

4-9-2024

# Session 1: Points in an elliptic curve

**Nombre:** Álvarez García Brandon Azarael

**Nombre de la materia:** Selected topics in  
cryptography

**Grupo:** 7CM1

**Nombre del profesor:** Dra. Sandra Díaz Santiago

## Introducción teórica del tema

Criptografía de curva elíptica En 1985, los matemáticos Neal Koblitz y Victor Miller presentaron de manera independiente una propuesta para usar curvas elípticas sobre cuerpos finitos en la elaboración de esquemas de cifrado.

En criptografía, no se usan las curvas elípticas basadas en números reales, ya que esto produce errores de redondeo. Por eso, se utilizan curvas elípticas definidas sobre cuerpos finitos, que se pueden representar por medio de la siguiente ecuación:

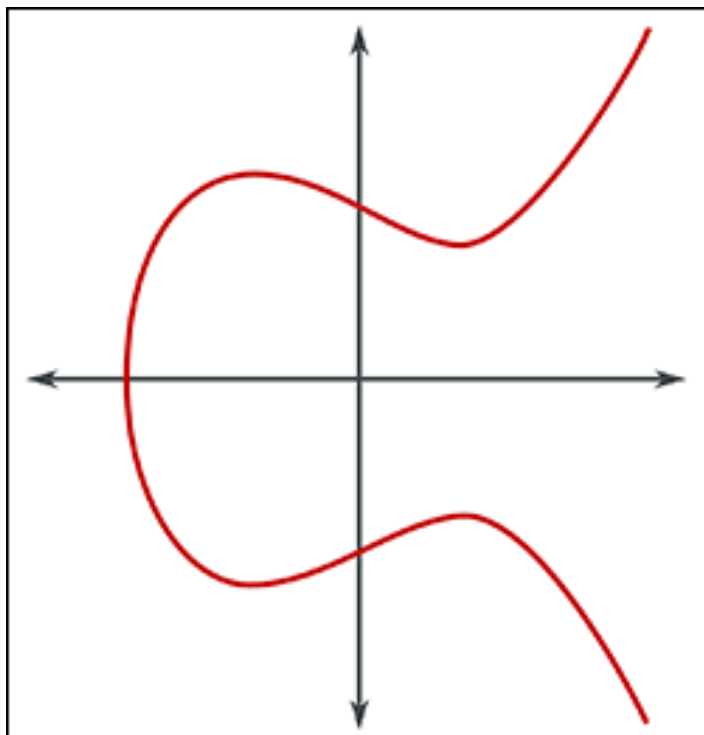
$$y^2 = x^3 + ax + b \mod p$$

donde  $4a^3 + 27b^2 \mod p \neq 0$

Esta criptografía de curva elíptica se caracteriza por tener un número de puntos finito, cuyas coordenadas serán únicamente números enteros. Esa característica es de suma importancia para este algoritmo de cifrado, ya que permite hacer cálculos de forma eficiente y sin errores de redondeo.

Los algoritmos de cifrado normalmente se basan en problemas matemáticos cuya solución aún no ha sido hallada. De esta forma, se garantiza que la función no sea reversible a manos de un tercero malicioso.

Se basa en el problema del logaritmo elíptico, también conocido como problema del logaritmo discreto en curvas elípticas para el cual hoy en día aún no se le ha encontrado una solución, lo cual hace que este algoritmo continúe siendo criptográficamente seguro.



## Ejercicios de programación

1. Design a function that receives as input a prime number  $p > 11$  to find the quadratic residues modulo  $p$  and also the square roots modulo  $p$ . For example if  $p = 11$  and we know that  $2^2 \bmod 11 = 4$  and  $9^2 \bmod 11 = 4$  we know that 4 is a quadratic residue and its square roots are 2 and 9.

```
1 import random
2 from sympy import *
3
4 2 usages
5 class EllipticCurve:
6     # Function to get quadratic residues & SR
7     1 usage
8     def quadraticResidues(self, p):
9         qr = set()
10        sr = {}
11
12        for i in range(1, p):
13            res = pow(base=i, exp=2) % p
14            qr.add(res)
15
16            if res not in sr:
17                sr[res] = [i]
18            else:
19                sr[res].append(i)
20
21        return qr, sr
```

Para todas las funciones cree una clase llamada EllipticCurve en donde pondría toda la lógica de las funciones, para esta función basta con instanciar un conjunto para guardar todos los residuos cuadráticos y se define como conjunto, ya que Python omite valores repetidos y una variable sr, para las raíces cuadradas y se define como un diccionario para relatar la posiciones de dichas raíces, lo que se realiza es recorrer los números desde 1 a p (donde p es un numero primo) e ir guardando esos residuos y comprobar si ese residuo coincide con alguna raíz y se añade, al final retorna el conjunto de residuos cuadráticos y el diccionario de raíces cuadradas.

```

34 # Function to generate the parameters of an elliptic curve
   1 usage
35 def generateParameters(self):
36
37     p = randprime(11, 1000)
38     a = random.randint(1, p - 1)
39     b = random.randint(1, p - 1)
40     function = 0
41
42     # Evaluate discriminant
43     while function != 0:
44         function = (4 * (a ** 3)) + (27 * (b ** 2))
45
46         if function == 0:
47             a = random.randint(1, p - 1)
48             b = random.randint(1, p - 1)
49
50
51     return a, b, p
52

```

Adicionalmente cree esta función que lo único que realiza es generar parámetros  $a, b, p$  para las funciones, en donde genero números aleatorios de 1 a  $p-1$  y evaluó si corresponden a una curva elíptica, si no vuelvo a generar estos valores, en caso de que correspondan, retorno los valores generados.

- Design a function that receives  $a$ ,  $b$  and  $p > 11$ , i.e. the parameters given for an elliptic curve  $y^2 = x^3 + ax + b \pmod p$ , and stores the result of evaluating  $x^3 + ax + b \pmod p$  for every  $0 \leq x \leq p-1$ . For example if  $y^2 = x^3 + x + 6 \pmod{11}$ , and  $x = 4$  then your function must return the result of calculating  $4^3 + 4 + 6 \pmod{11}$ , i.e. 8. You must repeat this, for every member of  $\mathbb{Z}_p$

```

23 # Function to evaluate an Elliptic Curve
   1 usage
24 def ellipticCurve(self, a, b, p):
25
26     result = list()
27
28     for i in range(1, p):
29         y = (pow(i, 3) + (a * i) + b) % p
30         result.append(y)
31
32     return result
33

```

Para esta función es evaluar con todos los elementos del conjunto  $p$ , en los bucles no se coloca  $p-1$ , ya que no realiza este último recorrido por tanto no es necesario y solo se evalúa la función y los resultados se van guardando en una lista que al final se retorna.

- Using the previous functions implement a function that receive the parameters of an elliptic curve, i.e.  $a$ ,  $b$  and  $p > 11$  and as output finds the points in the curve  $y^2 = x^3 + ax + b \pmod{p}$ . Also include the number of points in the curve. Print the parameters of the curve  $a, b$  and  $p$ , together with the list of points and the total number of points in a textfile.

```

1  from EllipticCurve import EllipticCurve
2
3  1 usage
4  def buildCurve(a,b,p):
5      result = curve.ellipticCurve(a,b,p)
6      qr,sr = curve.quadraticResidues(p)
7
8      return result,qr,sr
9
10 #Instance of EllipticCurve class
11 curve = EllipticCurve()
12
13 #Generate a random parameters of an elliptic curve
14 a,b,p = curve.generateParameters()
15
16 #Generate the results of the function
17 result,qr,sr = buildCurve(a,b,p)
18
19
20 print(f"a parameter: {a}, b parameter {b}, p number {p} \n")
21 print(f"Results of Evaluate a function x3 + ax +b mod p \n {result} \n")
22 print(f"Quadratic Residues \n {qr} \n")
23 print(f"Square roots \n {sr} \n")
24 print(f"Total number of points: {len(sr)} ")
25
26
27 file = open("ParametersofCurve.txt","w")
28 file.write(f"a parameter: {a}, b parameter {b}, p number {p} \n\n")
29 file.write('List of points \n'+str(sr))
30 file.write(f"\n\nTotal number of points: {len(sr)}")
31

```

Para esta última función se instancia en el main, ya que instancio un objeto de mi clase `EllipticCurve` e incorporo estas funciones dentro de `buildCurve`, retornando los valores necesarios de la curva o de dichas funciones, genero  $a, b, p$  con mi función y genero los valores de `result` que corresponde a los valores

evaluados de la curva, qr y sr de mi nueva función creada buildCurve, posteriormente imprimo los datos y los escribo en un archivo de texto solicitado.

## Screen shots del programa en funcionamiento.

### Prueba 1:

```
Project ▾
  ▾ Practica_1 D:\ESCOM\10ºSemestre\Cripto 2\Practic
    EllipticCurve.py
    main.py
    ParametersofCurve.txt
    Portada ESCOM.docx
    ~$rtada ESCOM.docx
    ~WRL0005.tmp
  ▸ External Libraries
    Scratches and Consoles

main.py ParametersofCurve.txt EllipticCurve.py
1 a parameter: 7, b parameter 199, p number 521
2
3 List of points
4 {1: [1, 520], 4: [2, 519], 9: [3, 518], 16: [4, 517], 25: [5, 516], 36: [6, 515], 49: [7, 514], 64: [8, 513], 81: [9, 512], 100: [10, 511], 121: [11, 510], 144: [12, 509], 169:
5
6 Total number of points: 260

Run main x
D:\Programs\Anaconda\python.exe "D:\ESCOM\10ºSemestre\Cripto 2\Practic\Practica_1\main.py"
a parameter: 7, b parameter 199, p number 521

Results of Evalte a function x3 + ax +b mod p
[207, 221, 247, 291, 359, 457, 70, 246, 470, 227, 44, 448, 483, 436, 32, 239, 21, 426, 418, 3, 229, 60, 23, 124, 369, 243, 273, 465, 304, 317, 510, 368, 418, 145, 76, 217, 53, 111, 397, 396, 114, 78, 294, 247, 464, 430, 151]

Quadratic Residues
{512, 1, 513, 2, 4, 517, 516, 519, 8, 9, 10, 11, 5, 13, 520, 16, 18, 20, 21, 22, 25, 26, 29, 31, 32, 36, 37, 40, 42, 44, 45, 47, 49, 50, 51, 52, 53, 55, 57, 58, 59, 62, 64, 65, 69, 71, 72, 74, 80, 81, 83, 84, 88, 90, 94, 97}

Square roots
{1: [1, 520], 4: [2, 519], 9: [3, 518], 16: [4, 517], 25: [5, 516], 36: [6, 515], 49: [7, 514], 64: [8, 513], 81: [9, 512], 100: [10, 511], 121: [11, 510], 144: [12, 509], 169: [13, 508], 196: [14, 507], 225: [15, 506], 256: [16, 505], 289: [17, 504], 324: [18, 503], 361: [19, 502], 400: [20, 501], 441: [21, 500], 484: [22, 499], 529: [23, 498], 576: [24, 497], 625: [25, 496], 676: [26, 495], 729: [27, 494], 784: [28, 493], 841: [29, 492], 900: [30, 491], 961: [31, 490], 1024: [32, 489], 1089: [33, 488], 1156: [34, 487], 1225: [35, 486], 1296: [36, 485], 1369: [37, 484], 1444: [38, 483], 1521: [39, 482], 1600: [40, 481], 1681: [41, 480], 1764: [42, 479], 1849: [43, 478], 1936: [44, 477], 2025: [45, 476], 2116: [46, 475], 2209: [47, 474], 2304: [48, 473], 2401: [49, 472], 2500: [50, 471], 2601: [51, 470], 2704: [52, 469], 2809: [53, 468], 2916: [54, 467], 3025: [55, 466], 3136: [56, 465], 3249: [57, 464], 3364: [58, 463], 3481: [59, 462], 3600: [60, 461], 3721: [61, 460], 3844: [62, 459], 3969: [63, 458], 4096: [64, 457], 4225: [65, 456], 4356: [66, 455], 4489: [67, 454], 4624: [68, 453], 4761: [69, 452], 4900: [70, 451], 5041: [71, 450], 5184: [72, 449], 5329: [73, 448], 5476: [74, 447], 5625: [75, 446], 5776: [76, 445], 5929: [77, 444], 6084: [78, 443], 6241: [79, 442], 6400: [80, 441], 6561: [81, 440], 6724: [82, 439], 6891: [83, 438], 7060: [84, 437], 7231: [85, 436], 7404: [86, 435], 7589: [87, 434], 7776: [88, 433], 7965: [89, 432], 8164: [90, 431], 8365: [91, 430], 8568: [92, 429], 8773: [93, 428], 8980: [94, 427], 9189: [95, 426], 9400: [96, 425], 9613: [97, 424], 9828: [98, 423], 10045: [99, 422], 10264: [100, 421], 10485: [101, 420], 10708: [102, 419], 10933: [103, 418], 11160: [104, 417], 11389: [105, 416], 11620: [106, 415], 11853: [107, 414], 12088: [108, 413], 12335: [109, 412], 12584: [110, 411], 12835: [111, 410], 13088: [112, 409], 13343: [113, 408], 13600: [114, 407], 13859: [115, 406], 14120: [116, 405], 14383: [117, 404], 14648: [118, 403], 14915: [119, 402], 15184: [120, 401], 15455: [121, 400], 15728: [122, 399], 16003: [123, 398], 16280: [124, 397], 16559: [125, 396], 16840: [126, 395], 17123: [127, 394], 17408: [128, 393], 17695: [129, 392], 17984: [130, 391], 18275: [131, 390], 18568: [132, 389], 18863: [133, 388], 19160: [134, 387], 19459: [135, 386], 19760: [136, 385], 20063: [137, 384], 20368: [138, 383], 20675: [139, 382], 20984: [140, 381], 21295: [141, 380], 21608: [142, 379], 21923: [143, 378], 22240: [144, 377], 22559: [145, 376], 22880: [146, 375], 23203: [147, 374], 23528: [148, 373], 23855: [149, 372], 24184: [150, 371], 24515: [151, 370], 24848: [152, 369], 25183: [153, 368], 25520: [154, 367], 25859: [155, 366], 26200: [156, 365], 26543: [157, 364], 26888: [158, 363], 27235: [159, 362], 27584: [160, 361], 27935: [161, 360], 28288: [162, 359], 28643: [163, 358], 29000: [164, 357], 29359: [165, 356], 29720: [166, 355], 30083: [167, 354], 30448: [168, 353], 30815: [169, 352], 31184: [170, 351], 31555: [171, 350], 31928: [172, 349], 32303: [173, 348], 32680: [174, 347], 33059: [175, 346], 33440: [176, 345], 33823: [177, 344], 34208: [178, 343], 34595: [179, 342], 34984: [180, 341], 35375: [181, 340], 35768: [182, 339], 36163: [183, 338], 36560: [184, 337], 36959: [185, 336], 37360: [186, 335], 37763: [187, 334], 38168: [188, 333], 38575: [189, 332], 38984: [190, 331], 39395: [191, 330], 39808: [192, 329], 40223: [193, 328], 40640: [194, 327], 41059: [195, 326], 41480: [196, 325], 41903: [197, 324], 42328: [198, 323], 42755: [199, 322], 43184: [200, 321], 43615: [201, 320], 44048: [202, 319], 44483: [203, 318], 44920: [204, 317], 45359: [205, 316], 45800: [206, 315], 46243: [207, 314], 46688: [208, 313], 47135: [209, 312], 47584: [210, 311], 48035: [211, 310], 48488: [212, 309], 48943: [213, 308], 49400: [214, 307], 49859: [215, 306], 50320: [216, 305], 50783: [217, 304], 51248: [218, 303], 51715: [219, 302], 52184: [220, 301], 52655: [221, 300], 53128: [222, 299], 53603: [223, 298], 54080: [224, 297], 54559: [225, 296], 55040: [226, 295], 55523: [227, 294], 56008: [228, 293], 56495: [229, 292], 56984: [230, 291], 57475: [231, 290], 57968: [232, 289], 58463: [233, 288], 58960: [234, 287], 59459: [235, 286], 59960: [236, 285], 60463: [237, 284], 60968: [238, 283], 61475: [239, 282], 61984: [240, 281], 62495: [241, 280], 63008: [242, 279], 63523: [243, 278], 64040: [244, 277], 64559: [245, 276], 65080: [246, 275], 65603: [247, 274], 66128: [248, 273], 66655: [249, 272], 67184: [250, 271], 67715: [251, 270], 68248: [252, 269], 68783: [253, 268], 69320: [254, 267], 69859: [255, 266], 70400: [256, 265], 70943: [257, 264], 71488: [258, 263], 72035: [259, 262], 72584: [260, 261], 73135: [261, 260], 73688: [262, 259], 74243: [263, 258], 74800: [264, 257], 75359: [265, 256], 75920: [266, 255], 76483: [267, 254], 77048: [268, 253], 77615: [269, 252], 78184: [270, 251], 78755: [271, 250], 79328: [272, 249], 79903: [273, 248], 80480: [274, 247], 81059: [275, 246], 81640: [276, 245], 82223: [277, 244], 82808: [278, 243], 83395: [279, 242], 83984: [280, 241], 84575: [281, 240], 85168: [282, 239], 85763: [283, 238], 86360: [284, 237], 86959: [285, 236], 87560: [286, 235], 88163: [287, 234], 88768: [288, 233], 89375: [289, 232], 89984: [290, 231], 90595: [291, 230], 91208: [292, 229], 91823: [293, 228], 92440: [294, 227], 93059: [295, 226], 93680: [296, 225], 94303: [297, 224], 94928: [298, 223], 95555: [299, 222], 96184: [300, 221], 96815: [301, 220], 97448: [302, 219], 98083: [303, 218], 98720: [304, 217], 99359: [305, 216], 100000: [306, 215], 100643: [307, 214], 101288: [308, 213], 101935: [309, 212], 102584: [310, 211], 103235: [311, 210], 103888: [312, 209], 104543: [313, 208], 105200: [314, 207], 105859: [315, 206], 106520: [316, 205], 107183: [317, 204], 107848: [318, 203], 108515: [319, 202], 109184: [320, 201], 109855: [321, 200], 110528: [322, 199], 111203: [323, 198], 111880: [324, 197], 112559: [325, 196], 113240: [326, 195], 113923: [327, 194], 114608: [328, 193], 115295: [329, 192], 115984: [330, 191], 116675: [331, 190], 117368: [332, 189], 118063: [333, 188], 118760: [334, 187], 119459: [335, 186], 120160: [336, 185], 120863: [337, 184], 121568: [338, 183], 122275: [339, 182], 122984: [340, 181], 123695: [341, 180], 124408: [342, 179], 125123: [343, 178], 125840: [344, 177], 126559: [345, 176], 127280: [346, 175], 128003: [347, 174], 128728: [348, 173], 129455: [349, 172], 130184: [350, 171], 130915: [351, 170], 131648: [352, 169], 132383: [353, 168], 133120: [354, 167], 133859: [355, 166], 134600: [356, 165], 135343: [357, 164], 136088: [358, 163], 136835: [359, 162], 137584: [360, 161], 138335: [361, 160], 139088: [362, 159], 139843: [363, 158], 140600: [364, 157], 141359: [365, 156], 142120: [366, 155], 142883: [367, 154], 143648: [368, 153], 144415: [369, 152], 145184: [370, 151], 145955: [371, 150], 146728: [372, 149], 147503: [373, 148], 148280: [374, 147], 149059: [375, 146], 149840: [376, 145], 150623: [377, 144], 151408: [378, 143], 152195: [379, 142], 152984: [380, 141], 153775: [381, 140], 154568: [382, 139], 155363: [383, 138], 156160: [384, 137], 156959: [385, 136], 157760: [386, 135], 158563: [387, 134], 159368: [388, 133], 160175: [389, 132], 160984: [390, 131], 161795: [391, 130], 162608: [392, 129], 163423: [393, 128], 164240: [394, 127], 165059: [395, 126], 165880: [396, 125], 166703: [397, 124], 167528: [398, 123], 168355: [399, 122], 169184: [400, 121], 169995: [401, 120], 170808: [402, 119], 171623: [403, 118], 172440: [404, 117], 173259: [405, 116], 174080: [406, 115], 174903: [407, 114], 175728: [408, 113], 176555: [409, 112], 177384: [410, 111], 178215: [411, 110], 179048: [412, 109], 179883: [413, 108], 180720: [414, 107], 181559: [415, 106], 182400: [416, 105], 183243: [417, 104], 184088: [418, 103], 184935: [419, 102], 185784: [420, 101], 186635: [421, 100], 187488: [422, 99], 188343: [423, 98], 189200: [424, 97], 190059: [425, 96], 190920: [426, 95], 191783: [427, 94], 192648: [428, 93], 193515: [429, 92], 194384: [430, 91], 195255: [431, 90], 196128: [432, 89], 197003: [433, 88], 197880: [434, 87], 198759: [435, 86], 199640: [436, 85], 200523: [437, 84], 201408: [438, 83], 202295: [439, 82], 203184: [440, 81], 204075: [441, 80], 204968: [442, 79], 205863: [443, 78], 206760: [444, 77], 207659: [445, 76], 208560: [446, 75], 209463: [447, 74], 210368: [448, 73], 211275: [449, 72], 212184: [450, 71], 213095: [451, 70], 214008: [452, 69], 214923: [453, 68], 215840: [454, 67], 216759: [455, 66], 217680: [456, 65], 218603: [457, 64], 219528: [458, 63], 220455: [459, 62], 221384: [460, 61], 222315: [461, 60], 223248: [462, 59], 224183: [463, 58], 225120: [464, 57], 226059: [465, 56], 226995: [466, 55], 227932: [467, 54], 228871: [468, 53], 229812: [469, 52], 230755: [470, 51], 231700: [471, 50], 232647: [472, 49], 233596: [473, 48], 234547: [474, 47], 235500: [475, 46], 236455: [476, 45], 237412: [477, 44], 238371: [478, 43], 239332: [479, 42], 240295: [480, 41], 241260: [481, 40], 242227: [482, 39], 243196: [483, 38], 244167: [484, 37], 245140: [485, 36], 246115: [486, 35], 247092: [487, 34], 248071: [488, 33], 249052: [489, 32], 250035: [490, 31], 251020: [491, 30], 252007: [492, 29], 252996: [493, 28], 253987: [494, 27], 254980: [495, 26], 255975: [496, 25], 256972: [497, 24], 257971: [498, 23], 258972: [499, 22], 259975: [500, 21], 260980: [501, 20], 261987: [502, 19], 262996: [503, 18], 264007: [504, 17], 265020: [505, 16], 266035: [506, 15], 267052: [507, 14], 268071: [508, 13], 269092: [509, 12], 270115: [510, 11], 271140: [511, 10], 272167: [512, 9], 273196: [513, 8], 274227: [514, 7], 275260: [515, 6], 276295: [516, 5], 277332: [517, 4], 278371: [518, 3], 279412: [519, 2], 280455: [520, 1]}

Total number of points: 260

Process finished with exit code 0
```

### Prueba 2:

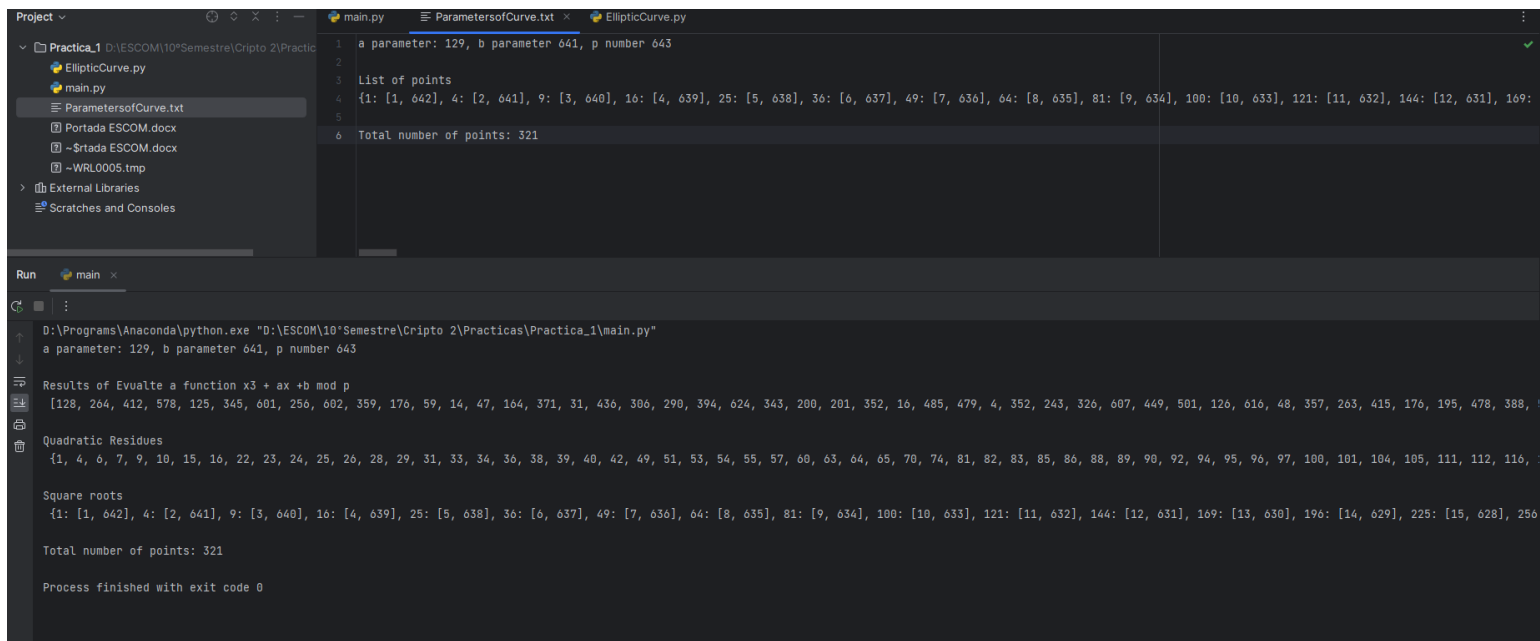
```
Project ▾
  ▾ Practica_1 D:\ESCOM\10ºSemestre\Cripto 2\Practic
    EllipticCurve.py
    main.py
    ParametersofCurve.txt
    Portada ESCOM.docx
    ~$rtada ESCOM.docx
    ~WRL0005.tmp
  ▸ External Libraries
    Scratches and Consoles

main.py ParametersofCurve.txt EllipticCurve.py
1 a parameter: 626, b parameter 280, p number 971
2
3 List of points
4 {1: [1, 970], 4: [2, 969], 9: [3, 968], 16: [4, 967], 25: [5, 966], 36: [6, 965], 49: [7, 964], 64: [8, 963], 81: [9, 962], 100: [10, 961], 121: [11, 960], 144: [12, 959], 169:
5
6 Total number of points: 485

Run main x
D:\Programs\Anaconda\python.exe "D:\ESCOM\10ºSemestre\Cripto 2\Practic\Practica_1\main.py"
a parameter: 626, b parameter 280, p number 971

Results of Evalte a function x3 + ax +b mod p
[907, 569, 243, 906, 622, 368, 150, 945, 817, 743, 729, 781, 905, 136, 422, 798, 299, 873, 584, 409, 354, 425, 628, 969, 483, 147, 938, 920, 99, 423, 927, 646, 557, 666, 8, 531, 299, 289, 507, 959, 680, 647, 866, 372, 142, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260, 261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299, 300, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310, 311, 312, 313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324, 325, 326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336, 337, 338, 339, 340, 341, 342, 343, 344, 345, 346, 347, 348, 349, 350, 351, 352, 353, 354, 355, 356, 357, 358, 359, 360, 361, 362, 363, 364, 365, 366, 367, 368, 369, 370, 371, 372, 373, 374, 375, 376, 377, 378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388, 389, 390, 391, 392, 393, 394, 395, 396, 397, 398, 399, 400, 401, 402, 403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416, 417, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428, 429, 430, 431, 432, 433, 434, 435, 436, 437, 438, 439, 440, 441, 442, 443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454, 455, 456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468, 469, 47
```

## Prueba 3:



```
1 a parameter: 129, b parameter 641, p number 643
2
3 List of points
4 {1: [1, 642], 4: [2, 641], 9: [3, 640], 16: [4, 639], 25: [5, 638], 36: [6, 637], 49: [7, 636], 64: [8, 635], 81: [9, 634], 100: [10, 633], 121: [11, 632], 144: [12, 631], 169:
5
6 Total number of points: 321
```

Run main

```
D:\Programs\Anaconda\python.exe "D:\ESCOM\10ºSemestre\Cripto 2\Practicas\Practica_1\main.py"
a parameter: 129, b parameter 641, p number 643

Results of Evalute a function x3 + ax +b mod p
[128, 264, 412, 578, 125, 345, 601, 256, 602, 359, 176, 59, 14, 47, 164, 371, 31, 436, 306, 290, 394, 624, 343, 200, 201, 352, 16, 485, 479, 4, 352, 243, 326, 607, 449, 501, 126, 616, 48, 357, 263, 415, 176, 195, 478, 388,

Quadratic Residues
{1, 4, 6, 7, 9, 10, 15, 16, 22, 23, 24, 25, 26, 28, 29, 31, 33, 34, 36, 38, 39, 40, 42, 49, 51, 53, 54, 55, 57, 60, 63, 64, 65, 70, 74, 81, 82, 83, 85, 86, 88, 89, 90, 92, 94, 95, 96, 97, 100, 101, 104, 105, 111, 112, 116,

Square roots
{1: [1, 642], 4: [2, 641], 9: [3, 640], 16: [4, 639], 25: [5, 638], 36: [6, 637], 49: [7, 636], 64: [8, 635], 81: [9, 634], 100: [10, 633], 121: [11, 632], 144: [12, 631], 169: [13, 630], 196: [14, 629], 225: [15, 628], 256

Total number of points: 321

Process finished with exit code 0
```