① (a) $1 - \mathbb{P}(X_1 \neq 6 \cap X_2 \neq 6 \cap X_3 \neq 6 \cap X_4 \neq 6) = 1 - \mathbb{P}(X_1 \neq 6)^4 = 1 - \left(\frac{5}{6}\right)^4$

(b) Bet 1: win := guessing first 6 coin tosses

$\Pr(\text{win}) = \left(\frac{1}{2}\right)^6 \Rightarrow \mathbb{E}[\text{gain}] = \Pr(\text{win}) \cdot 50 - \Pr(\text{loss}) \cdot 1$

$= \frac{1}{26} \cdot 50 - \left(1 - \frac{1}{26}\right) \cdot 1 = -\frac{13}{64}$

Bet 2: win := guessing 7 coin tosses as a subsqce of length 10 tosses

Let A_1 := 7 coin toss guess appears in the first 7 positions

A_2 := " 2-8 "

A_3 := " 3-9 "

A_4 := " 4-10 "

$\Pr(\text{win}) = \Pr(A_1 \cup A_2 \cup A_3 \cup A_4) = \sum_{i=1}^{4} \Pr(A_i) - \sum_{i<j} \Pr(A_i \cap A_j) + \sum_{i<j<k} \Pr(A_i \cap A_j \cap A_k) - \Pr(A_1 \cap A_2 \cap A_3 \cap A_4)$

↓

This prob. is maximized when the events A_i's have no intersection, so $\Pr(\cdot \cap \cdot) = 0$.

BEST: So, if you are going for Bet 2, winning prob. is maximized with a guess that can appear exactly once as a subsqce of length 10 sqce.

Ex: HHTTHHT

$\Pr(\text{win}) = \sum_{i=1}^{4} \Pr(A_i) = 4 \cdot \frac{1}{2^7} = \frac{1}{32} \Rightarrow \mathbb{E}[\text{gain}] = \Pr(\text{win}) \cdot 50 - \Pr(\text{loss}) \cdot 1$

$= \frac{1}{32} \cdot 50 - \left(1 - \frac{1}{32}\right) \cdot 1 = \frac{19}{32}$.

② X_i, Y_i are indep. $\Leftrightarrow \Pr(X_i = x, Y_i = y) = \Pr(X_i = x) \Pr(Y_i = y) \quad \forall x, y \in \{0,1\}$

$P_1(X_1, Y_1)$: $P_1(0,0) = \frac{1}{2}$ $P_1(0,1) = 0$ $P_1(1,0) = 0$ $P_1(1,1) = \frac{1}{2}$

$\Rightarrow$ not indep.: $\Pr(X_i = 0, Y_i = 0) = \frac{1}{2} \neq \Pr(X_i = 0) \Pr(Y_i = 0) = \frac{1}{2} \cdot \frac{1}{2}$

$P_2(X_2, Y_2)$: $P_2(0,0) = \frac{1}{3}$ $P_2(0,1) = \frac{1}{3}$ $P_2(1,0) = 0$ $P_2(1,1) = \frac{1}{3}$

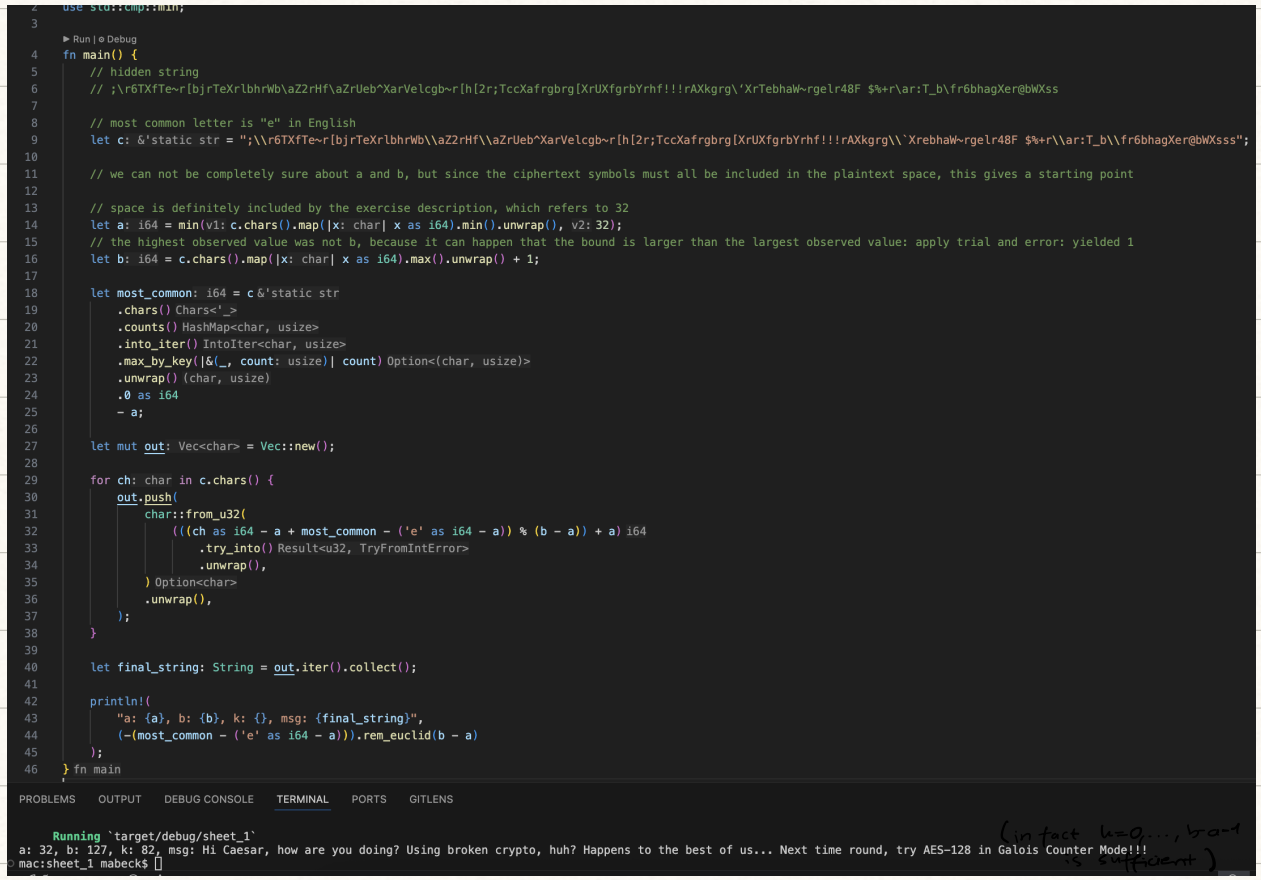
$\Rightarrow$ not indep.: $\Pr(X_i = 0, Y_i = 0) = \frac{1}{3} \neq \Pr(X_i = 0) \Pr(Y_i = 0) = \frac{2}{3} \cdot \frac{1}{3}$

$P_3(X_3, Y_3)$: $P_3(0,0) = \frac{1}{4}$ $P_3(0,1) = \frac{1}{4}$ $P_3(1,0) = \frac{1}{4}$ $P_3(1,1) = \frac{1}{4}$

$\Rightarrow X_3$ and Y_3 are independent.

Show that desired equality holds for all $(X_3, Y_3) \in \{0,1\}^2$.

3. In summary, fix some candidates for a and b, then find key k by brute force or frequency analysis or some other strategy.



```rust
use std::cmp::min;

▶ Run | ⚙ Debug
fn main() {
    // hidden string
    // ;\r6TXfTe~r[bjrTeXrlbhrWb\aZ2rHf\aZrUeb^XarVelcgb~r[h[2r;TccXafrgbrg[XrUXfgrbYrhf!!!rAXkgrg\'XrTebhaW~rgelr48F $%+r\ar:T_b\fr6bhagXer@bWXss

    // most common letter is "e" in English
    let c: &'static str = ";\\r6TXfTe~r[bjrTeXrlbhrWb\\aZ2rHf\\aZrUeb^XarVelcgb~r[h[2r;TccXafrgbrg[XrUXfgrbYrhf!!!rAXkgrg\\`XrebhaW~rgelr48F $%+r\\ar:T_b\\fr6bhagXer@bWXsss";

    // we can not be completely sure about a and b, but since the ciphertext symbols must all be included in the plaintext space, this gives a starting point

    // space is definitely included by the exercise description, which refers to 32
    let a: i64 = min(v1: c.chars().map(|x: char| x as i64).min().unwrap(), v2: 32);
    // the highest observed value was not b, because it can happen that the bound is larger than the largest observed value: apply trial and error: yielded 1
    let b: i64 = c.chars().map(|x: char| x as i64).max().unwrap() + 1;

    let most_common: i64 = c &'static str
        .chars() Chars<'_>
        .counts() HashMap<char, usize>
        .into_iter() IntoIter<char, usize>
        .max_by_key(|&(_, count: usize)| count) Option<(char, usize)>
        .unwrap() (char, usize)
        .0 as i64
        - a;

    let mut out: Vec<char> = Vec::new();

    for ch: char in c.chars() {
        out.push(
            char::from_u32(
                (((ch as i64 - a + most_common - ('e' as i64 - a)) % (b - a)) + a) i64
                    .try_into() Result<u32, TryFromIntError>
                    .unwrap(),
            ) Option<char>
            .unwrap(),
        );
    }

    let final_string: String = out.iter().collect();

    println!(
        "a: {a}, b: {b}, k: {}, msg: {final_string}",
        (-(most_common - ('e' as i64 - a))).rem_euclid(b - a)
    );
} fn main
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  GITLENS

    Running `target/debug/sheet_1`
a: 32, b: 127, k: 82, msg: Hi Caesar, how are you doing? Using broken crypto, huh? Happens to the best of us... Next time round, try AES-128 in Galois Counter Mode!!!
mac:sheet_1 mabeck$
```

(in fact k=0..., b-a sufficient)

(4) Physical security systems vary in how well they follow Kerckhoff's principle. When they rely too much on obscurity (like hidden camera replacement or weak locks mechanisms), they become unreliable.