

C-3.17

|  |
| --- |
| // Java Code to find elements whose  // frequency yis more than n/k  import java.util.\*;    public class Main  {      public static void morethanNdK(int a[], int n, int k)      {          int x = n / k;          // Hash map initialization          HashMap<Integer, Integer> y = new HashMap<>();            // count the frequency of each element.          for (int i = 0; i < n; i++)          {              // is element doesn't exist in hash table              if (!y.containsKey(a[i]))                  y.put(a[i], 1);                  // if lement does exist in the hash table              else              {                  int count = y.get(a[i]);                  y.put(a[i], count + 1);              }          }            // iterate over each element in the hash table          // and check their frequency, if it is more than          // n/k, print it.          for (Map.Entry m : y.entrySet())          {              Integer temp = (Integer)m.getValue();              if (temp > x)                  System.out.println(m.getKey());          }      }        // Driver Code      public static void main(String[] args)      {            int a[] = new int[] { 1, 1, 2, 2, 3, 5, 4,                                2, 2, 3, 1, 1, 1 };          int n = 12;          int k = 4;          morethanNdK(a, n, k);      }  } |

**Output**

1

2

C-3.18

|  |
| --- |
| // Java program to Merge an array of  // size n into another array of size m + n    class MergeArrays {      /\* Function to move m         elements at the end of array       \* mPlusN[] \*/      void moveToEnd(int mPlusN[], int size)      {          int i, j = size - 1;          for (i = size - 1; i >= 0; i--) {              if (mPlusN[i] != -1) {                  mPlusN[j] = mPlusN[i];                  j--;              }          }      }        /\* Merges array N[] of         size n into array mPlusN[]         of size m+n\*/      void merge(int mPlusN[], int N[], int m, int n)      {          int i = n;            /\* Current index of i/p part of mPlusN[]\*/          int j = 0;            /\* Current index of N[]\*/          int k = 0;            /\* Current index of output mPlusN[]\*/          while (k < (m + n))          {              /\* Take an element from mPlusN[] if              a) value of the picked element is smaller and we              have not reached end of it b) We have reached              end of N[] \*/              if ((i < (m + n) && mPlusN[i] <= N[j])                  || (j == n)) {                  mPlusN[k] = mPlusN[i];                  k++;                  i++;              }              else // Otherwise take element from N[]              {                  mPlusN[k] = N[j];                  k++;                  j++;              }          }      }        /\* Utility that prints out an array on a line \*/      void printArray(int arr[], int size)      {          int i;          for (i = 0; i < size; i++)              System.out.print(arr[i] + " ");            System.out.println("");      }        // Driver Code      public static void main(String[] args)      {          MergeArrays mergearray = new MergeArrays();            /\* Initialize arrays \*/          int mPlusN[] = { 2, 8, -1, -1, -1, 13, -1, 15, 20 };          int N[] = { 5, 7, 9, 25 };          int n = N.length;          int m = mPlusN.length - n;            /\*Move the m elements at the end of mPlusN\*/          mergearray.moveToEnd(mPlusN, m + n);            /\*Merge N[] into mPlusN[] \*/          mergearray.merge(mPlusN, N, m, n);            /\* Print the resultant mPlusN \*/          mergearray.printArray(mPlusN, m + n);      }  }    // This code has been contributed by Mayank Jaiswal |

**Output**

2 5 7 8 9 13 15 20 25