```
//PriuyankaaBalaji
//Intermediate -Patolli

import java.io.File;
import java.io.FileNotFoundException;
import java.util.*;
public class PriuyankaaBalaji_INTc4Patolli {

    public static boolean isPrime(int num)
       {
          if(num == 1) return false;
          int check =0;
          for(int i=2; i<= num/2; i++)
          {
             check = num%i;
             if(check == 0)
                return false;
          }
          return true;
       }

    public static boolean isPerfectSquare(int num)
       {
        int x = (int) Math.sqrt(num);
        if(Math.pow(x,2) == num && num>4) return true;
        return false;
       }

    public static void main(String args[]) throws FileNotFoundException
    {
        Scanner sc = new Scanner(new File("4int_testdata"));
        while (sc.hasNextLine())
        {
            String input = sc.nextLine();
        String tokens[] = input.split(" ");
        int[] tokenint = new int[tokens.length];
        for(int i=0; i<tokens.length; i++)
        {
            tokenint[i] = Integer.parseInt(tokens[i]);
        }
        int[] playerIndexes = new int[3];
        int j=0;
        for(int i=3; i<=5; i++)
        {
            playerIndexes[j] = tokenint[i];
            j++;
        }
        int[] locations = new int[53];

        locations[playerIndexes[0]]=1;
        locations[playerIndexes[1]]=1;
        locations[playerIndexes[2]]=1;
        locations[tokenint[0]]=1;
        locations[tokenint[1]]=1;
        locations[tokenint[2]]=1;

        int rollindex =7;
        for(int k=0; k<(tokenint[6]); k++) {
        Arrays.sort(playerIndexes);

        int newIndex = playerIndexes[0]+tokenint[rollindex];
        if(newIndex <=52 && locations[newIndex] == 0 )
        {
            if(newIndex == 52)
            {
                playerIndexes[0]= 52;
            }
            else if(isPrime(newIndex))
            {
```

```
    boolean check = false;
    int i =1;
    while(check == false && i<7)
    {
        if(locations[newIndex+i] == 1)
        {
            locations[newIndex+i-1] = 1;
            check = true;
            locations[playerIndexes[0]] =0;
            playerIndexes[0] = newIndex+i-1;
        }
        i++;
    }
    if(check == false && ((newIndex + 6 ) <= 52))
    {
        if ((newIndex + 6) == 52)
        {
            locations[playerIndexes[0]] =0;
            playerIndexes[0] = newIndex+6;
        }
        else
        {
            locations[newIndex+6] = 1;
            locations[playerIndexes[0]] =0;
            playerIndexes[0] = newIndex+6;
        }
    }
    else if(check == false && ((newIndex + 6 ) > 52))
    {
        locations[newIndex] = 1;
        locations[playerIndexes[0]] =0;
        playerIndexes[0] = newIndex;
    }

}
else if(isPerfectSquare(newIndex))
{
    boolean check = false;
    int i =1;
    while(check == false && i<7)
    {
        if(locations[newIndex-i] == 1)
        {
            locations[newIndex-i+1] = 1;
            check = true;
            locations[playerIndexes[0]] =0;
            playerIndexes[0] = newIndex-i+1;
        }
        i++;
    }
    if(check == false)
    {
        locations[newIndex-6] = 1;
        locations[playerIndexes[0]] =0;
        playerIndexes[0] = newIndex-6;
    }
}
else
{
    int[] cross = {7,12,17,22,27,35,40,45,50};
    int diceValue = tokenint[rollindex];
    boolean hasCrossed = false;
    int i=0;
    while(!hasCrossed && i< cross.length)
    {
        if(playerIndexes[0]<= (cross[i]-1) && (newIndex)>= (cross[i]+1))
        {
            hasCrossed = true;
        }
```

```
                    i++;
                }
                if(!hasCrossed)
                {
                    locations[newIndex] = 1;
                    locations[playerIndexes[0]] =0;
                    playerIndexes[0] = newIndex;
                }
                else
                {
                    int a =0;
                    boolean check = false;
                    while((playerIndexes[0]+a)<=newIndex && !check)
                    {
                        if((locations[playerIndexes[0]+a]== 0) &&
                        ((playerIndexes[0]+a)%diceValue == 0))
                        {
                            locations[playerIndexes[0]+a] = 1;
                            check = true;
                            locations[playerIndexes[0]] =0;
                            playerIndexes[0] = playerIndexes[0]+a;
                        }
                        a++;
                    }

                }
            }
        }

        rollindex++;
    }
        Arrays.parallelSort(playerIndexes);
        for(int i=0; i<playerIndexes.length; i++)
        {
            if(playerIndexes[i]!= 52) System.out.print(playerIndexes[i]+" ");
        }
        System.out.println();
    }


    }

}
```