## 1. naloga

```
import java.util.*;

public class Prva {

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        int idx = 0;
        int lastClen = 1;
        boolean foundIt = false;
        while (sc.hasNextInt()) {
            int clen = sc.nextInt();
            if (clen % lastClen != 0) {
                System.out.println(idx);
                foundIt = true;
                break;
            }
            lastClen = clen;
            idx++;
        }
        if (!foundIt)
            System.out.println(idx);
    }
}
```

## 2. naloga

```
import java.util.*;

public class Druga {

    public static int zadnjaVrsticaZLocilom(char[][] krizanka) {
        int idxTarget = -1;
        for (int i = 0; i < krizanka.length; i++) {
            for (int j = 0; j < krizanka[i].length; j++) {
                if (krizanka[i][j] == '-')
                    idxTarget = i;
            }
        }
        return idxTarget;
    }

    public static char[] ktaBeseda(char[][] krizanka, int stolpec, int k) {
        StringBuilder sb = new StringBuilder();
        int stBesede = 1;
        boolean foundIt = false;
        for (int j = 0; j < krizanka.length; j++) {
            if (stBesede == k)
                foundIt = true;
            if (krizanka[j][stolpec] == '-')
                stBesede++;
            else if (stBesede == k)
                sb.append(krizanka[j][stolpec]);
        }
        if (stBesede == k && !foundIt && sb.length() == 0)
            return new char[] {};
        return !foundIt && sb.length() == 0 ? null : sb.toString().toCharArray();
    }
}
```

## 3. naloga

```
import java.util.*;

public class Tretja {

    public static abstract class Ukaz {

        public abstract int getBilanca();
        public abstract void izvedi(int[] stolpi);
```

```
    public static int bilanca(Ukaz[] ukazi) {
        int result = 0;
        for (Ukaz ukaz : ukazi)
            result += ukaz.getBilanca();
        return result;
    }

    /** [Zaporedje] */

    private static class ZaporedjeUkazov extends Ukaz {
        private Ukaz prvi;
        private Ukaz drugi;

        public ZaporedjeUkazov(Ukaz prvi, Ukaz drugi) {
            this.prvi = prvi;
            this.drugi = drugi;
        }

        @Override
        public void izvedi(int[] stolpi) {
            prvi.izvedi(stolpi);
            drugi.izvedi(stolpi);
        }

        @Override
        public int getBilanca() {
            return prvi.getBilanca() + drugi.getBilanca();
        }

        @Override
        public String toString() {
            return String.format("[%s, %s]", prvi.toString(), drugi.toString());
        }
    }

    public Ukaz zaporedje(Ukaz drugi) {
        return new ZaporedjeUkazov(this, drugi);
    }
}

public static class Postavi extends Ukaz {
    private int kam;

    public Postavi(int kam) {
        this.kam = kam;
    }

    @Override
    public int getBilanca() {
        return 1;
    }

    @Override
    public void izvedi(int[] stolpi) {
        if (this.kam >= 0 && this.kam < stolpi.length)
            stolpi[kam]++;
    }

    @Override
    public String toString() {
        return String.format("+%d", this.kam);
    }
}

public static class Odvzemi extends Ukaz {
    private int odkod;

    public Odvzemi(int odkod) {
        this.odkod = odkod;
    }

    @Override
    public int getBilanca() {
        return -1;
    }

    public void izvedi(int[] stolpi) {
```

```
            if (this.odkod >= 0 && this.odkod < stolpi.length) {
                if (stolpi[odkod] > 0)
                    stolpi[odkod]--;
            }
        }

        @Override
        public String toString() {
            return String.format("-%d", this.odkod);
        }
    }
}
```

## 4. naloga

```
import java.util.*;

public class Cetrta {

    private static void glasuj(Map<String, Integer> skrinjica, String stranka) {
        if (skrinjica.containsKey(stranka)) {
            int stGlasov = skrinjica.get(stranka);
            skrinjica.put(stranka, (stGlasov + 1));
        } else
            skrinjica.put(stranka, 1);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        Map<String, Integer> primarniGlasovi = new HashMap<>();
        Map<String, Integer> sekundarniGlasovi = new HashMap<>();

        TreeSet<String> stranke = new TreeSet<>(
            (p, q) -> {
                int stpp = primarniGlasovi.get(p) == null ? 0 : primarniGlasovi.get(p);
                int stpq = primarniGlasovi.get(q) == null ? 0 : primarniGlasovi.get(q);
                int stsp = sekundarniGlasovi.get(p) == null ? 0 : sekundarniGlasovi.get(p);
                int stsq = sekundarniGlasovi.get(q) == null ? 0 : sekundarniGlasovi.get(q);
                // primarni DESC
                // sekundarni DESC
                if (stpp != stpq)
                    return (stpq - stpp);
                if (stsp != stsq)
                    return (stsq - stsp);
                return p.compareTo(q);
            }
        );

        int n = sc.nextInt();
        for (int i = 0; i < n; i++) {
            String primarnaStranka = sc.next();
            String sekundarnaStranka = sc.next();

            glasuj(primarniGlasovi, primarnaStranka);
            glasuj(sekundarniGlasovi, sekundarnaStranka);
        }

        stranke.addAll(primarniGlasovi.keySet());
        stranke.addAll(sekundarniGlasovi.keySet());

        System.out.println(stranke.toString());
    }
}
```