PRACTICAL - 07

Code:

```
// Ashwin Navange A-38 CSE
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
typedef long long int dlong;
typedef struct {
  dlong x, y;
} epnt;
typedef struct {
  long a, b;
  dlong N;
  epnt G;
  dlong r;
} curve;
typedef struct {
  long a, b;
} pair;

const long mxN = 1073741789;
const long mxr = 1073807325;
const long inf = -2147483647;
curve e;
epnt zerO;
int inverr;
long exgcd (long v, long u)
{
register long q, t;
long r = 0, s = 1;
if (v < 0) v += u;

  while (v) {
    q = u / v;
    t = u - q * v;
    u = v; v = t;
    t = r - q * s;
    r = s; s = t;
  }
  if (u != 1) {
    printf (" impossible inverse mod N, gcd = %d\n", u);
    inverr = 1;
  }
return r;
}
static inline dlong modn (dlong a)
{
  a %= e.N;
  if (a < 0) a += e.N;
return a;
}
```

```
dlong modr (dlong a)
{
  a %= e.r;
  if (a < 0) a += e.r;
return a;
}
long disc (void)
{
dlong c, a = e.a, b = e.b;
  c = 4 * modn(a * modn(a * a));
return modn(-16 * (c + 27 * modn(b * b)));
}
int isO (epnt p)
{
return (p.x == inf) && (p.y == 0);
}
int ison (epnt p)
{
long r, s;
if (! isO (p)) {
  r = modn(e.b + p.x * modn(e.a + p.x * p.x));
  s = modn(p.y * p.y);
}
return (r == s);
}
void padd (epnt *r, epnt p, epnt q)
{
dlong la, t;

if (isO(p)) {*r = q; return;}
if (isO(q)) {*r = p; return;}

if (p.x != q.x) {
  t = p.y - q.y;
  la = modn(t * exgcd(p.x - q.x, e.N));
}
else
  if ((p.y == q.y) && (p.y != 0)) {
    t = modn(3 * modn(p.x * p.x) + e.a);
    la = modn(t * exgcd (2 * p.y, e.N));
  }
  else
    {*r = zerO; return;}

t = modn(la * la - p.x - q.x);
r->y = modn(la * (p.x - t) - p.y);
r->x = t; if (inverr) *r = zerO;
}
void pmul (epnt *r, epnt p, long k)
{
epnt s = zerO, q = p;

  for (; k; k >>= 1) {
```

```
    if (k & 1) padd(&s, s, q);
    if (inverr) {s = zerO; break;}
    padd(&q, q, q);
  }
*r = s;
}
void pprint (char *f, epnt p)
{
dlong y = p.y;

  if (isO (p))
    printf ("%s (0)\n", f);

  else {
    if (y > e.N - y) y -= e.N;
    printf ("%s (%lld, %lld)\n", f, p.x, y);
  }
}
int ellinit (long i[])
{
long a = i[0], b = i[1];
  e.N = i[2]; inverr = 0;

if ((e.N < 5) || (e.N > mxN)) return 0;

  e.a = modn(a);
  e.b = modn(b);
  e.G.x = modn(i[3]);
  e.G.y = modn(i[4]);
  e.r = i[5];

if ((e.r < 5) || (e.r > mxr)) return 0;

printf ("\nE: y^2 = x^3 + %dx + %d", a, b);
printf (" (mod %lld)\n", e.N);
pprint ("base point G", e.G);
printf ("order(G, E) = %lld\n", e.r);

return 1;
}
double rnd(void)
{
return rand() / ((double)RAND_MAX + 1);
}
pair signature (dlong s, long f)
{
long c, d, u, u1;
pair sg;
epnt V;

printf ("\nsignature computation\n");
do {
  do {
```

```
    u = 1 + (long)(rnd() * (e.r - 1));
    pmul (&V, e.G, u);
    c = modr(V.x);
  }
  while (c == 0);

  u1 = exgcd (u, e.r);
  d = modr(u1 * (f + modr(s * c)));
}
while (d == 0);
printf ("one-time u = %d\n", u);
pprint ("V = uG", V);

sg.a = c; sg.b = d;
return sg;
}
int verify (epnt W, long f, pair sg)
{
long c = sg.a, d = sg.b;
long t, c1, h1, h2;
dlong h;
epnt V, V2;

  t = (c > 0) && (c < e.r);
  t &= (d > 0) && (d < e.r);
  if (! t) return 0;

printf ("\nsignature verification\n");
  h = exgcd (d, e.r);
  h1 = modr(f * h);
  h2 = modr(c * h);
  printf ("h1,h2 = %d, %d\n", h1,h2);
  pmul (&V, e.G, h1);
  pmul (&V2, W, h2);
  pprint ("h1G", V);
  pprint ("h2W", V2);
  padd (&V, V, V2);
  pprint ("+ =", V);
  if (isO (V)) return 0;
  c1 = modr(V.x);
  printf ("c' = %d\n", c1);

return (c1 == c);
}

void ec_dsa (long f, long d)
{
long i, s, t;
pair sg;
epnt W;
  t = (disc() == 0);
  t |= isO (e.G);
  pmul (&W, e.G, e.r);
```

```
  t |= ! isO (W);
  t |= ! ison (e.G);
  if (t) goto errmsg;

printf ("\nkey generation\n");
  s = 1 + (long)(rnd() * (e.r - 1));
  pmul (&W, e.G, s);
  printf ("private key s = %d\n", s);
  pprint ("public key W = sG", W);
  t = e.r;
  for (i = 1; i < 32; i <<= 1)
    t |= t >> i;
  while (f > t) f >>= 1;
  printf ("\naligned hash %x\n", f);

  sg = signature (s, f);
  if (inverr) goto errmsg;
  printf ("signature c,d = %d, %d\n", sg.a, sg.b);

  if (d > 0) {
    while (d > t) d >>= 1;
    f ^= d;
    printf ("\ncorrupted hash %x\n", f);
  }

  t = verify (W, f, sg);
  if (inverr) goto errmsg;
  if (t)
    printf ("Valid\n_____\n");
  else
    printf ("invalid\n_______\n");

  return;

errmsg:
printf ("invalid parameter set\n");
printf ("_____\n");
}


int main (void)
{
printf("Ashwin Navange A-38 CSE\n");
typedef long eparm[6];
long d, f;
zerO.x = inf; zerO.y = 0;
srand(time(NULL));
eparm *sp, sets[3] = {
  {355, 671, 1073741789, 13693, 10088, 1073807281},
  { 3,   2,         5,     2,     1,         5},
  //{  0,   7,   67096021,  6580,   779,   67079644},
};
  f = 0x789abcde; d = 0;
```

```
  for (sp = sets; ; sp++) {
    if (ellinit (*sp))
      ec_dsa (f, d);

    else
      break;
  }
}
```

Output: