```
In[ ]:= (*Сложение двух точек элептической кривой*)
EllipticAdd[p_, a_, b_, c_, P_List, Q_List] := Module[{lam, x3, y3, P3},
                                                 программный модуль
  Which[
  условный оператор с множественными ветвями
   P == {O}, Q,
        О большое
   Q == {O}, P,
        О большое
   P[[1]] ≠ Q[[1]],
       lam = Mod[(Q[[2]] - P[[2]]) PowerMod[Q[[1]] - P[[1]], p - 2, p], p];
             остаток от деления          степень по модулю
       x3 = Mod[lam^2 - a - P[[1]] - Q[[1]], p];
            остаток от деления
       y3 = Mod[-(lam (x3 - P[[1]]) + P[[2]]), p];
            остаток от деления
       {x3, y3},
   (P == Q) ∧ (P[[2]] == 0), {O},
                             О большое
   (P == Q) ∧ (P ≠ {O}),
                  О большое
       lam = Mod[(3 * P[[1]]^2 + 2 a * P[[1]] + b) PowerMod[2 P[[2]], p - 2, p], p];
             остаток от деления                    степень по модулю
       x3 = Mod[lam^2 - a - P[[1]] - Q[[1]], p];
            остаток от деления
       y3 = Mod[-(lam (x3 - P[[1]]) + P[[2]]), p];
            остаток от деления
       {x3, y3},
   (P[[1]] == Q[[1]]) ∧ (P[[2]] ≠ Q[[2]]), {O}
                                           О большое
  ]
 ]
```

In[ ]:=
```
(*Умножение точки элептической кривой на константу*)
EllipticMultSlow[p0_, a0_, b0_, c0_, pointP0_, n0_] := Module[
                                                       программный модуль
  {p = p0, a = a0, b = b0, c = c0, pointP = pointP0, n = n0, pointQ = pointP0},
  Do[pointQ = EllipticAdd[p, a, b, c, pointP, pointQ], {i, 2, n}];
  оператор цикла
  pointQ
 ]
EllipticMult[p0_, a0_, b0_, c0_, pointP0_, n0_] := Module[
                                                   программный модуль
  {pointP = pointP0, n = n0, p = p0, a = a0, b = b0, c = c0, pointQ = {0, 0}, binN},
  binN = IntegerDigits[n, 2];
         цифры целого числа
  Do[
  оператор цикла
   If[binN[[i]] == 0,
   условный оператор
    pointQ = EllipticAdd[p, a, b, c, pointQ, pointQ]
    , pointQ = EllipticAdd[p, a, b, c, EllipticAdd[
       p, a, b, c, pointQ, pointQ
      ], pointP]
   ]
   , {i, 1, Length[binN]}];
            длина
  pointQ]
```

In[ ]:=
```
(*Нахождение порядка элептической кривой*)
EllipticRank[p0_, a0_, b0_, c0_, pointP0_] := Module[
                                              программный модуль
  {p = p0, a = a0, b = b0, c = c0, pointP = pointP0, pointQ = pointP0, i = 1},
  While[pointQ ≠ {O}, pointQ = EllipticAdd[p, a, b, c, pointP, pointQ];
  цикл-пока          О большое
   i++];
  i
 ]
```