import com.google.gson.Gson;

import com.google.gson.reflect.TypeToken;

import java.io.FileReader;

import java.io.IOException;

import java.math.BigInteger;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

public class ShamirSecretSharing {

public static class Root {

public String base;

public String value;

public BigInteger decode() {

return new BigInteger(value, Integer.parseInt(base));

}

}

public static class TestCase {

public int n;

public int k;

public Map<Integer, Root> points;

}

public static BigInteger findSecret(TestCase testCase) {

int k = testCase.k;

List<BigInteger[]> points = new ArrayList<>();

for (Map.Entry<Integer, Root> entry : testCase.points.entrySet()) {

int x = entry.getKey();

BigInteger y = entry.getValue().decode();

points.add(new BigInteger[]{BigInteger.valueOf(x), y});

}

BigInteger secret = BigInteger.ZERO;

for (int i = 0; i < k; i++) {

BigInteger product = BigInteger.ONE;

for (int j = 0; j < k; j++) {

if (i != j) {

BigInteger denominator = points.get(i)[0].subtract(points.get(j)[0]);

product = product.multiply(points.get(j)[0].negate()).multiply(denominator.modInverse(BigInteger.TWO.pow(256)));

}

}

secret = secret.add(points.get(i)[1].multiply(product)).mod(BigInteger.TWO.pow(256));

}

return secret;

}

public static void main(String[] args) {

try (FileReader reader = new FileReader("test\_cases.json")) {

Gson gson = new Gson();

List<TestCase> testCases = gson.fromJson(reader, new TypeToken<List<TestCase>>() {}.getType());

for (TestCase testCase : testCases) {

BigInteger secret = findSecret(testCase);

System.out.println("Secret for Test Case: " + secret);

}

} catch (IOException e) {

e.printStackTrace();

}

}

}