**Extendible Hashing :**

package exhashing;

import java.util.ArrayList;

import java.util.\* ;

public class Exhashing {

static List<int[]> myList = new ArrayList<int[]>();

public static void main(String[] args) {

int i;

Scanner sc=new Scanner(System.in);

for(i=0;i<4;i++)

{

int arr[]=new int[8] ;

arr[0]=i ;

arr[1]=2 ;

arr[2]=0 ;

arr[3]=0 ;

arr[4]=1 ;

myList.add(arr) ;

}

int n,x,lvl=2,y,z,k=1,j=2,l,t ;

boolean f ;

System.out.println("Enter the the no. of keys: ");

n=sc.nextInt() ;

for(t=1;t<=n;t++)

{

x=sc.nextInt() ;

z=x%16 ;

f=true ;

j=2 ;

while(f)

{

y=bin(z,j) ;

for(i=0;i<myList.size();i++)

{

int arr[]=myList.get(i);

if(arr[4]==-1)

continue ;

if(arr[0]==y)

{

if(arr[2]<3)

{

arr[arr[2]+5]=x ;

arr[2]++ ;

f=false ;

break ;

}

else if(arr[2]==3 && arr[1]<lvl)

{

for(l=5;l<=7;l++)

func(arr[l],lvl+1);

arr[4]=-1 ;

break ;

}

else if(arr[2]==3)

{

arr[4]=-1 ;

for(l=0;l<4;l++)

{

int arr1[]=new int[8] ;

arr1[0]=l ;

arr1[1]=lvl+1 ;

arr1[2]=0 ;

arr1[3]=l ;

arr1[4]=1 ;

myList.add(arr1) ;

int arr2[]=new int[8] ;

arr2[0]=l+ (int)Math.pow(2,lvl) ;

arr2[1]=lvl+1 ;

arr2[2]=0 ;

arr2[3]=l ;

arr2[4]=1 ;

myList.add(arr2) ;

}

//System.out.println("yo "+myList.size());

// display() ;

for(l=5;l<=7;l++)

func(arr[l],lvl+1);

lvl++ ;

//System.out.println("yo2 "+4817%16);

break ;

}

}

}

j++ ;

}

display() ;

}

}

public static void display()

{

int i,j ;

for(i=0;i<myList.size();i++)

{

int arr[]=myList.get(i) ;

System.out.print("\n"+arr[0]+": " ) ;

if(arr[4]!=-1)

for(j=5;j<8;j++)

System.out.print(arr[j]+" " ) ;

}

System.out.print("\n---------------------------------------------\n");

}

public static void func(int x,int lvl)

{

int i,l,y ;

int z=x%16 ;

boolean f=true ;

int j=2 ;

while(f)

{

y=bin(z,j+1) ;

for(i=0;i<myList.size();i++)

{

int arr[]=myList.get(i);

if(arr[4]==-1)

continue ;

if(arr[0]==y)

{

if(arr[2]<3)

{

arr[arr[2]+5]=x ;

arr[2]++ ;

//display() ;

f=false ;

break ;

}

}

}

j++ ;

}

}

public static int bin(int n,int l)

{

int x=(int) Math.pow(2,l);

return n%x ;

}

}

**OUTPUT:**

Enter the the no. of keys:

13

4068

0: 4068 0 0

1: 0 0 0

2: 0 0 0

3: 0 0 0

---------------------------------------------

1752

0: 4068 1752 0

1: 0 0 0

2: 0 0 0

3: 0 0 0

---------------------------------------------

3429

0: 4068 1752 0

1: 3429 0 0

2: 0 0 0

3: 0 0 0

---------------------------------------------

2130

0: 4068 1752 0

1: 3429 0 0

2: 2130 0 0

3: 0 0 0

---------------------------------------------

2854

0: 4068 1752 0

1: 3429 0 0

2: 2130 2854 0

3: 0 0 0

---------------------------------------------

1591

0: 4068 1752 0

1: 3429 0 0

2: 2130 2854 0

3: 1591 0 0

---------------------------------------------

2203

0: 4068 1752 0

1: 3429 0 0

2: 2130 2854 0

3: 1591 2203 0

---------------------------------------------

1423

0: 4068 1752 0

1: 3429 0 0

2: 2130 2854 0

3: 1591 2203 1423

---------------------------------------------

3017

0: 4068 1752 0

1: 3429 3017 0

2: 2130 2854 0

3: 1591 2203 1423

---------------------------------------------

2333

0: 4068 1752 0

1: 3429 3017 2333

2: 2130 2854 0

3: 1591 2203 1423

---------------------------------------------

3923

0: 4068 1752 0

1: 3429 3017 2333

2: 2130 2854 0

3: 2203 3923 0

4: 0 0 0

5: 0 0 0

6: 0 0 0

7: 1591 1423 0

---------------------------------------------

4817

0: 4068 1752 0

1: 3017 4817 0

2: 2130 2854 0

3: 2203 3923 0

4: 0 0 0

5: 3429 2333 0

6: 0 0 0

7: 1591 1423 0

---------------------------------------------

4876

0: 4068 1752 4876

1: 3017 4817 0

2: 2130 2854 0

3: 2203 3923 0

4: 0 0 0

5: 3429 2333 0

6: 0 0 0

7: 1591 1423 0

---------------------------------------------