 2-proportion z test, compare_ads function
                // explained below.
                compare_ads(Ad[i], Ad[j], groupSize);
            }
    }
```

Fuction : compare_ads (ad1, ad2, groupSize) :

```
1.  check sufficiency of both sample data for 2 proportion z-interval (OPTIONAL)
            if (any 1 ad is insufficient)
                    // it cannot be compared with any other ad and thus we store result of
                    for
            // comparison of this ad with other and vice-versa as – No_Result i.e
            increase
            // counter of no_result.
        Store remark for insufficient ad in Ad_Result_Map as ‘INSIFFICIENT’
2. calculate 95% confidence interval for 2 ads – (lower_limit, upper_limit)
3. if (interval falls in positive side)
        //i.e lower_limit > 0 && upper_limit > 0
        ad1 Wins over ad2 with 95% confidence
4. if (interval falls in negative side)
        ad2 Wins over ad1 with 95% confidence
5. else
        its not determinate who can win.
   Update the win count, lose count, no_result count for both ads, as per results in
   Ad_Result_Map
```

Formula Used:

```
1. for sufficiency check for a ad’s sample data
     1.
     given : c = num_clicks, n = num_impressions, check = condition check value (differs like
     5, 10, 15)
     calculation :
         p = c/ n
         q = 1 – p
     result :
         true – if
             p*n >= check && q*n >= check
         false – otherwise.
     2.
     calculation:
         lower_bound = p - 3*(square_root(pq/n))
         upper_bound = p + 3*(square_root(pq/n))
     result:
         true – if range lies between 0 and 1.

2. finding 95% interval for 2 ads given clicks and impressions.
     given : ad1, ad2, confidence level = 95%
         c1 = clicks of ad1, n1 = impressions of ad1,
         c2 = clicks of ad2, n2 = impressions of ad2,
     calculation :
         p1 = c1/ n1,     p2 = c2/ n2
         q1 = 1 – p1,     q2 = 1 – p2

         interval_limit1 = point_estimate  -  critical_value * standard_error
         interval_limit2 = point estimate + critical_value * standard_error

         where,
               point_estimate =  p1 – p2
               critical_value = (from the standard table z value for 95% i.e ‘z for 0.05/2’) = 1.96
               standard_error = square_root(p1q1/n1 + p2q2/n2)

     result :
            ad1 WINS – if
                    interval_limit1 and interval_limit2 are +ve values
             ad2 LOSES

         ad2 WINS – if
                    interval_limit1 and interval_limit2 are -ve values
             ad1 LOSES

         NO_RESULT – otherwise
```