КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра програмних систем і технологій

Захист програм та даних

Практична робота № 2.3

Виконали студенти

групи ІПЗ-33

Щука Богдан

Ємець Євгенія

Олійник Анастасія

Перевірила

доцент Супрун О.М.

Київ-2018

**Хід роботи**

**Завдання 1**

У практичній роботі необхідно привести послідовність виконання процедур генерації і перевірки ЕЦП з використанням наступних способів:

- на базі алгоритмів RSA;

- по ГОСТ 34.10-94;

- по ГОСТ 34.10-2001.

Лістинг програми

**import** java.util.Random;  
**import** java.math.BigInteger;  
  
**public class** RSA {  
 **static int**[] *primeNumbers* = {2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47};  
 **static** Random *random*;  
 **static int** *f*;  
 **static int** *p*;  
 **static int** *q*;  
 **static int** *n*;  
 **static int** *e*;  
 **static int** *d*;  
 **static int** *h*;  
 **static int** *s*;  
  
 **static** {  
 *random* = **new** Random(System.*currentTimeMillis*());  
 *p*=*getPrimeByIndex*(*random*.nextInt(3+(*primeNumbers*.**length**-3)));  
 **do** {  
 *q*=*getPrimeByIndex*(3+(*random*.nextInt(*primeNumbers*.**length**-3)));  
 }  
 **while** (*p* == *q*);  
 *n* = *p* \* *q*;  
 *f* = (*p* - 1) \* (*q* - 1);  
 **for** (**int** i = 0; ; i++) {  
 **if** (*f* % *getPrimeByIndex*(i) != 0) {  
 *e* = *getPrimeByIndex*(i);  
 **break**; } }  
  
 **for** (**int** i = 1; ; i++) {  
 **if** ((i \* *e*) % *f* == 1) {  
 *d* = i;  
 **break**; } } }  
 **private static int** getPrimeByIndex(**int** index) {  
 **return** *primeNumbers*[index];  
 }  
 **private static void** setHash() {  
 *h* = 12;  
 }  
 **private static int** getHash(**int** h, **int** d) {  
 **int** degree = h;  
 **for** (**int** i = 0; i < d - 1; i++) {  
 h = (**int**) h \* degree;  
 h = h % *n*;  
 }  
 **return** h;  
 }  
  
 **public static void** checkECP() {  
 *setHash*();  
 *s* = *getHash*(*h*, *d*);  
 System.***out***.println(**"Пользователь Алиса генерирует ключи:"**);  
 System.***out***.println(**"e: "** + *e*);  
 System.***out***.println(**"d: "** + *d*);  
 System.***out***.println(**"n: "** + *n*);  
 System.***out***.println(**"\nВычисление хэш-образа сообщения:"**);  
 System.***out***.println(**"h(T): "** + *h*);  
 System.***out***.println(**"\nВыработка цифровой подписи:"**);  
 System.***out***.println(**"s: "** + *s*);  
 System.***out***.println(**"\nПользователь Алиса отправляет исходное \nсообщение и цифровую подпись пользователю Бобу"**);  
 System.***out***.println(**"\nПользователь Боб вычисляет хэш-образ \nпо полученному сообщению:"**);  
 System.***out***.println(**"h' = h(T') = "** + *h*);  
 System.***out***.println(**"\nВычисление хэш-образа из цифровой подписи h:"**);  
 System.***out***.println(**"h: "** + *getHash*(*s*, *e*));  
 System.***out***.println(**"\nСравнение h' и h: "**);  
 System.***out***.println(**"h' = h ("** + *h* + **" = "** + *getHash*(*s*, *e*) + **") ?\n"**);  
 **if**(*h* == *getHash*(*s*, *e*)) {  
 System.***out***.println(**"Проверка сертификата прошла успешно"**); }  
 **else** {  
 System.***out***.println(**"Проверка сертификата провалилась"**);}}}

**public class** HOST12 {  
 **private static int** *h* = 26;  
 **private static int** *e*;  
 **private static int** *es*;  
 **private static int** *rs*;  
 **private static int** *q* = 47;  
 **private static int** *k*;  
 **private static int** *xp* = 7;  
 **private static int** *yp* = 17;  
 **private static int** *xc* = 16;  
 **private static int** *yc* = 16;  
 **private static int** *r* = 0;  
 **private static int** *s* = 0;  
 **private static int** *d* = 10;  
 **private static int** *v*;  
 **private static int** *z1*;  
 **private static int** *z2*;  
 **private static int** *xcs* = 16;  
 **private static int** *ycs* = 16;  
 **static** Random *random*;  
 **static** {  
 *random* = **new** Random();  
 }  
 **public static void** sendMessage(){  
 *e* = *h* % *q*;  
 **while**(*s*==0){  
 **while**(*r*==0){  
 *k* = *random*.nextInt(*q*-2) + 2;  
 *r* = *xc* % *q*;  
 }  
 *s* = ( *r* \* *d* + *k* \* *e*)%*q*;  
 }  
 }  
 **public static void** getMessage(){  
 *es* = *h* % *q*;  
 *es* = 27;  
 *v* = *es* % *q*;  
 *z1* = ( *s* \* *v*) % *q*;  
 *z2* = ((*q*-*r*)\**v*)%*q*;  
 *rs* = *xcs* % *q*;  
 }  
 **public static void** checkECP() {  
 System.***out***.println(**"Пользователь Алиса генерирует ключи:"**);  
 *sendMessage*();  
 System.***out***.println(**"e: "** + *e*);  
 System.***out***.println(**"d: "** + *d*);  
 System.***out***.println(**"k: "** + *k*);  
 System.***out***.println(**"Точка эллиптической кривой C(xc, yc) = ("** + *xc* + **", "** + *yc* + **")"**);  
 System.***out***.println(**"r: "** + *r*);  
 System.***out***.println(**"\nВычисление хэш-образа сообщения:"**);  
 System.***out***.println(**"h(T): "** + *h*);  
 *sendMessage*();  
 System.***out***.println(**"\nВыработка цифровой подписи:"**);  
 System.***out***.println(**"s: "** + *s*);  
 System.***out***.println(**"\nПользователь Алиса отправляет исходное \nсообщение и цифровую подпись пользователю Бобу"**);  
 *getMessage*();  
 *getMessage*();  
 System.***out***.println(**"\nПользователь Боб вычисляет хэш-образ \nпо полученному сообщению:"**);  
 System.***out***.println(**"h' = h(T') = "** + *h*);  
 System.***out***.println(**"e': "** + *es*);  
 System.***out***.println(**"z1: "** + *z1*);  
 System.***out***.println(**"z2: "** + *z2*);  
 System.***out***.println(**"v: "** + *v*);  
 System.***out***.println(**"Точка эллиптической кривой C'(xc', yc') = ("** + *xcs* + **", "** + *ycs* + **")"**);  
 System.***out***.println(**"r': "** + *rs*);  
 **if**(*r* == *rs*) {  
 System.***out***.println(**"Проверка сертификата прошла успешно"**);  
 }  
 **else** {  
 System.***out***.println(**"Проверка сертификата провалилась"**);  
 }  
 }  
}

**public class** HOST94 {  
 **private static int** *p* = 79;  
 **private static int** *q* = 13;  
 **private static int** *a*;  
 **private static int** *x*;  
 **private static int** *y*;  
 **private static int** *h* = 21;  
 **private static int** *w*;  
 **private static int** *k*;  
 **private static int** *s* = 0;  
 **private static int** *ws* = 0;  
 **private static int** *v*;  
 **private static int** *z1*;  
 **private static int** *z2*;  
 **private static** BigInteger *u*;  
 **static** Random *random*;  
  
 **static**{  
 *random* = **new** Random();  
 *a* = *random*.nextInt(*p* - 3) + 2;  
 **while**(*getHash*(*a*, *q*, *p*)%*p*!=1){  
 *a* = *random*.nextInt(*p* - 3) + 2; }  
 *x* = *random*.nextInt(*q* - 2) + 2;  
 *y* = *getHash*(*a*, *x*, *p*); }  
 **private static int** getHash(**int** h, **int** d, **int** p) {  
 **int** degree = h;  
 **for** (**int** i = 0; i < d - 1; i++) {  
 h = (**int**) h \* degree;  
 h = h % p; }  
  
 **return** h; }  
 **public static void** sendMessage(){  
 **while**(*s* == 0){  
 **while**(*ws* == 0){  
 *k* = *random*.nextInt(*q* - 2) + 2;  
 *w* = *getHash*(*a*, *k*, *p*);  
 *ws* = *w* % *q*; }  
 *s* = (*x*\**ws* + *k*\**h*)%*q*; } }  
 **public static void** getMessage(){  
 *v* = *getHash*(*h*, *q*-2, *q*);  
 *z1* = (*s* \* *v*) % *q*;  
 *z2* = ((*q* - *ws*)\**v*)%*q*;  
 *u* = BigInteger.*valueOf*((**long**)Math.*pow*(*a*, *z1*));  
 *u* = *u*.multiply(BigInteger.*valueOf*((**long**)Math.*pow*(*y*, *z2*)));  
 *u* = *u*.mod(BigInteger.*valueOf*(*p*));  
 *u* = *u*.mod(BigInteger.*valueOf*(*q*));  
 }  
 **public static void** checkECP() {  
 System.***out***.println(**"Пользователь Алиса генерирует ключи:"**);  
 System.***out***.println(**"p: "** + *p*);  
 System.***out***.println(**"q: "** + *q*);  
 System.***out***.println(**"a: "** + *a*);  
 System.***out***.println(**"x: "** + *x*);  
 System.***out***.println(**"y: "** + *y*);  
 System.***out***.println(**"k: "** + *k*);  
 System.***out***.println(**"w: "** + *w*);  
 System.***out***.println(**"w': "** + *ws*);  
 System.***out***.println(**"\nВычисление хэш-образа сообщения:"**);  
 System.***out***.println(**"h(T): "** + *h*);  
 *sendMessage*();  
 System.***out***.println(**"\nВыработка цифровой подписи:"**);  
 System.***out***.println(**"s: "** + *s*);  
 System.***out***.println(**"\nПользователь Алиса отправляет исходное \nсообщение и цифровую подпись пользователю Бобу"**);  
 *getMessage*();  
 System.***out***.println(**"\nПользователь Боб вычисляет хэш-образ \nпо полученному сообщению:"**);  
 System.***out***.println(**"h' = h(T') = "** + *h*);  
 System.***out***.println(**"v: "** + *v*);  
 System.***out***.println(**"u: "** + *u*);  
 **if**(*u*.intValue() == *ws*) {  
 System.***out***.println(**"Проверка сертификата прошла успешно"**); }  
 **else** {  
 System.***out***.println(**"Проверка сертификата провалилась"**);  
 }  
 }  
}

**public class** Main {  
 **public static void** main(String[] args) {  
 RSA.*checkECP*();  
 *HOST94.checkECP();  
 HOST12.checkECP();* }  
}

Приклад роботи програми



