بسمه تعالی

دانشگاه اصفهان

دانشکده مهندسی کامپیوتر

گزارش پروژه نهایی درس رمزنگاری و امنیت شبکه

استاد درس:

دکتر ملا

هانیه ترک ترابی (۴۰۰۳۶۱۳۰۱۲)

```
private static final SecureRandom random = new SecureRandom();

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

    System.out.print("Enter the secret (S): ");
    int secret = input.nextInt();

    System.out.print("Enter the threshold (t): ");
    int t = input.nextInt();

    System.out.print("Enter the total number of shares (n): ");
    int n = input.nextInt();

    System.out.print("Enter the prime number (p): ");
    int prime = input.nextInt();

    List<Share> shares = generateShares(secret, n, t, prime);
    System.out.println("Shares: " + shares);

    List<Share> selectedShares = shares.subList(0, t);
    BigInteger reconstructedSecret = reconstructSecret(selectedShares, prime);
    System.out.println("Reconstructed Secret: " + reconstructedSecret);

    input.close();
}
```

کلاس اصلی Main تعریف شده است. یک نمونه از SecureRandom برای تولید اعداد تصادفی ایجاد می‌شود.

و با تعامل با کاربر مقادیر s و t و n و p را از کاربر دریافت می‌کند.

با انجام عملیات‌هایی بر روی ورودی‌ها سهم‌ها را تولید می‌کند و راز s که تولید کرده است چاپ می‌کند.

```
public static List<Share> generateShares(int secret, int totalShares, int threshold, int prime) {
    BigInteger[] coefficients = new BigInteger[threshold];
    coefficients[0] = BigInteger.valueOf(secret);
    for (int i = 1; i < threshold; i++) {
        coefficients[i] = new BigInteger( numBits: prime - 1, random);
    }
    List<Share> shares = new ArrayList<>();
    for (int i = 1; i <= totalShares; i++) {
        shares.add(new Share(i, evalPolynomial(coefficients, i, prime)));
    }
    return shares;
}
```

تابع generateShares سهم‌ها را از راز secret تولید می‌کند. ضرایب چندجمله‌ای را ایجاد می‌کند. اولین ضریب برابر رازsecret است و سایر ضرایب به صورت تصادفی انتخاب می‌شوند. برای هر سهم، مقدار چندجمله‌ای در نقطه مشخصی ارزیابی می‌شود و به لیست سهم‌ها اضافه می‌گردد.

```java
public static BigInteger evalPolynomial(BigInteger[] coefficients, int x, int prime) {
    BigInteger result = BigInteger.ZERO;
    BigInteger xi = BigInteger.ONE;
    for (BigInteger coefficient : coefficients) {
        result = result.add(coefficient.multiply(xi)).mod(BigInteger.valueOf(prime));
        xi = xi.multiply(BigInteger.valueOf(x));
    }
    return result;
}
```

این متد مقدار چندجمله‌ای را در نقطه x محاسبه می‌کند.

```java
public static BigInteger modularInverse(BigInteger a, int prime) {
    return a.modInverse(BigInteger.valueOf(prime));
}
```

معکوس پیمانه‌ای یک عدد را محاسبه می‌کند.

```java
public static BigInteger reconstructSecret(List<Share> shares, int prime) {
    BigInteger secret = BigInteger.ZERO;
    for (int j = 0; j < shares.size(); j++) {
        BigInteger xj = BigInteger.valueOf(shares.get(j).x);
        BigInteger yj = shares.get(j).y;
        BigInteger numerator = BigInteger.ONE;
        BigInteger denominator = BigInteger.ONE;
        for (int m = 0; m < shares.size(); m++) {
            if (m != j) {
                BigInteger xm = BigInteger.valueOf(shares.get(m).x);
                numerator = numerator.multiply(xm.negate()).mod(BigInteger.valueOf(prime));
                denominator = denominator.multiply(xj.subtract(xm)).mod(BigInteger.valueOf(prime));
            }
        }
        BigInteger lagrangePolynomial = numerator.multiply(modularInverse(denominator, prime)).mod(BigInteger.valueOf(prime));
        secret = secret.add(yj.multiply(lagrangePolynomial)).mod(BigInteger.valueOf(prime));
    }
    return secret;
}
```

این متد از الگوریتم لاگرانژ برای بازسازی راز استفاده می‌کند.

برای هر سهم، چندجمله‌ای لاگرانژ محاسبه می‌شود و سهم‌ها با هم ترکیب می‌شوند تا راز بازسازی شود.

```java
static class Share {
    int x;
    BigInteger y;

    Share(int x, BigInteger y) {
        this.x = x;
        this.y = y;
    }

    @Override
    public String toString() {
        return "(" + x + ", " + y + ")";
    }
}
```

یک کلاس ساده که هر سهم را به عنوان جفتی از x و y نگهداری می‌کند.

متد toString برای نمایش سهم‌ها به صورت (x, y).