Java-题目描述：

对一个数据 a 进行分类，分类方法为：此数据 a（四个字节大小）的四个字节相加对一个给定的值 b 取模，如果得到的结果小于一个给定的值 c，则数据 a 为有效类型，其类型为取模的值；如果得到的结果大于或者等于 c，则数据 a 为无效类型。

比如一个数据 a=0x01010101，b=3，按照分类方法计算

（0x01+0x01+0x01+0x01）%3=1，所以如果 c=2，则此 a 为有效类型，其类型为 1，如果 c=1，则此 a 为无效类型；

又比如一个数据 a=0x01010103，b=3，按照分类方法计算

（0x01+0x01+0x01+0x03）%3=0，所以如果 c=2，则此 a 为有效类型，其类型为 0，如果 c=0，则此 a 为无效类型。

输入 12 个数据，第一个数据为 c，第二个数据为 b，剩余 10 个数据为需要分类的数据，请找到有效类型中包含数据最多的类型，并输出该类型含有多少个数据。

输入描述：

输入 12 个数据，用空格分隔，第一个数据为 c，第二个数据为 b，剩余 10 个数据为需要分类的数据。

输出描述：

输出最多数据的有效类型有多少个数据。

补充说明：

示例 1

输入：

```
3 4 256 257 258 259 260 261 262 263 264 265
```

输出：

```
3
```

说明：

10 个数据 4 个字节相加后的结果分别为 1 2 3 4 5 6 7 8 9 10，故对 4 取模的结果为 1 2 3 0 1 2 3 0 1 2，c 为 3，所以 0 1 2 都是有效类型，类型为 1 和 2 的有 3 个数据，类型为 0 的只有 2 个数据，故输出 3

示例 2

输入：

```
1 4 256 257 258 259 260 261 262 263 264 265
```

输出：

```
2
```

说明：

10 个数据 4 个字节相加后的结果分别为 1 2 3 4 5 6 7 8 9 10，故对 4 取模的结果为 1 2 3 0 1 2 3 0 1 2，c 为 1，所以只有 0 是有效类型，类型为 0 的有 2 个数据，故输出 2

```
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class Main {
    public static int mod(int number, int b) {
        int sum = 0;
        for (int i = 0; i < 4; ++i) {
            int r1 = number % 16;
            int r2 = (number / 16) % 16;
```

```
            number = number / 16 / 16;
            sum += r1 + r2 * 16;
        }
        return sum % b;
    }

    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        while (in.hasNextInt()) {
            int c = in.nextInt();
            int b = in.nextInt();
            int[] nums = new int[10];
            int maxKey = -1, maxVal = 0;
            Map<Integer, Integer> maps = new HashMap<>();
            for (int i = 0; i < 10; ++i) {
                nums[i] = in.nextInt();
                int val = mod(nums[i], b);
                if (val >= c) {
                    continue;
                }
                if (!maps.containsKey(val)) {
                    maps.put(val, 1);
                    if (maxKey == -1) {
                        maxKey = val;
                        maxVal = 1;
                    }
                } else {
                    maps.put(val, maps.get(val) + 1);
                    if (maps.get(val) > maxVal) {
                        maxVal = maps.get(val);
                        maxKey = val;
                    }
                }
            }
            System.out.println(maxVal);
        }
    }
}
```