# CS152-Homework10

Hongchen Cao 2019533114

2020.12.02

## 1

22392

232836

```python
from math import sqrt, ceil, gcd
from sage.all import inverse_mod, factor, crt
```

```python
def shanks(n: int, alpha: int, beta: int) -> int:
    m = ceil(sqrt(n))
    L1 = list()
    for j in range(0, m):
        L1.append((j, pow(alpha, m * j, n + 1)))
    L1.sort(key=lambda x: x[1])

    L2 = list()
    for i in range(0, m):
        L2.append((i, pow(pow(inverse_mod(alpha, n + 1), i) * beta, 1, n + 1)))
    L2.sort(key=lambda x: x[1])

    for pair1 in L1:
        for pair2 in L2:
            if pair1[1] == pair2[1]:
                return pow(m * pair1[0] + pair2[0], 1, n)


print(shanks(24691 - 1, 106, 12375))
print(shanks(458009 - 1, 6, 248388))
```

# 2

smallest i = 444

res = 40007

```python
from math import sqrt, ceil, gcd
from sage.all import inverse_mod, factor, crt
```

```python
def pollardRhoSubFunction(x: int, a: int, b: int, alpha: int, beta: int, n: int, p: int) -> tuple:
    if x % 3 == 1:
        return pow(beta * x, 1, p), pow(a, 1, p), pow(b + 1, 1, n)
    elif x % 3 == 0:
        return pow(x, 2, p), pow(2 * a, 1, n), pow(2 * b, 1, n)
    else:
        return pow(alpha * x, 1, p), pow(a + 1, 1, n), pow(b, 1, p)


def pollardRhoMainFunction(n: int, alpha: int, beta: int, p: int) -> tuple:
    x, a, b = pollardRhoSubFunction(1, 0, 0, alpha, beta, n, p)
    x_, a_, b_ = pollardRhoSubFunction(x, a, b, alpha, beta, n, p)
    ctr = 1
    while x != x_:
        x, a, b = pollardRhoSubFunction(x, a, b, alpha, beta, n, p)
        x_, a_, b_ = pollardRhoSubFunction(x_, a_, b_, alpha, beta, n, p)
        x_, a_, b_ = pollardRhoSubFunction(x_, a_, b_, alpha, beta, n, p)
        ctr += 1

    if gcd(b_ - b, n) != 1:
        return None
    else:
        return pow((a - a_) * inverse_mod((b_ - b), n), 1, n), ctr


print(pollardRhoMainFunction(57251, 2, 56851, 458009))
```

# 3

3909

17102

```
from math import sqrt, ceil, gcd
from sage.all import inverse_mod, factor, crt
```

```
def pohligHellmanSub(p: int, n: int, alpha: int, beta: int, q: int, c: int):
    j = 0
    betaList = list()
    betaList.append(beta)
    a = list()

    while j <= c - 1:
        delta = pow(betaList[j], int(n / pow(q, j + 1)), p)
        i = 0
        while True:
            if delta == pow(alpha, int(i * n / q), p):
                break
            i += 1
        a.append(i)
        betaList.append(pow(betaList[j] * pow(inverse_mod(alpha, p), a[j] * pow(q, j, p))
                                                                , 1, p))
        j += 1

    res = 0
    for i in range(len(a)):
        res += pow(a[i] * pow(2, i), 1, p)
    return res


def pohligHellmanMain(p: int, alpha: int, beta: int):
    n = p - 1
    equations1 = list()
    equations2 = list()
    for pair in factor(n):
        temp = pohligHellmanSub(p, n, alpha, beta, pair[0], pair[1])
        equations1.append(temp)
        equations2.append(pow(pair[0], pair[1]))
    return crt(equations1, equations2)


print(pohligHellmanMain(28703, 5, 8563))
print(pohligHellmanMain(31153, 10, 12611))
```

# 4

## 4.1

$$2^{32} \equiv 176 = 2^4 \times 11$$
$$2^{40} \equiv 110 = 2 \times 5 \times 11$$
$$2^{59} \equiv 60 = 2^2 \times 3 \times 5$$
$$2^{156} \equiv 28 = 2^2 \times 7$$

## 4.2

let $log3 = a$, $log5 = b$, $log7 = c$ and $log11 = d$

$$\begin{cases} 4 \times 1 + d & = 32 \\ 1 + b + d & = 40 \\ 2 \times 1 + a + b & = 59 \\ 2 \times 2 + c & = 156 \end{cases} \tag{1}$$

So, $log3 = 46$, $log5 = 11$, $log7 = 154$, $log11 = 28$

## 4.3

$$173 \times 2^{177} \equiv 168 = 2^3 3^1 7^1$$
$$log173 = 3log2 + log3 + log7 - 177 = 26$$