**Prime Checker**

**import** java.io.\*;

**import** java.util.\*;

**import** java.text.\*;

**import** java.math.\*;

**import** java.util.regex.\*;

**import** java.lang.reflect.\*;

**import** **static** java.lang.System.in;

**class** Prime{

    ArrayList<Integer> arr=**new** ArrayList<Integer>(5);

**public** **void** checkPrime(**int**...n1) {

**int** flag=0;

**for**(**int** i:n1) {

**if**(java.math.BigInteger.valueOf(i).isProbablePrime(1)) {

**if**(!arr.contains(i)) {

                    arr.add(i);

                }

            }

        }

        Iterator itr=arr.iterator();

**while**(itr.hasNext()) {

            System.out.print(itr.next()+" ");

        }

        System.out.println();

    }

}

**public** **class** Solution {

**public** **static** **void** main(String[] args) {

**try**{

        BufferedReader br=**new** BufferedReader(**new** InputStreamReader(in));

**int** n1=Integer.parseInt(br.readLine());

**int** n2=Integer.parseInt(br.readLine());

**int** n3=Integer.parseInt(br.readLine());

**int** n4=Integer.parseInt(br.readLine());

**int** n5=Integer.parseInt(br.readLine());

        Prime ob=**new** Prime();

        ob.checkPrime(n1);

        ob.checkPrime(n1,n2);

        ob.checkPrime(n1,n2,n3);

        ob.checkPrime(n1,n2,n3,n4,n5);

        Method[] methods=Prime.**class**.getDeclaredMethods();

        Set<String> set=**new** HashSet<>();

**boolean** overload=**false**;

**for**(**int** i=0;i<methods.length;i++)

        {

**if**(set.contains(methods[i].getName()))

            {

                overload=**true**;

**break**;

            }

            set.add(methods[i].getName());

        }

**if**(overload)

        {

**throw** **new** Exception("Overloading not allowed");

        }

        }

**catch**(Exception e)

        {

            System.out.println(e);

        }

    }

}