In [1]:
```
p = 13
```

In [2]:
```
a = 5
```

In [3]:
```
legendre_symbol(a, p)
```

Out[3]:
```
-1
```

In [4]:
```
[power_mod(k, 2, p) for k in range(0, p+1)]
```

Out[4]:
```
[0, 1, 4, 9, 3, 12, 10, 10, 12, 3, 9, 4, 1, 0]
```

In [5]:
```
legendre_symbol(10, p)
```

Out[5]:
```
1
```

In [122…
```
b = 20
p = 29
```

In [123…
```
Zp = IntegerModRing(29)
Zp
```

Out[123]:
```
Ring of integers modulo 29
```

In [124…
```
legendre_symbol(b, 29)
```

Out[124]:
```
1
```

In [125…
```
Pol = PolynomialRing(Zp, 'x')
Pol
```

Out[125]:
```
Univariate Polynomial Ring in x over Ring of integers modulo 29
```

In [126…
```
f = Pol(x^2-b)
f
```

Out[126]:
```
x^2 + 9
```

In [127…
```
R = PolynomialQuotientRing(Pol, f, 'a')
R
```

Out[127]:
```
Univariate Quotient Polynomial Ring in a over Ring of integers modulo 29 with modu
lus x^2 + 9
```

In [128…
```
a = R(x)
```

In [129…
```
f1 = R(1+a)
f2 = R(2+3*a)
```

In [130…
```
f1, f2
```

Out[130]:
```
(a + 1, 3*a + 2)
```

In [131…
```
f1*f2
```

Out[131]: 5*a + 4

In [132…
```
z = Zp.random_element()
z
```

Out[132]: 7

In [133…
```
elemento = R(1+z*a)^((p-1)//2)
elemento
```

Out[133]: 28

In [134…
```
elemento[1]
```

Out[134]: 0

In [135…
```
while elemento[1] == 0:
    z = Zp.random_element()
    elemento = R(1+z*a)^((p-1)//2)
```

In [136…
```
u, v = elemento
elemento
```

Out[136]: 4*a

In [137…
```
u, v
```

Out[137]: (0, 4)

In [138…
```
type(v)
```

Out[138]: <class 'sage.rings.finite_rings.integer_mod.IntegerMod_int'>

In [139…
```
sol1, sol2, sol3 = -u/v, (1-u)/v, (-1-u)/v
```

In [140…
```
sol1^2 == b, sol2^2 == b, sol3^2 == b
```

Out[140]: (False, True, True)

In [141…
```
sol1^2, sol2^2 , sol3^2
```

Out[141]: (0, 20, 20)

In [142…
```
sol2
```

Out[142]: 22

In [67]:
```
sol3
```

Out[67]: 6

In [68]:
```
b
```

Out[68]: 10