FISM INFORMATION

1. INPUT VALUES

SIZE_ARRAY: This parameter used to decide size of two dimension array

K_MIN: Minimum number of list contain set of values

L_ELEMENT: Number of elements must exist together is K_MIN list as reuired

2. ALGORITHM

STEP 1: Build dictionary with each key as value from input array and value of dictionary is a list which include all of list contain this value

STEP 2: With keys from key in dictionary in step 1. Create a combination from this key set with chosen value as L_ELEMENT

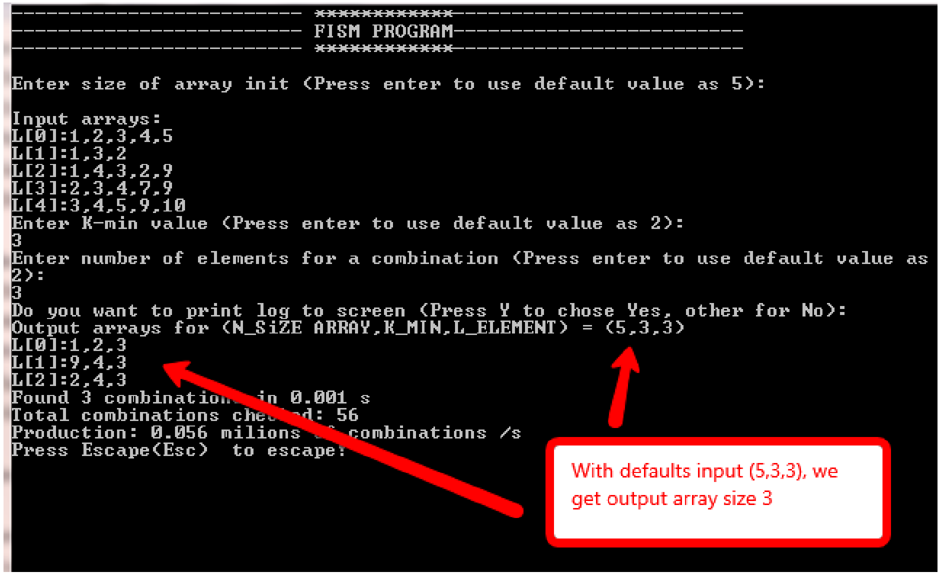
STEP 3: With each combination, we check by joining list contain values get out from a combination. If joined list return after joining all list reference by combination contain minimum K_MIN then it is a satified combination else move to next combination.

Pseudo-code:

```
Int[] FISM_Process(input,K,L)
{
        // Step1:
        Dictionary<int,List<int>> inputDictionary = BuildInputDictionary();
        // Step 2:
        For keys in inputDictionary.Keys do
        {
                BuildCombination(inputDictionary.Keys,L);
        }
        // Step 3
        For each combination do
        {
                List<int> joinedList = JoinAllListReferByValues(combination)
                If (joinedList.Count >= K_MIN) then
                {
                   AddCombinationToResult();
                }
        }
}
```

3. FEATURES AND SCREEN SHOTS

- Default input values (N,K,L) = (5,3,3)



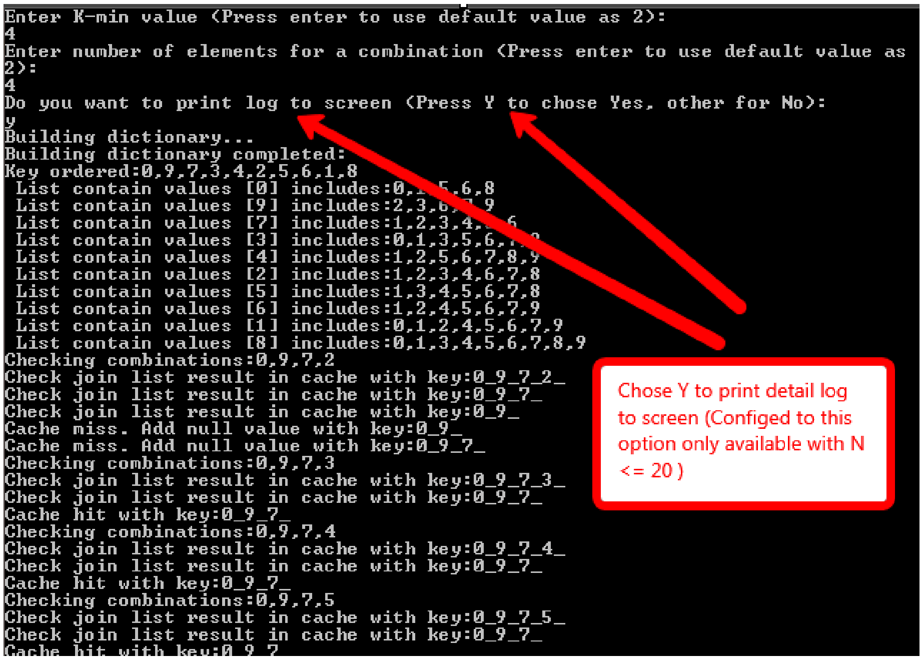
```
------------------------- ************-------------------------
------------------------- FISM PROGRAM-------------------------
------------------------- ************-------------------------

Enter size of array init (Press enter to use default value as 5):

Input arrays:
L[0]:1,2,3,4,5
L[1]:1,3,2
L[2]:1,4,3,2,9
L[3]:2,3,4,7,9
L[4]:3,4,5,9,10
Enter K-min value (Press enter to use default value as 2):
3
Enter number of elements for a combination (Press enter to use default value as
2):
3
Do you want to print log to screen (Press Y to chose Yes, other for No):
Output arrays for (N_SiZE ARRAY,K_MIN,L_ELEMENT) = (5,3,3)
L[0]:1,2,3
L[1]:9,4,3
L[2]:2,4,3
Found 3 combinations in 0.001 s
Total combinations checked: 56
Production: 0.056 milions of combinations /s
Press Escape(Esc)  to escape!
```

With defaults input (5,3,3), we get output array size 3

- Show details log of processing



```
Enter K-min value (Press enter to use default value as 2):
4
Enter number of elements for a combination (Press enter to use default value as
2):
4
Do you want to print log to screen (Press Y to chose Yes, other for No):
y
Building dictionary...
Building dictionary completed:
Key ordered:0,9,7,3,4,2,5,6,1,8
 List contain values [0] includes:0,1,5,6,8
 List contain values [9] includes:2,3,6,7,9
 List contain values [7] includes:1,2,3,4,5,6
 List contain values [3] includes:0,1,3,5,6,7,8
 List contain values [4] includes:1,2,5,6,7,8,9
 List contain values [2] includes:1,2,3,4,6,7,8
 List contain values [5] includes:1,3,4,5,6,7,8
 List contain values [6] includes:1,2,4,5,6,7,9
 List contain values [1] includes:0,1,2,4,5,6,7,9
 List contain values [8] includes:0,1,3,4,5,6,7,8,9
Checking combinations:0,9,7,2
Check join list result in cache with key:0_9_7_2_
Check join list result in cache with key:0_9_7_
Check join list result in cache with key:0_9_
Cache miss. Add null value with key:0_9_
Cache miss. Add null value with key:0_9_7_
Checking combinations:0,9,7,3
Check join list result in cache with key:0_9_7_3_
Check join list result in cache with key:0_9_7_
Cache hit with key:0_9_7_
Checking combinations:0,9,7,4
Check join list result in cache with key:0_9_7_4_
Check join list result in cache with key:0_9_7_
Cache hit with key:0_9_7_
Checking combinations:0,9,7,5
Check join list result in cache with key:0_9_7_5_
Check join list result in cache with key:0_9_7_
Cache hit with key:0_9_7_
```

Chose Y to print detail log to screen (Configed to this option only available with N <= 20)

4. PERFORMANCE TESTING

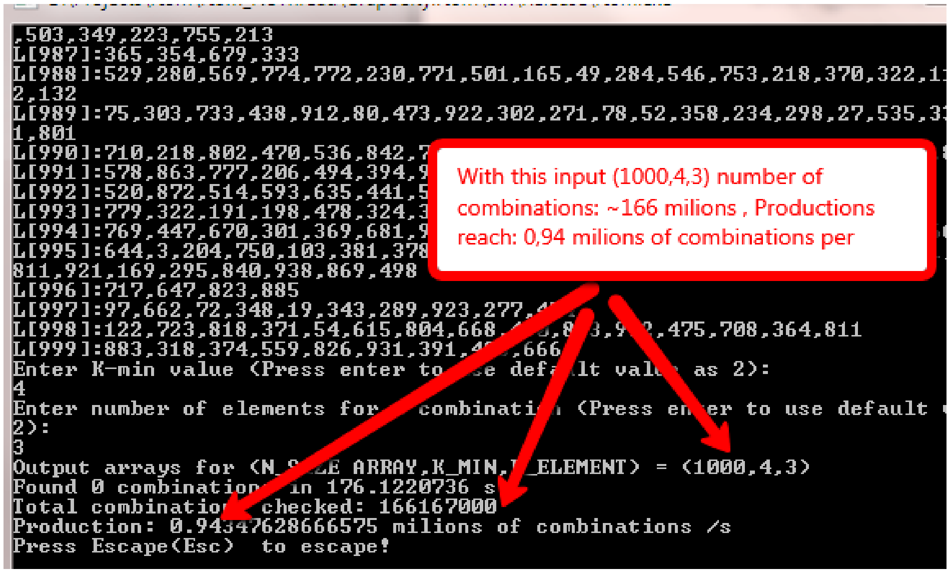
All tests placed on a system with Windows 7, core i5, 64 bit, 8 GB RAM, 2,5Ghz

Measure methods: Basically with this algorithm, we depend on two values that is SIZE_ARRAY and L_ELEMENT and this will effect to number of combinations generated. It is easy to drive this program to a combinatorial explosion which number of combination quickly to a very large number. Also for avoiding problems with storage combination I use a combination generator which allow generating combinations one by one so basically it is not limited to testing on large number, it is just trouble how many time we can reach to the end of the process.

Other information in testing progress:

CPU occuppied by this process: 25%

Memory occuppied by this process < 10M for normal actions (SIZE_ARRAY < 1000, number of combinations: 166 milions) (Maybe increased depend on size of input array)



```
,503,349,223,755,213
L[987]:365,354,679,333
L[988]:529,280,569,774,772,230,771,501,165,49,284,546,753,218,370,322,1
2,132
L[989]:75,303,733,438,912,80,473,922,302,271,78,52,358,234,298,27,535,3
1,801
L[990]:710,218,802,470,536,842,7
L[991]:578,863,777,206,494,394,9
L[992]:520,872,514,593,635,441,5
L[993]:779,322,191,198,478,324,3
L[994]:769,447,670,301,369,681,9
L[995]:644,3,204,750,103,381,378
811,921,169,295,840,938,869,498
L[996]:717,647,823,885
L[997]:97,662,72,348,19,343,289,923,277,4
L[998]:122,723,818,371,54,615,804,668,[illegible],8[illegible]3,9[illegible]2,475,708,364,811
L[999]:883,318,374,559,826,931,391,4[illegible],666
Enter K-min value (Press enter to use default value as 2):
4
Enter number of elements for combination (Press enter to use default
2):
3
Output arrays for (N_SIZE ARRAY,K_MIN,N_ELEMENT) = (1000,4,3)
Found 0 combinations in 176.1220736 s
Total combinations checked: 166167000
Production: 0.94347628666575 milions of combinations /s
Press Escape(Esc)  to escape!
```

With this input (1000,4,3) number of combinations: ~166 milions , Productions reach: 0,94 milions of combinations per