2]

public class Main {

public static void main(String[] args) {

int A\_dice[] = {1, 2, 3, 4, 5, 6};

int B\_dice[] = {1, 2, 3, 4, 5, 6};

int Sum\_old[][] = new int[A\_dice.length][B\_dice.length];

int count = 0;

int temp = 0;

for (int i = 0; i < A\_dice.length; i++) {

for (int j = 0; j < B\_dice.length; j++) {

count++;

Sum\_old[i][j] = A\_dice[i] + B\_dice[j];

}

}

int frequency\_old[] = new int[13];

for (int row\_old[] : Sum\_old) {

for (int ele\_old : row\_old) {

frequency\_old[ele\_old]++;

}

}

int flag=0;

int o\_count\_A=0;

int e\_count\_A=0;

int new\_A[] = {1, 2, 3, 2, 3, 4};

for(int i=0;i<6;i++)

{

if(new\_A[i]>4)

{

flag=1;

break;

}

if(new\_A[i]%2==0)

{

e\_count\_A++;

}

if(new\_A[i]%2!=0)

{

o\_count\_A++;

}

}

int new\_B[] = new int[6];

new\_B[0] = 1;

new\_B[5] = 8;

//Assuming odd\_digit and even\_digit frequency to be same as in normal dices and consecutive sides have it's digit's difference as 1 in the 4 spots left in B

new\_B[1]=3;

new\_B[2]=4;

new\_B[3]=5;

new\_B[4]=6;

//there are 2 more combinations for these elements but got this satisfying by Brute-Force method in 1st attempt itself

int o\_count\_B=0;

int e\_count\_B=0;

for(int i=0;i<6;i++)

{

if(new\_B[i]%2==0)

{

e\_count\_B++;

}

if(new\_B[i]%2!=0)

{

o\_count\_B++;

}

}

if(flag==1)

{

System.out.println("A has element greater than 4 which breaks condiition");

}

if(o\_count\_A==e\_count\_A&&o\_count\_B==e\_count\_B&&flag==0)

{

System.out.println("conditions met proceed to frequncy check!");

}

else{

System.out.println("conditions not met hence failed!");

System.exit(0);

}

if(checkFrequency(new\_A, new\_B,frequency\_old))

{

System.out.println(" new dice A:");

for(int i=0;i<6;i++)

{

System.out.println(new\_A[i]);

}

System.out.println(" new dice B:");

for(int i=0;i<6;i++)

{

System.out.println(new\_B[i]);

}

}

else

{

System.out.println("new Dices do not have the same frequency distribution as old.");

}

}

public static boolean checkFrequency(int []new\_A,int []new\_B,int []frequency\_old)

{

int Sum\_new[][] = new int[new\_A.length][new\_B.length];

for (int i = 0; i < new\_A.length; i++)

{

for (int j = 0; j < new\_B.length; j++)

{

Sum\_new[i][j] = new\_A[i] + new\_B[j];

}

}

int frequency\_new[] = new int[13];

for (int row\_new[] : Sum\_new)

{

for (int ele\_new : row\_new)

{

frequency\_new[ele\_new]++;

}

}

for(int i=2;i<=12;i++)

{

if(frequency\_old[i]!=frequency\_new[i])

{

return false;

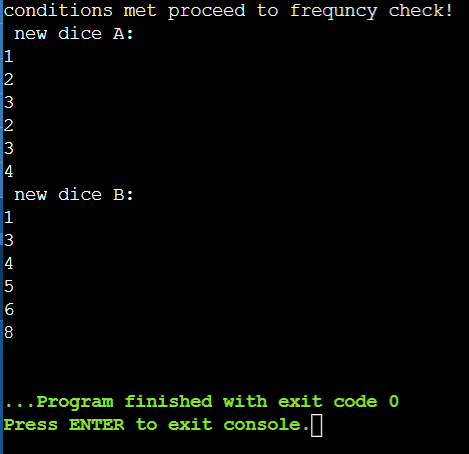
}

}

return true;

}

}



The corner cases sum of 1 and 12 was took care with {1,4} in A and{1,8} in B ,the rest just had to be consecutive elements in all 4 spots to have a difference by 1 and to maintain odd\_frequency =even\_frequency throughout the arrays.