Cryptography Problems

Jayadev Ghanta

August 2024

1 Elliptic Curves

Problem 1

Suppose that P is a 2-torsion point on an elliptic curve E, and that P is not the point at infinity. Show that the y-coordinate of P is 0.

Proof. The equation for an elliptic curve is given by $y^2 = x^3 + ax + b$. A point P on an elliptic curve is a 2-torsion point if $2P = \mathcal{O}$. For clarification, an elliptic curve forms an abelian group where the operation of addition for two distinct points P and Q are done as the following:

- Find the line that intersects both P and Q.
- Find the third point where this line intersects the elliptic curve, R.
- Calculate the reflection of this point R, which we will call R'.
- Now we have $P + Q = R'$ as our operation definition, with the identity element being the point at infinity.

Since we are only dealing with one point P, we must take the tangent line of the elliptic curve. Doing some basic implicit differentiation, we find that the slope of the line λ must be $\frac{3x_1^2+a}{2y_1}$, where the coordinates of P are (x_1, y_1). Since we are doing the operation $P + P$, we have that the line is $y = \lambda x + b$. We can find b by plugging in x_1 and y_1. Now the updated line is $y = \lambda(x - x_1) + y_1$. Since the line intersects the elliptic curve at a third point to complete the operation, we can plug in the line equation into the elliptic curve equation: $(\lambda(x - x_1) + y_1)^2 = x^3 + ax + b$. The following is the rest of my algebra for the problem.

$$(\lambda(x - x_1) + y_1)^2 = x^3 + ax + b$$

Moving everything to one side, we can get the following:

$$x^3 - \lambda^2(x - x_1)^2 + \square x + \square$$

We can set $\lambda^2 = x_3 + 2x_1$, which makes $x_3 = \lambda^2 - 2x_1$. We can stop here before we solve for y_3. Since we know that x_3 must equal to infinity in order to form the group, we must find a way to make x_3 to be infinite. Since the denominator of λ is $2y_1$, y_1 must equal to 0. □

Problem 2

The right triangle with sides 3, 4, and 5 has area 6. Using elliptic curves, find three more right triangles with rational sides and area 6.

Proof. We start by introducing the concept of a congruent number. An integer N is defined as a *Congruent Number* if there exist rational numbers α, β, and γ such that $\gamma^2 = \alpha^2 + \beta^2$ and $N = \frac{1}{2}\alpha\beta$. Given that N is our target area, multiplying N by 4 yields the relations $(\alpha + \beta)^2 = \gamma^2 - 4N$ and $(\alpha - \beta)^2 = \gamma^2 + 4N$. Dividing these equations by 4, we derive the identities $\left(\frac{\alpha+\beta}{2}\right)^2 = \left(\frac{\gamma^2}{2}\right) - N$ and $\left(\frac{\alpha-\beta}{2}\right)^2 = \left(\frac{\gamma^2}{2}\right) + N$.

By multiplying these equations, we obtain $\left(\frac{\alpha^2-\beta^2}{4}\right)^2 = \left(\frac{\gamma^2}{4}\right)^2 + N^2$. Introducing the substitutions $v = \frac{\alpha^2-\beta^2}{4}$ and $u = \frac{\gamma^2}{2}$, we arrive at the equation $v^2 = u^4 - N^2$. Further manipulation by multiplying by u^2 leads to $(uv)^2 = u^6 - N^2u^2$. Setting $x = u^2 = \left(\frac{\gamma^2}{2}\right)^2$ and $y = uv = \frac{\gamma(\alpha^2-\beta^2)}{8}$, the resulting equation is $y^2 = x^3 - N^2x$, which describes an elliptic curve.

For $N > 0$, there is a one-to-one correspondence between the set of right triangles with rational sides (α, β, γ) and area N, and the set of rational points (x, y) on the elliptic curve $y^2 = x^3 - N^2x$ with $y \neq 0$. The mutually inverse correspondences between these sets are given by $(\alpha, \beta, \gamma) \mapsto \left(\frac{N\beta}{\gamma-\alpha}, \frac{2N^2}{\gamma-\alpha}\right)$ and $(x, y) \mapsto \left(\frac{x^2-N^2}{y}, \frac{2Nx}{y}, \frac{x^2+N^2}{y}\right)$.

This elliptic curve formulation connects to the classic Congruent Number Problem (CNP) through the question: For a whole number N, does there exist a rational point (x, y) with $y \neq 0$ on the elliptic curve $E_N : y^2 = x^3 - N^2x$? Notice that given a right triangle with rational sides and area N, a corresponding rational point (x, y) can be found on the curve E_N.

Applying the Nagell–Lutz Theorem, consider an elliptic curve E in short Weierstrass normal form $E : y^2 = x^3 + Ax + B$, with integral coefficients $A, B \in \mathbb{Z}$. Let $O \neq P = (x, y) \in E(\mathbb{Q}_{\text{tors}})$. Then $x, y \in \mathbb{Z}$, and either $2P = O$ or y^2 divides $\Delta_0 = -\frac{\Delta}{16} = 4A^3 + 27B^2$. Specifically, for the family of elliptic curves of the form $E_N : y^2 = x^3 - N^2x$, where $\Delta_0 = 4N^6$, the torsion points of E_N are either $y = 0$ or y^2 divides $4N^6$.

Finally, using computational tools such as Sage, these theoretical insights can be applied to find explicit examples of right triangles with rational sides and area 6.

```
(18 : 72 : 1)]
[sage: P = E.gens()[0]; P
 (-3 : 9 : 1)
[sage: P*2
 (25/4 : -35/8 : 1)
[sage: P*3
 (-1587/1369 : -321057/50653 : 1)
[sage: P*4
 (1442401/19600 : 1726556399/2744000 : 1)
 sage:
```

The three right triangles with rational sides and area 6, found using the elliptic curve method, have the following sets of sides: $\frac{120}{7}, \frac{7}{10}, \frac{1201}{70}$; $\frac{4653}{851}, \frac{3404}{1551}, \frac{7776485}{1319901}$; and $\frac{1437599}{168140}, \frac{2017680}{1437599}, \frac{2094350404801}{241717895860}$. I found these side lengths by plugging in the rational points found in the elliptic curve into the mutually inverse correspondence that we have calculated before.

□

Problem 3

Let E be the elliptic curve $y^2 = x^3 + 2x + 4$. Write down all the points of $E(\mathbb{F}_5)$. Do the same for $E(\mathbb{F}_7)$ and $E(\mathbb{F}_{11})$.

Proof. In order to find the points, I plugged in numbers from 1 through k for any $\mathbb{F}_k$ for x in the elliptic curve equation. Then I solved the quadratic residue for $y^2 \equiv x^3 + 2x + 4 \pmod{k}$. Now I will showcase my results. For $E(\mathbb{F}_5)$, I get the points: $(0, 2), (0, 3), (2, 2), (2, 3), (3, 0), (4, 2), (4, 3)$. For $E(\mathbb{F}_7)$, I get the points: $(0, 2), (0, 5), (1, 0), (2, 0), (3, 2), (3, 5)$. For $E(\mathbb{F}_{11})$, I get the points: $(0, 2), (0, 9), (3, 3), (3, 8), (7, 2), (7, 9)$. □

2 Zero-Knowledge Proofs

Problem 1

Victor is colorblind. Peggy has two billiard balls, which are identical except for color: one is green and one is red. Peggy would like to convince Victor that they are different, without telling him which one is which. Design a protocol for them to use.

Proof. In this proof we just need to be able to prove distinguishability between the balls. Peggy has two billiard balls, one green and one red, while Victor, who is colorblind, cannot distinguish between them. To convince Victor that the balls are different colors without revealing which is which, they use a clever protocol. Victor secretly flips a coin to decide whether to shuffle the balls behind his back or leave them as they are, then shows them to Peggy. Because Peggy can see the colors, she can easily tell if the balls have been shuffled or not. She informs Victor of her observation, and they repeat this process multiple times. If Peggy consistently identifies whether the balls have been shuffled, Victor becomes increasingly convinced that the balls are indeed different colors. Since Peggy's correct identification could only be random if the balls were the same color, and the odds of her guessing correctly every time by chance are extremely low, Victor realizes that the balls must be of different colors, even though he still doesn't know which is which. This protocol effectively demonstrates a zero-knowledge proof, where Peggy proves the difference without revealing any additional information. □

Problem 2

Find a commitment scheme for a coin flip based on the discrete logarithm problem.

Proof. To design a commitment scheme for a fair coin flip based on the discrete logarithm problem, Alice and Bob first agree on a large prime p and a generator g of the multiplicative group $\mathbb{Z}_p^*$. Alice chooses a random bit b (0 or 1) to represent her coin flip and a random secret r. She then computes her commitment $C = g^b \cdot h^r \mod p$, where h is a value derived from g but unknown to Bob. Alice sends this commitment C to Bob. Similarly, Bob selects his own bit b', a secret r', and computes his commitment $C' = g^{b'} \cdot h^{r'} \mod p$, which he sends to Alice. After exchanging commitments, Alice and Bob reveal their respective bits and secrets, allowing each to verify the other's commitment. Then they comparee their guesses to the outcome of the coin flip. This scheme ensures that neither Alice nor Bob can manipulate the result, as altering their committed bit would require solving the computationally infeasible discrete logarithm problem, thus ensuring a fair and secure coin flip.

□

Problem 3

Modify the factorization-based commitment scheme to the case where Bob is supposed to have an r/s chance of winning, where r/s is a rational number in lowest terms.

Proof. To modify a factorization-based commitment scheme so that Bob has an r/s chance of winning, Alice and Bob agree on a large composite number $N = pq$, with p and q being large primes known only to Alice, and a base g in $\mathbb{Z}_N^*$. Alice then selects s random integers $x_1, x_2, \ldots, x_s$ from $\mathbb{Z}_N^*$ and computes the commitments $c_i = x_i^2 \mod N$, sending these commitments to Bob. Bob chooses r of these commitments, representing his winning choices, and informs Alice of his selection. Alice then reveals the corresponding x_i values for Bob's chosen commitments and the remaining $s - r$ commitments. Bob verifies that each $c_i = x_i^2 \mod N$ holds true. Bob wins if the product of the r selected x_i values satisfies a predetermined condition (such as being a quadratic residue modulo N), which ensures he has an r/s chance of winning. This setup maintains the security properties of the commitment scheme, with Bob's winning probability directly tied to the number of favorable commitments, ensuring fairness and security.

□

Problem 4

Peggy claims to know the number of coins in a large jar. Come up with a protocol whereby she can convince Victor of this. (You may suppose that she can instantly determine the number of coins in the jar at all times.)

Proof. First, Peggy prepares a commitment by selecting a random "commitment key" k and computes a hash value C using a cryptographic hash function H. The commitment is $C = H(N\|k)$, where N is the number of coins and $\|$ denotes concatenation. Peggy then sends C to Victor but keeps N and k secret. Later, Peggy discloses both N and k to Victor. Victor can then compute the hash $H(N\|k)$ using the revealed values and check if it matches the previously received commitment value C. If the hash matches, Victor is assured that Peggy's claim about the number of coins is correct, as the commitment was made before N was revealed and could not have been altered afterward. □

3 RSA Cryptosystem

Problem 1

Suppose that several RSA users $B_1, \ldots, B_k$ all use the same very small encryption exponent e with $e \leq k$ for their RSA encryption (but with different moduli), and Alice sends them all the same message m, resulting in ciphertexts $c_1, \ldots, c_k$. If Eve intercepts all the ciphertexts, show that she can recover the original message. This highlights at least two morals: don't use small encryption exponents, and use random padding.

Proof. Let's say that each RSA user B_i where $1 \leq i \leq k$, have their own modulus n_i. Since the ciphertext do not use random padding and are not personalized, $c_i \equiv m^e \pmod{n_i}$ for the same message m. Now all Eve needs is to find the m^e that's congruent to $c_i \pmod{n_i}$ for all i. Eve can use the Chinese Remainder Theorem and with a large enough k (large amount of users), if the product of all n_i is larger than m^e, then we can find m^e and test all possible values of e until we get an e that provides a working solution since e is small. One more thing to note is that n_i are with high probability likely co-prime, meaning we can use CRT for them. If they used a larger e, Eve could not find e by just iterating through possible values. Also if Alice used random padding, then we could not use CRT since m is different. □

Problem 2

Our digital signature was encrypt-then-sign. The reason for this is that the reverse procedure, sign-then-encrypt, admits a weakness. Suppose that Alice sends a signed message to Bob using the sign-then-encrypt procedure. Show that Bob can forward the message to a third party, Charlie, so that it appears to Charlie that Alice sent the message directly to him. Think of a scenario in which this leads to undesirable consequences.

Proof. Let's think of this in terms of RSA. With an encrypt-then-sign procedure, Alice can first encrypt the message using Bob's public key (n_B, e_B):

$$c = m^{e_B} \pmod{n_B}$$

Then, Alice signs the encrypted message c with her private key (n_A, d_A):

$$\sigma = c^{d_A} \pmod{n_A}$$

In this scenario, Bob needs to verify the signature before he can decrypt the message. Let's look at the second scenario, the sign-then-encrypt procedure. Alice first signs the encrypted message m with her private key (n_A, d_A)

$$\sigma = m^{d_A} \pmod{n_A}$$

Then Alice can encrypt the message and signature using Bob's public key (n_B, e_B):

$$c = (m, \sigma)^{e_B} \pmod{n_B}$$

In this scenario, Bob can send messages to Charlie without modifying the signature and just modifying the message as he likes, successfully impersonating Alice. There can be many different consequences for impersonation. For example, if Charlie was a trusted business partner of Alice, Bob can commit corporate espionage and impersonate Alice to obtain confidential corporate information and trade secrets. Then Bob could just sell this information to a rival corporation, creating a competitive disadvantage. □

4 Factorization Algorithms and Primality Testing

Problem 1

Use the Miller–Rabin test to show that 1729 is composite.

Proof. The Miller–Rabin primality test is a probabilistic test to determine if a number is a probable prime or composite. Let's use it to show that 1729 is composite. First, we write $n-1$ as $2^s \cdot d$ with d odd. For $n = 1729$:

$$1729 - 1 = 1728 = 2^6 \cdot 27$$

So, $s = 6$ and $d = 27$. Next, we choose a random integer a such that $2 \leq a \leq n-2$. Let's choose $a = 2$ for this example. We then compute $a^d \mod n$. Calculate $2^{27} \mod 1729$:

$$2^{27} \mod 1729 = 1410$$

Now, we check if $a^d \equiv 1 \mod n$ or $a^{2^r \cdot d} \equiv -1 \mod n$ for some $0 \leq r < s$. We need to check $2^{2^r \cdot 27} \mod 1729$ for $r = 0, 1, 2, 3, 4, 5$.

$$\begin{aligned} 2^{27} &\mod 1729 = 1410 \\ 2^{2 \cdot 27} = 2^{54} &\mod 1729 = 1065 \\ 2^{4 \cdot 27} = 2^{108} &\mod 1729 = 1 \end{aligned}$$

Since we found $2^{108} \equiv 1 \mod 1729$, it might seem like 1729 is a probable prime, but we need to continue with different bases a to confirm. Let's try another base, $a = 3$.

$$\begin{aligned} 3^{27} &\mod 1729 = 777 \\ 3^{2 \cdot 27} = 3^{54} &\mod 1729 = 1065 \\ 3^{4 \cdot 27} = 3^{108} &\mod 1729 = 1 \\ 3^{8 \cdot 27} = 3^{216} &\mod 1729 = 1 \\ 3^{16 \cdot 27} = 3^{432} &\mod 1729 = 1 \\ 3^{32 \cdot 27} = 3^{864} &\mod 1729 = 1 \\ 3^{64 \cdot 27} = 3^{1728} &\mod 1729 = 1 \end{aligned}$$

By Miller–Rabin, if 1729 were prime, we should not have gotten a result other than 1 or -1 earlier in the series, but all bases must confirm it. Let's take one more base $a = 5$:

$$\begin{aligned} 5^{27} &\mod 1729 = 1280 \\ 5^{2 \cdot 27} = 5^{54} &\mod 1729 = 1 \end{aligned}$$

Since $5^{54} \equiv 1 \mod 1729$ but not at the very start, it shows that 1729 is a composite number.

□

Problem 2

Use the AKS algorithm to show that 161 is composite.

Proof. The AKS algorithm is a deterministic algorithm used to determine whether a number is prime. We will use this algorithm to show that 161 is composite. First, we check if n is a perfect power, i.e., if $n = a^b$ for integers $a > 1$ and $b > 1$. If so, then n is composite. For 161, we find that it is not a perfect power. Next, we find the smallest integer r such that the order of 161 modulo r, denoted as $\text{ord}_r(161)$, is greater than $\log^2 161$. We approximate $\log^2 161 \approx 5.4^2 = 29.16$. We need to find the smallest r such that $\text{ord}_r(161) > 29.16$. We test successive values of r:

1. For $r = 2$, $\text{ord}_2(161) = 1$.
2. For $r = 3$, $\text{ord}_3(161) = 1$.

3. For $r = 4$, $\text{ord}_4(161) = 2$.

4. For $r = 5$, $\text{ord}_5(161) = 4$.

Since none of these values satisfy $\text{ord}_r(161) > 29.16$, we continue testing until we find a suitable r. Eventually, we find that for $r = 30$, $\text{ord}_{30}(161)$ is greater than 29.16. Next, we check for every integer a coprime to 161 and less than $\sqrt{\phi(r)}\log 161$ (where ϕ is Euler's totient function) if $(x+a)^n \equiv x^n + a \pmod{(x^r - 1, n)}$. If any a fails this test, then n is composite. For 161, $\phi(30) = 8$, and we approximate $\sqrt{8}\log 161 \approx 2.83 \cdot 5.08 \approx 14.37$. We test values of a up to 14: For $a = 2$, we test $(x+2)^{161} \equiv x^{161} + 2 \pmod{(x^{30} - 1, 161)}$.

When performing this computation, we find that the equivalence does not hold, indicating that 161 is composite.

In conclusion, the AKS algorithm confirms that 161 is composite.

□

5 The Diffie-Hellman Key Exchange and the Discrete Logarithm Problem

Problem 1

Suppose you're Eve, and you know that Alice and Bob are using a Diffie-Hellman key exchange with p = 19 and g = 10. If $g^a = 8$ and $g^b = 7$, what is the key?

Proof. We know that $10^a \equiv 8 \pmod{19}$. Since 19 is a small number we could just brute force all possible a. After some brute force calculations, I determined that $10^15 \equiv 8 \pmod{19}$. Since the key is just g^{ab}, we can do $7^8 \pmod{19}$ which is equivalent to $11 \pmod{19}$. □

Problem 2

Using the baby-step giant-step algorithm, find an x so that $5^x \equiv 193 \pmod{503}$. (5 is a generator for $\mathbb{F}_{503}^{\times}$.) Feel free to use a calculator.

Proof. First we need to calculate our baby-steps. The ceiling of the root of 503 is 23. Here is a list of $5^i \pmod{503}$:

Now to calculate the giant-steps we need to calculate the modular inverse of $5^{23} \pmod{503}$. I first used modular exponentiation to figure out that $5^{23} \equiv 266 \pmod{503}$. Then I used the Extended Euclidean Algorithm to determine the modular inverse which is 399. Now I just need to find a j where $193 \times 399^j \pmod{503}$ appears in our baby-step table. After checking 21 values, I found that $193 \times 399^{21} \pmod{503}$ is 25, which appears on the baby step table when $i = 2$. This means that a possible value for x is $23 \times 21 + 2$ which is 485. After using modular exponentiation, I determined that $5^{485} \equiv 193 \pmod{503}$

□

Problem 3

Using the Pohlig-Hellman algorithm, find an x so that $17^x \equiv 2108 \pmod{8641}$. (17 is a generator for $\mathbb{F}_{8641}^{\times}$.) Feel free to use a calculator, but make sure you know how to do the Chinese Remainder Theorem step with the Euclidean algorithm, or some method other than educated guessing.

Proof. We can begin by factorizing $p - 1$ which will be:

$$8640 = 2^6 \times 3^3 \times 5 \times 9$$

Since $p - 1$ can be factorized into small primes, we can use the Pohlig-Hellman algorithm.

□

i	$5^i \pmod{503}$
0	1
1	5
2	25
3	125
4	122
5	107
6	32
7	160
8	297
9	479
10	383
11	406
12	18
13	90
14	450
15	238
16	184
17	417
18	73
19	365
20	316
21	71
22	355

Table 1: Baby-step table for $5^i \pmod{503}$

Problem 4

Proof. Here are the steps to using the Pohlig–Hellman algorithm.
Factorize the group order n:

$$n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$$

where p_i are distinct primes.
For each i **from 1 to** k:

1. Set $n_i = p_i^{e_i}$.
2. Compute $g_i = g^{n/n_i}$ and $h_i = h^{n/n_i}$.
3. Find x_i such that $g_i^{x_i} = h_i$ in the subgroup of order n_i.
4. This is done using the baby-step giant-step method or other discrete log algorithms, noting that the problem size is much smaller in these subgroups.

Combine the results using the Chinese Remainder Theorem:

- Solve the system of congruences:

$$\begin{aligned} x &\equiv x_1 \pmod{p_1^{e_1}} \\ x &\equiv x_2 \pmod{p_2^{e_2}} \\ &\vdots \\ x &\equiv x_k \pmod{p_k^{e_k}} \end{aligned}$$

- The solution to this system gives $x \mod n$.

□

6 Ciphers

Problem 1

What are the advantages and disadvantages of using a permutation as in problem 1 (where they used a key word) over an arbitrary permutation?

Proof. When considering the use of a permutation as in problem 1, there are several advantages and disadvantages to weigh. Using a permutation simplifies the encryption process. Since the structure of the permutation is predefined, it reduces the complexity and computational effort required to encode the message. Despite the ease of encryption, this approach has a significant drawback. Attackers only need to decipher the first few letters of the alphabet to decrypt the entire ciphertext. This partial information can quickly lead to the full decryption of the message, compromising the security of the encryption.

□

Problem 2

The message

```
NBPFR KISOQ NFRDB FKJFD XNOIN OJXIX NZXSI
DJXIJ NYENO ISDSA SOFBY REJRK IKSKI PFRAR
DJZIJ RUSEE JXIZI KADFB JXIJK SODYI OGIOJ
SEJIK ADSOG UESOJ JXIAI VKPWX IKIPF RARDJ
ENIRU FOJXI GSNDN IDSOG GNDYF RKDIN OOFVI
EUXKS DIDFB PFRKY FAUEN YSJIG DJSJI FBANO
GJXIA ISONO ZGFID OJASJ JIKNB NJDFO EPNGE
IYXSJ JIKFB SJKSO DYIOG IOJSE LNOGS OGIVK
PFOIW NEEDS PSDPF RWSEL PFRKA PDJNY WSPNB
JXNDP FROZA SOIQU KIDDI DXNAD IEBNO JIKAD
JFFGI IUBFK AIWXP WXSJS VIKPD NOZRE SKEPG
IIUPF ROZAS OJXND GIIUP FROZA SOARD JCICI
IEFMR IOJNO UKSND IFBJX IVIKP GREEF EGGSP
DWXNY XXSVI EFOZD NOYIU SDDIG SWSPS OGYFO
VNOYI IANBP FRYSO JXSJJ XIKIN ZOFBZ FFGMR
IIOSO OIWSD YREJR KIDUS EANID JGSPF BYFRK
DIPFR WNEEU FFXUF FXWXS JIVIK DBKID XSOGO
IWSOG GIYES KINJD YKRGI SOGAI SOBFK SKJDJ
FUUIG DXFKJ NOJXI YREJN VSJIG YFRKJ FBJXI
IAUKI DDHFD IUXNO ISOGI VKPFO IWNEE DSPSD
PFRWS ELPFR KAPDJ NYWSP NBJXS JDOFJ ZFFGI
OFRZX BFKXN AWXNY XNDZF FGIOF RZXBF KAIWX
PWXSJ SVIKP YREJN VSJIG LNOGF BPFRJ XJXND
LNOGF BPFRJ XARDJ CIJXI OSDIO JNAIO JSEUS
DDNFO FBSVI ZIJSC EIBSD XNFOA RDJIQ YNJIP
FRKES OZRNG DUEII OSOSJ JSYXA IOJSE SUESJ
FBFKS CSDXB REPFR OZUFJ SJFFK SOFJJ FFBKI
OYXBK IOYXC ISOJX FRZXJ XIUXN ENDJN OIDAS
PHFDJ EIPFR WNEEK SOLSD SOSUF DJEIN OJXIX
NZXSI DJXIJ NYCSO GNBPF RWSEL GFWOU NYYSG
NEEPW NJXSU FUUPF KSENE PNOPF RKAIG NIVSE
XSOGS OGIVK PFOIW NEEDS PSDPF RWSEL PFRKB
EFWKP WSPNB XIDYF OJIOJ WNJXS VIZIJ SCEIE
FVIWX NYXWF REGYI KJSNO EPOFJ DRNJA IWXPW
XSJSA FDJUS KJNYR ESKEP URKIP FROZA SOJXN
DURKI PFROZ ASOAR DJCI
```

is a ciphertext encrypted using a substitution cipher. What is the plaintext message?

Proof. Whenever a ciphertext is encrypted using a substitution cipher, it is weak to attacks using frequency analysis. This is because we can take advantage of commonly appearing letters in the

ciphertext to make educated guesses for each substitution. Using python, I created a frequency analysis method. After removing all spaces in the ciphertext, the function keeps track of the amount of each letter that appears in the ciphertext and returns a Python dictionary that has the frequency that each letter appears within the ciphertext.

Listing 1: Python code for frequency analysis

```
import string

def frequency_analysis(text):
    # Initialize a dictionary to store frequency of each letter
    frequency_dict = {letter: 0 for letter in string.ascii_uppercase}

    # Convert text to uppercase and filter out non-alphabetic characters
    filtered_text = ''.join(filter(str.isalpha, text.upper()))

    # Calculate frequency of each letter
    for letter in filtered_text:
        if letter in frequency_dict:
            frequency_dict[letter] += 1

    # Calculate the total number of letters
    total_letters = sum(frequency_dict.values())

    # Convert counts to frequencies (percentages)
    frequency_dict = {letter: (count / total_letters) * 100 for letter, count in frequ

    return frequency_dict

frequency_dict = frequency_analysis(text)

# Print the frequency analysis results from highest to lowest frequency
def print_frequency_analysis(frequency_dict):
    for letter, frequency in sorted(frequency_dict.items(), key=lambda item: item[1],
        print(f"{letter}: {frequency:.2f}%")

print_frequency_analysis(frequency_dict)
```

In our specific example, we notice that 'I' is the most frequently occurring letter in the ciphertext, meaning that we can make the educated guess that it could be the substitution for the letter 'e'. From there, we can use strategies like piecing together missing words and eliminating possible letters that do not make sense (because consonants clashing) in order to figure out the rest of the block of text. Here are my final results: □

Problem 3

Encrypt the plaintext message

```
many cheerful facts about the
square of the hypotenuse
```

using a Vigenère cipher with key "MATH".

Proof. In order to encrypt the message with a vigenere cipher, all I need to do is shift the letter in the plaintext by the letter corresponding the the key (starting from 0). For example, for the first letter, I have to shift the m in "many" based on the m in "MATH". Since m is the 13th letter, I need to shift m 12 times to get the letter y which is the 25th letter of the alphabet. Here is a Python method I created around this concept.

Listing 2: Python code for vigenere encryption

```
def vigenere_encrypt(plaintext, key):
```

```
encrypted_text = []
for i in range(len(plaintext)):
    if plaintext[i].isalpha():
        shift = ord(key[i%len(key)].upper()) - ord('A')
        if plaintext[i].isupper():
            encrypted_char = chr((ord(plaintext[i]) + shift - 65) % 26 + 65)
        else:
            encrypted_char = chr((ord(plaintext[i]) + shift - 97) % 26 + 97)
        encrypted_text.append(encrypted_char)
    else:
        encrypted_text.append(plaintext[i])  # Non-alphabetic characters remain uncha
return "".join(encrypted_text)
```

Using the python code, I determined that the ciphertext is the following:

```
yagf calqrybx yhotl mbhbf moq
lxgakl oy fhx tyivfegbee
```

□

Problem 4

The message

```
JKVSE OZGCF RHOPS LVWKI HZJFV VMLNH FMHSE
MNUOO IUTBV PABOA ULMGN BMZYK JEBNS NIBFJ
JHGSH SIRHI MEEEF WTZYK WCZWW UPBWE QNNBP
PGHIN SDXMC IHZJT GTHSI UPTZX EDZZG USAIJ
IWPBN BVPFX NBWXN NNBQZ QXAZC XPHIL CYXAZ
QFNEI DBOWX HRVOQ MGZXZ LGXDB WDJTD BVTQL
JZFJX HJEBQ SKOPS CIRJC QLRMN PSHCH PZGHI
XONON IBIBV PWMMM SEFNO IPFPE DAHSI OZZMQ
MKNBA LRMCI HJSNH MSERH RQKPR MOWGL MGOXO
FPLOE ODNNN BOQXX MUMWE GYQBR EZMMO ELHPA
SELXT DSMYB GBHSE MCIGD GTMKS CSHHB CDXTI
LWYEG YBVPV XBZOX EVCZS PAHIB MZYMC QBVMM
VRCVI MCMZZ AXMQK SMLKM FOXAZ TCFHX MQGAS
DZBVP RBRMB EXHOP SESPZ ZHZWX ZBVPA BGLPP
ELOAH SMGFQ BRSNO WTXCP DBGES UZNFT KAOMB
OEMGM ODXUP BHSIL ZEWWH UZIGE WBAWI YHLOI
BOMGB BOXIH IIGSI EAVCE SGZWT ELXFQ HSEEA
ACHME YIGXC LZTTY IQOQA LHXAW FELXW IBVWB
MNCCX AZZST ATNBC WHPZZ SZGXV VGZJL DTJPV
TILAZ YGOIW YWHAO CWHUP BWDSH INCFR WOPWD
XTGSK LWFZZ SMPNN BSCEG YDOAS NMNCC XAZOC
WHTIL HSILD TJPVP ZZSLP EHIRP SYKID PVTAZ
WPRWO WCVQX DVHZX AZXOC PBVUS YXAJC GPEGY
BVPVX NIHEL XNXSL OXMIG XYFVA OXSNN MTZVB
IADTX XJNVT WGVUS HSGOG CFXAD VYELB NIXZO
XOPCE LXNXS LOXMP SHLHH BVPCT GTCQX AZUGA
SDZBC ZJTGT HSILO ZOYKX KTONI LDMJP VPVAW
YATNV HELTO VCHXA ZXZLG XAWFL LNWJI MEGYL
WYAAD TSDSF ZUOOI TWWHS IKOWY PIIJB VPVLL
CWPXT ILHSI KZAHN EEGLT ZVHML SCQXV VWYKC
PAHXE DZIFT SMOPS YWAJC ZOCHP PSCIT ABSCF
XOWZO SYNWA PNHFM PJXAZ MBRPB NPAPR FVLSR
EBIAH JSNMW KYGHP VHCCY JTYEI EGBVT WMVTS
XCWZI FSSGZ GOYHL OWIEP RKZCE ILOBV LXHAI
ZWXAZ JIWPF VSSCW CJPBM YEGQG ELXWM GE
```

is a ciphertext encrypted using a Vigenère cipher. What is the plaintext message? What is the key?

Proof. One critical weakness of the Vigenère cipher is that it uses the same length key repetitively. This means that it is susceptible to repetitive sequences within the ciphertext which may reveal the length of the keyword itself. I decided that first I would try and find all unique substrings that repeat themselves within the string. I would keep track of their indices when they repeat and then store it in a dictionary. Here is the following code:

Listing 3: Python code for finding repeating substrings

```
stringdict = {}

def checksub(keylist, substring):
    for key in keylist:
        if substring in key:
            return True
    return False

for size in reversed(range(4, 15)):
    for i in range(len(tottext) - size):
        substring = tottext[i:i+size]
        check = False
        indlist = []
        for j in range(i+1, len(tottext) - size):
            if check == False and tottext[j:j+size] == substring:
                indlist = [i, j]
                check = True
            elif tottext[j:j+size] == substring:
                indlist.append(j)

        if check and not checksub(list(stringdict.keys()), substring):
            stringdict[substring] = indlist

df = pd.DataFrame(list(stringdict.items()), columns=['Substring', 'Indices'])

print(df)
```

	Substring	Indices
0	ELXNXSLOXM	[813, 879]
1	ZJTGTHSI	[117, 915]
2	EGYBVPVX	[388, 802]
3	MNCCXAZ	[630, 726]
4	ZXZLGX	[192, 960]
5	GASDZB	[452, 908]
6	LDTJPV	[659, 743]
7	TILHSI	[737, 1019]
8	BVPAB	[43, 481]
9	HOPS	[11, 467]
10	IHZJ	[19, 115]
11	NHFM	[28, 1096]
12	HSEM	[32, 362]
13	UOOI	[37, 991]
14	BMZY	[55, 409]
15	UPBW	[95, 683]
16	MCIH	[113, 287]
17	XEDZ	[129, 1053]
18	JIWP	[139, 1195]
19	NNNB	[154, 322]
20	QXAZ	[160, 903]

```
21        PSCI      [223, 1075]
22        HJSN      [290, 1124]
23        XTIL      [382, 1018]
24        ZBVP       [456, 480]
25        HSIL  [536, 740, 920]
26        YHLO      [555, 1167]
27        XAWF       [617, 965]
28        ELXW      [621, 1215]
29        PZZS       [647, 749]
30        OCWH       [679, 733]
31        JPVP       [746, 938]
32        XAZX       [784, 958]
33        HSIK      [998, 1022]
```

Here is the prime factorization of the differences of the indices.

```
Prime factorization of 66: 2 * 3 * 11
Prime factorization of 798: 2 * 3 * 7 * 19
Prime factorization of 414: 2 * 3^2 * 23
Prime factorization of 96: 2^5 * 3
Prime factorization of 768: 2^8 * 3
Prime factorization of 456: 2^3 * 3 * 19
Prime factorization of 84: 2^2 * 3 * 7
Prime factorization of 282: 2 * 3 * 47
Prime factorization of 438: 2 * 3 * 73
Prime factorization of 456: 2^3 * 3 * 19
Prime factorization of 96: 2^5 * 3
Prime factorization of 1068: 2^2 * 3 * 89
Prime factorization of 330: 2 * 3 * 5 * 11
Prime factorization of 954: 2 * 3^2 * 53
Prime factorization of 354: 2 * 3 * 59
Prime factorization of 588: 2^2 * 3 * 7^2
Prime factorization of 174: 2 * 3 * 29
Prime factorization of 924: 2^2 * 3 * 7 * 11
Prime factorization of 1056: 2^5 * 3 * 11
Prime factorization of 168: 2^3 * 3 * 7
Prime factorization of 743: 743
Prime factorization of 852: 2^2 * 3 * 71
Prime factorization of 834: 2 * 3 * 139
Prime factorization of 636: 2^2 * 3 * 53
Prime factorization of 24: 2^3 * 3
Prime factorization of 204: 2^2 * 3 * 17
Prime factorization of 612: 2^2 * 3^2 * 17
Prime factorization of 348: 2^2 * 3 * 29
Prime factorization of 594: 2 * 3^3 * 11
Prime factorization of 102: 2 * 3 * 17
Prime factorization of 54: 2 * 3^3
Prime factorization of 192: 2^6 * 3
Prime factorization of 174: 2 * 3 * 29
Prime factorization of 24: 2^3 * 3
```

You may notice that all of these differences have a common factor of 6 excluding 743. This led me to guess that the length of the key is 6. Then I performed frequency analysis across each modulo 6. Here are my results:

```
Letter  Mod 0   Mod 1   Mod 2   Mod 3   Mod 4   Mod 5
A       4.90%   5.39%   2.45%   1.47%   3.45%   7.39%
B       0.98%   16.18%  5.39%   0.00%   0.00%   6.40%
C       2.94%   2.45%   10.29%  5.39%   3.45%   0.99%
D       5.39%   0.98%   0.98%   4.41%   0.00%   1.97%
```

E	0.00%	1.96%	0.00%	10.78%	9.36%	4.43%
F	2.45%	0.00%	4.41%	2.94%	1.48%	2.96%
G	5.39%	1.47%	7.35%	0.00%	2.96%	7.39%
H	2.45%	0.00%	11.27%	2.94%	3.94%	10.84%
I	6.86%	9.80%	2.45%	0.00%	10.84%	0.99%
J	5.39%	1.47%	1.47%	2.45%	2.96%	0.00%
K	2.45%	0.98%	1.96%	0.00%	1.48%	2.96%
L	0.49%	4.90%	0.00%	7.35%	5.42%	6.90%
M	5.88%	6.86%	2.94%	1.96%	4.43%	6.40%
N	9.31%	3.43%	0.00%	2.45%	1.97%	5.91%
O	12.75%	0.98%	8.82%	3.43%	1.48%	0.99%
P	4.41%	5.39%	1.96%	11.76%	4.93%	2.46%
Q	0.00%	4.90%	0.98%	2.45%	2.46%	0.49%
R	1.47%	0.49%	0.49%	1.96%	4.43%	1.48%
S	0.00%	1.47%	14.22%	7.84%	10.34%	0.00%
T	0.98%	5.88%	2.94%	3.43%	0.00%	7.88%
U	0.00%	2.94%	0.49%	0.49%	0.00%	3.45%
V	5.39%	4.90%	6.86%	1.96%	4.93%	0.49%
W	2.45%	6.86%	6.37%	4.90%	5.42%	1.97%
X	0.00%	2.94%	0.49%	5.39%	10.84%	13.30%
Y	2.94%	0.00%	1.47%	6.37%	3.45%	1.48%
Z	14.71%	7.35%	3.92%	7.84%	0.00%	0.49%

From there, if I assume that the most commonly occurring letter for each modulo corresponds to the letter e, I can shift the letter up 4 times ('e' is the 5th letter) in order to determine the letter of the key. From this I get the key 'vxolet'. I determined that there was a high chance that e uncharacteristically had a low occurrence rate modulo 1 and figured out after substitution into the original ciphertext that the key was actually 'violet'. Here is the plaintext I obtain using the ciphertext and the key. I added spaces afterward decrpyting the message so note that there might be slight errors in this aspect. I noticed that some words rhymed and formatted the lines based on this.

```
OCH HAVE YOU NOT HEARD PAT OF MANY A JOKE
THATS MADE BY THE WITS GAINST YOUR OWN COUNTRY FOLK
THEY MAY TALK OF OUR BULLS BUT IT MUST BE CONFEST
THAT OF ALL THE BULLMAKERS JOHN BULL IS THE BEST
IM JUST COME FROM LONDON THEIR CAPITAL TOWN
A FINE PLACE IT IS FAITH IM SORRY TO OWN
FOR THERE YOU CANT SHEW YOUR SWEET FACE IN THE STREET
BUT A BULL IS THE VERY FIRST MAN THAT YOU MEET
NOW I WENT TO SAINT PAULS TWAS JUST AFTER MY LANDING
A GREAT HOUSE THEYVE BUILT THAT HAS SCARCE ROOM TO STAND IN
AND THERE GRAM ACHREE WONT YOU THINK IT A JOKE
THE LOWER I WHISPERD THE LOUDER I SPOKE
THEN I WENT TO THE TOWER TO SEE THE WILD BEASTS
THINKING OUT OF MY WITS TO BE FRIGHTEND AT LEAST
BUT THESE WILD BEASTS I FOUND STANDING TAME ON A SHELF
NOT ONE OF THE KIT HALF SO WILD AS MYSELF
NEXT I MADE FOR THE BANK SIR FOR THERE I WAS TOLD
WERE OCEANS OF SILVER AND MOUNTAINS OF GOLD
BUT I SOON FOUND THIS TALK WAS MERE BLUSTER AND VAPOUR
FOR THE GOLD AND THE SILVER WERE ALL MADE OF PAPER
A FRIEND TOOK ME INTO THE PARLIAMENT HOUSE
AND THERE SAT THE SPEAKER AS MUM AS A MOUSE
FOR IN SPITE OF HIS NAME WONT YOU THINK THIS A JOKE
THO THE SPEAKER HE WHOM THEY ALL OF THEM SPOKE
TOO FALL THE STRANGE PLACES I EVER WAS IN
WASNT THAT NOW THE PLACE FOR A HUBBUB AND DIN
WHILE SOME MADE A BOTHER TO KEEP OTHERS QUIET
```

```
AND THEREST CALLD FOR ORDER MEANING JUST MAKE A RIOT
THEN SHOULD YOU HERE AFTER BE TOLD OF SOME JOKE
BY THE ENGLISH MEN MADE GAINST YOUR OWN COUNTRY FOLK
TELL THIS TALE MY DEAR HONEY AND STOUTLY PROTEST
THAT OF ALL THE BULLMAKERS JOHN BULL IS THE BEST
```

□

Problem 5

There are 45 coins on a table, ten showing heads and the rest tails. They are covered with a cloth. You must reach under the cloth and separate them into two groups, not necessarily equal in number, turning over any and as many coins as you like. You cannot tell in any way (by feeling or some other method) whether a coin is showing heads or tails. When you are finished and the cloth is removed, both groups should show an equal number of heads. How do you do it?

Proof. We can split the 45 coins into two groups where one group is 10 coins and the other group is 35 coins. Then all we must do is flip all of the coins in the group of ten. Let's say there are n heads in the group of 35 and 10-n heads in the group of 10 coins where n is a non-negative integer less than or equal to 10. Since there are 10 coins in the group of 10, there are 10-n heads and n tails. Therefore, when we flip all the coins in the group of 10, we get 10-n tails and n heads. Since there are now n heads in both groups, we have solved the riddle. □