28May

C ++ Prgm : Given an array arr[] of size N and an integer K. The task is to find the last remaining element in the array after reducing the array.

|  |
| --- |
|  |
|  | #include<iostream> |
|  | using namespace std; |
|  |  |
|  | // A structure to store an element and its current count |
|  | struct eleCount |
|  | { |
|  | int e; // Element |
|  | int c; // Count |
|  | }; |
|  |  |
|  | // Prints elements with more than n/k occurrences in arr[] of |
|  | // size n. If there are no such elements, then it prints nothing. |
|  | void moreThanNdK(int arr[], int n, int k) |
|  | { |
|  | // k must be greater than 1 to get some output |
|  | if (k < 2) |
|  | return; |
|  |  |
|  | /\* Step 1: Create a temporary array (contains element |
|  | and count) of size k-1. Initialize count of all |
|  | elements as 0 \*/ |
|  | struct eleCount temp[k-1]; |
|  | for (int i=0; i<k-1; i++) |
|  | temp[i].c = 0; |
|  |  |
|  | /\* Step 2: Process all elements of input array \*/ |
|  | for (int i = 0; i < n; i++) |
|  | { |
|  | int j; |
|  |  |
|  | /\* If arr[i] is already present in |
|  | the element count array, then increment its count \*/ |
|  | for (j=0; j<k-1; j++) |
|  | { |
|  | if (temp[j].e == arr[i]) |
|  | { |
|  | temp[j].c += 1; |
|  | break; |
|  | } |
|  | } |
|  |  |
|  | /\* If arr[i] is not present in temp[] \*/ |
|  | if (j == k-1) |
|  | { |
|  | int l; |
|  |  |
|  | /\* If there is position available in temp[], then place |
|  | arr[i] in the first available position and set count as 1\*/ |
|  | for (l=0; l<k-1; l++) |
|  | { |
|  | if (temp[l].c == 0) |
|  | { |
|  | temp[l].e = arr[i]; |
|  | temp[l].c = 1; |
|  | break; |
|  | } |
|  | } |
|  |  |
|  | /\* If all the position in the temp[] are filled, then |
|  | decrease count of every element by 1 \*/ |
|  | if (l == k-1) |
|  | for (l=0; l<k; l++) |
|  | temp[l].c -= 1; |
|  | } |
|  | } |
|  |  |
|  | /\*Step 3: Check actual counts of potential candidates in temp[]\*/ |
|  | for (int i=0; i<k-1; i++) |
|  | { |
|  | // Calculate actual count of elements |
|  | int ac = 0; // actual count |
|  | for (int j=0; j<n; j++) |
|  | if (arr[j] == temp[i].e) |
|  | ac++; |
|  |  |
|  | // If actual count is more than n/k, then print it |
|  | if (ac > n/k) |
|  | cout << "Number:" << temp[i].e |
|  | << " Count:" << ac << endl; |
|  | } |
|  | } |
|  |  |
|  | /\* Driver program to test above function \*/ |
|  | int main() |
|  | { |
|  | cout << "First Test\n"; |
|  | int arr1[] = {4, 5, 6, 7, 8, 4, 4}; |
|  | int size = sizeof(arr1)/sizeof(arr1[0]); |
|  | int k = 3; |
|  | moreThanNdK(arr1, size, k); |
|  |  |
|  | cout << "\nSecond Test\n"; |
|  | int arr2[] = {4, 2, 2, 7}; |
|  | size = sizeof(arr2)/sizeof(arr2[0]); |
|  | k = 3; |
|  | moreThanNdK(arr2, size, k); |
|  |  |
|  | cout << "\nThird Test\n"; |
|  | int arr3[] = {2, 7, 2}; |
|  | size = sizeof(arr3)/sizeof(arr3[0]); |
|  | k = 2; |
|  | moreThanNdK(arr3, size, k); |
|  |  |
|  | cout << "\nFourth Test\n"; |
|  | int arr4[] = {2, 3, 3, 2}; |
|  | size = sizeof(arr4)/sizeof(arr4[0]); |
|  | k = 3; |
|  | moreThanNdK(arr4, size, k); |
|  |  |
|  | return 0; |
|  | } |