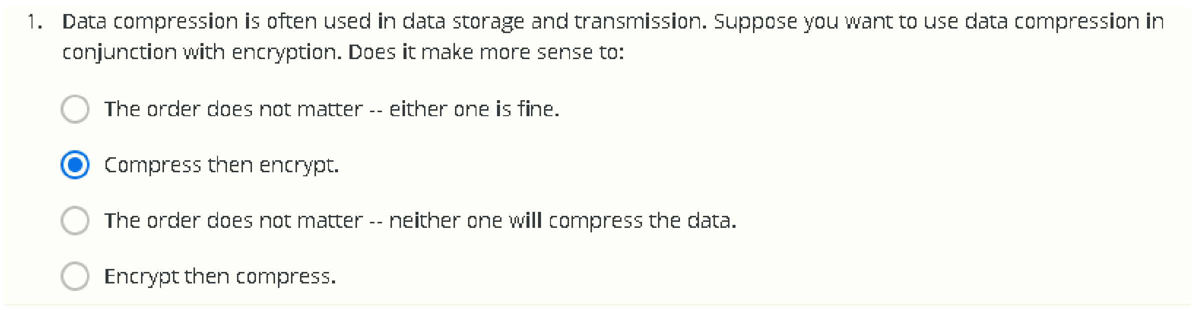
W1 Problem Set && Programming Assignment

Q1：

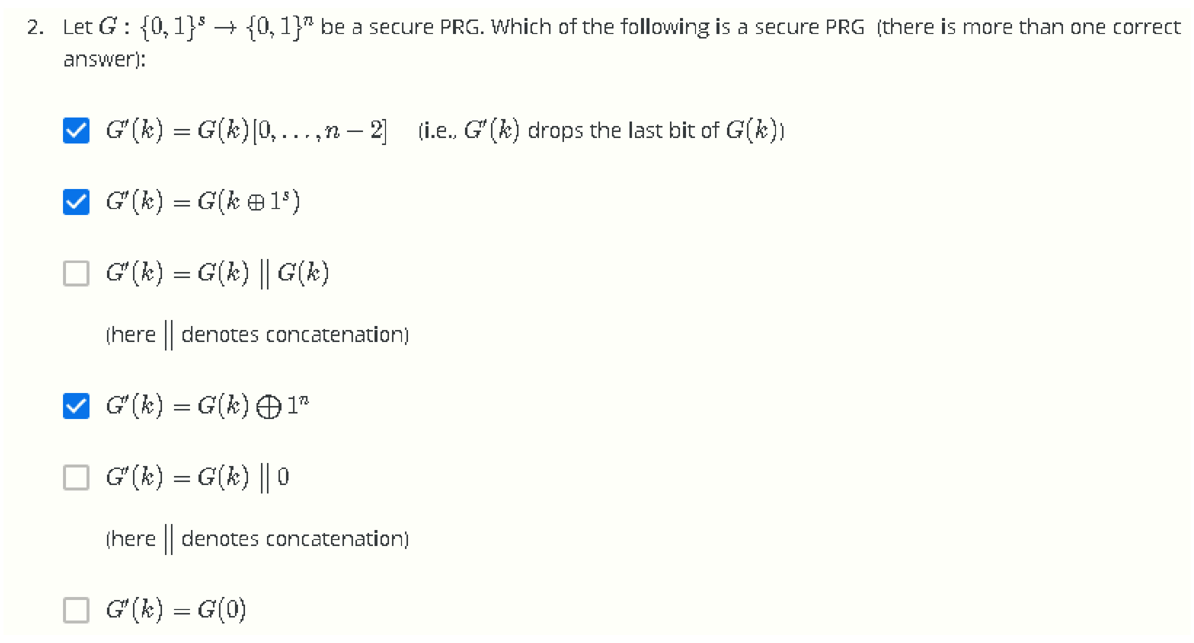


1. Data compression is often used in data storage and transmission. Suppose you want to use data compression in conjunction with encryption. Does it make more sense to:

- ○ The order does not matter -- either one is fine.
- [✓ Compress then encrypt.]
- ○ The order does not matter -- neither one will compress the data.
- ○ Encrypt then compress.

密文看上去像随机的字符串，因此应当在加密前就进行压缩

Q2：



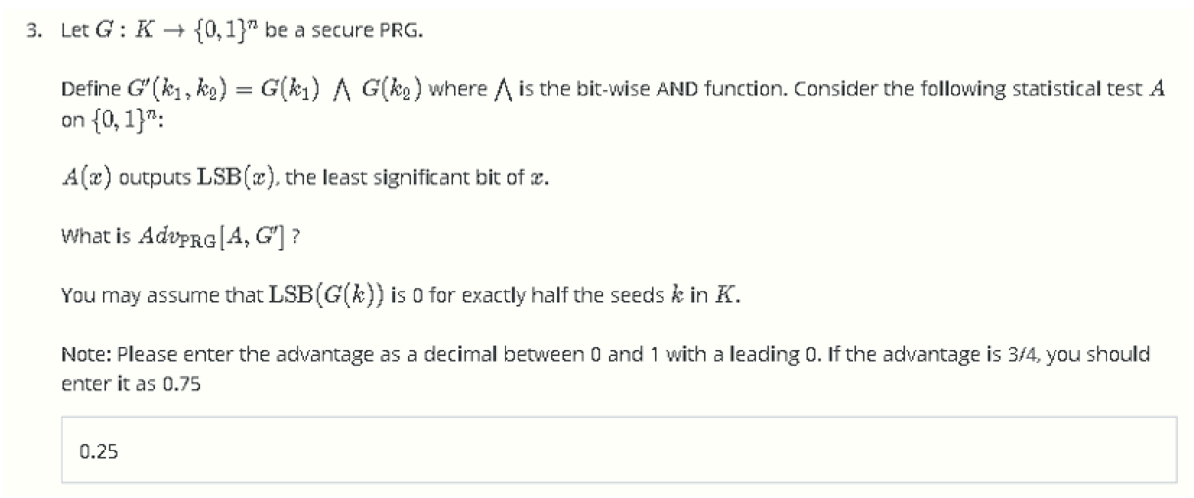
2. Let $G: \{0,1\}^s \to \{0,1\}^n$ be a secure PRG. Which of the following is a secure PRG (there is more than one correct answer):

- ✓ $G'(k) = G(k)[0, \ldots, n-2]$ (i.e., $G'(k)$ drops the last bit of $G(k)$)
- ✓ $G'(k) = G(k \oplus 1^s)$
- ☐ $G'(k) = G(k) \parallel G(k)$

  (here $\parallel$ denotes concatenation)
- ✓ $G'(k) = G(k) \bigoplus 1^n$
- ☐ $G'(k) = G(k) \parallel 0$

  (here $\parallel$ denotes concatenation)
- ☐ $G'(k) = G(0)$

Q3:



3. Let $G: K \to \{0,1\}^n$ be a secure PRG.

Define $G'(k_1, k_2) = G(k_1) \ \bigwedge \ G(k_2)$ where $\bigwedge$ is the bit-wise AND function. Consider the following statistical test A on $\{0,1\}^n$:

$A(x)$ outputs $\text{LSB}(x)$, the least significant bit of x.

What is $Adv_{\text{PRG}}[A, G']$?

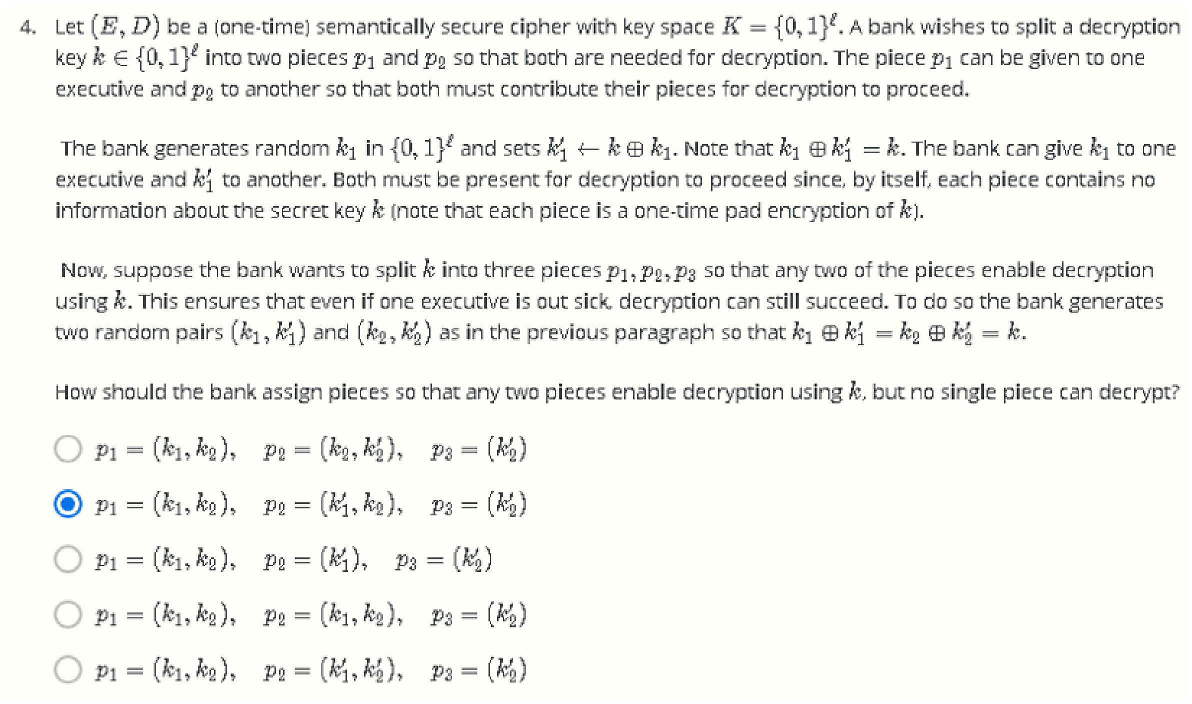
You may assume that $\text{LSB}(G(k))$ is 0 for exactly half the seeds k in K.

Note: Please enter the advantage as a decimal between 0 and 1 with a leading 0. If the advantage is 3/4, you should enter it as 0.75

0.25

$G(k_1)$和$G(k_2)$)同时为1时，A输出1

Q4:



4. Let (E, D) be a (one-time) semantically secure cipher with key space $K = \{0,1\}^\ell$. A bank wishes to split a decryption key $k \in \{0,1\}^\ell$ into two pieces p_1 and p_2 so that both are needed for decryption. The piece p_1 can be given to one executive and p_2 to another so that both must contribute their pieces for decryption to proceed.

The bank generates random k_1 in $\{0,1\}^\ell$ and sets $k_1' \leftarrow k \oplus k_1$. Note that $k_1 \oplus k_1' = k$. The bank can give k_1 to one executive and k_1' to another. Both must be present for decryption to proceed since, by itself, each piece contains no information about the secret key k (note that each piece is a one-time pad encryption of k).

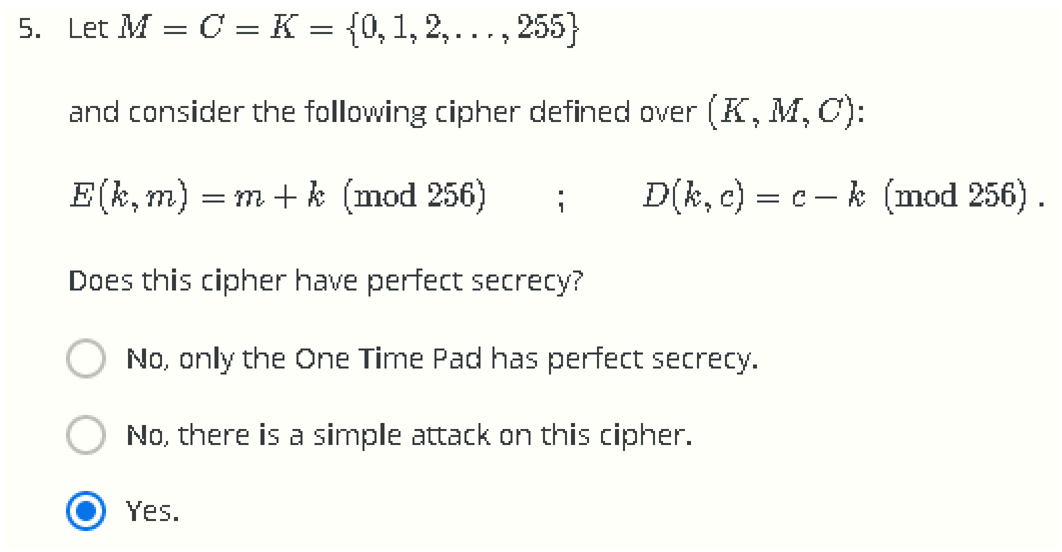
Now, suppose the bank wants to split k into three pieces p_1, p_2, p_3 so that any two of the pieces enable decryption using k. This ensures that even if one executive is out sick, decryption can still succeed. To do so the bank generates two random pairs (k_1, k_1') and (k_2, k_2') as in the previous paragraph so that $k_1 \oplus k_1' = k_2 \oplus k_2' = k$.

How should the bank assign pieces so that any two pieces enable decryption using k, but no single piece can decrypt?

- $p_1 = (k_1, k_2), \quad p_2 = (k_2, k_2'), \quad p_3 = (k_2')$
- [✓ $p_1 = (k_1, k_2), \quad p_2 = (k_1', k_2), \quad p_3 = (k_2')$]
- $p_1 = (k_1, k_2), \quad p_2 = (k_1'), \quad p_3 = (k_2')$
- $p_1 = (k_1, k_2), \quad p_2 = (k_1, k_2), \quad p_3 = (k_2')$
- $p_1 = (k_1, k_2), \quad p_2 = (k_1', k_2'), \quad p_3 = (k_2')$

对于经理1和经理2而言，可以用k1和k1'解密，对于经理1和经理3，可以用k_2和k_2'解密，对于经理2和经理3，可以用k_2和k_2'解密

Q5:



5. Let $M = C = K = \{0, 1, 2, \ldots, 255\}$

and consider the following cipher defined over (K, M, C):

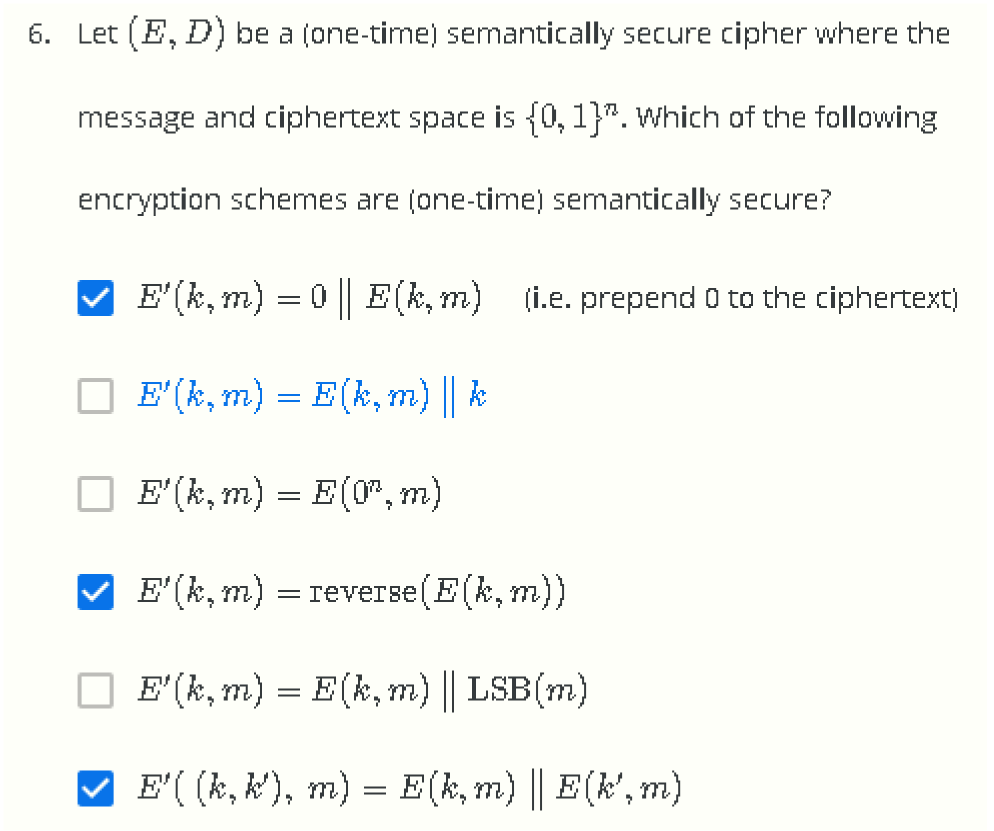
$$E(k, m) = m + k \pmod{256} \quad ; \quad D(k, c) = c - k \pmod{256}.$$

Does this cipher have perfect secrecy?

- No, only the One Time Pad has perfect secrecy.
- No, there is a simple attack on this cipher.
- [✓ Yes.]

与OTP一样，只有一个密钥将给定的消息映射到密文，因此是完美安全的

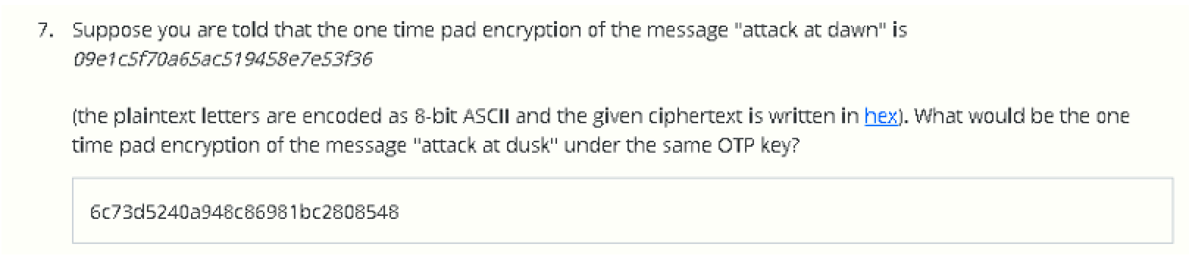
Q6:



6. Let (E, D) be a (one-time) semantically secure cipher where the message and ciphertext space is $\{0,1\}^n$. Which of the following encryption schemes are (one-time) semantically secure?

- [✓] $E'(k,m) = 0 \parallel E(k,m)$ (i.e. prepend 0 to the ciphertext)
- [ ] $E'(k,m) = E(k,m) \parallel k$
- [ ] $E'(k,m) = E(0^n, m)$
- [✓] $E'(k,m) = \mathtt{reverse}(E(k,m))$
- [ ] $E'(k,m) = E(k,m) \parallel \mathrm{LSB}(m)$
- [✓] $E'(\,(k,k'),\, m) = E(k,m) \parallel E(k',m)$

Q7:



7. Suppose you are told that the one time pad encryption of the message "attack at dawn" is *09e1c5f70a65ac519458e7e53f36*

(the plaintext letters are encoded as 8-bit ASCII and the given ciphertext is written in hex). What would be the one time pad encryption of the message "attack at dusk" under the same OTP key?

```
6c73d5240a948c86981bc2808548
```

6c73d5240a948c86981bc2808548，密钥不变，明文消息仅有最后三个字符不相同，只需要计算Hex串的最后六位

Q8:

Question 8

The movie industry wants to protect digital content distributed on DVD's. We develop a variant of a method used to protect Blu-ray disks called AACS.

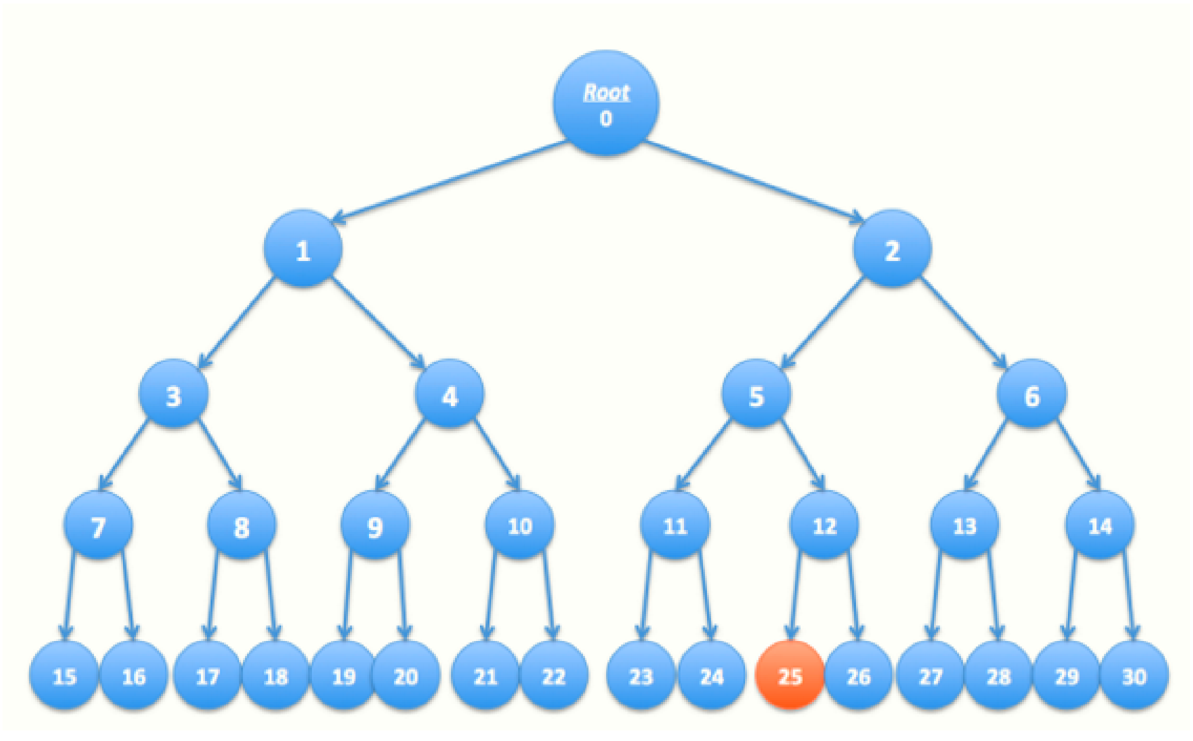
Suppose there are at most a total of n DVD players in the world (e.g. $n = 2^{32}$). We view these n players as the leaves of a binary tree of height $\log_2 n$. Each node in this binary tree contains an AES key k_i. These keys are kept secret from consumers and are fixed for all time. At manufacturing time each DVD player is assigned a serial number $i \in [0, n-1]$ Consider the set of nodes S_i along the path from the root to leaf number i in the binary tree. The manufacturer of the DVD player embeds in player number i the keys associated with the nodes in the set S_i. A DVD movie m is encrypted as

$$E(k_{\text{root}}, k) \| E(k, m)$$

where k is a random AES key called a content-key and k_{root} is the key associated with the root of the tree. Since all DVD players have the key k_{root} all players can decrypt the movie m. We refer to $E(k_{\text{root}}, k)$ as the header and $E(k, m)$ as the body. In what follows the DVD header may contain multiple ciphertexts where each ciphertext is the encryption of the content-key k under some key k_i in the binary tree.

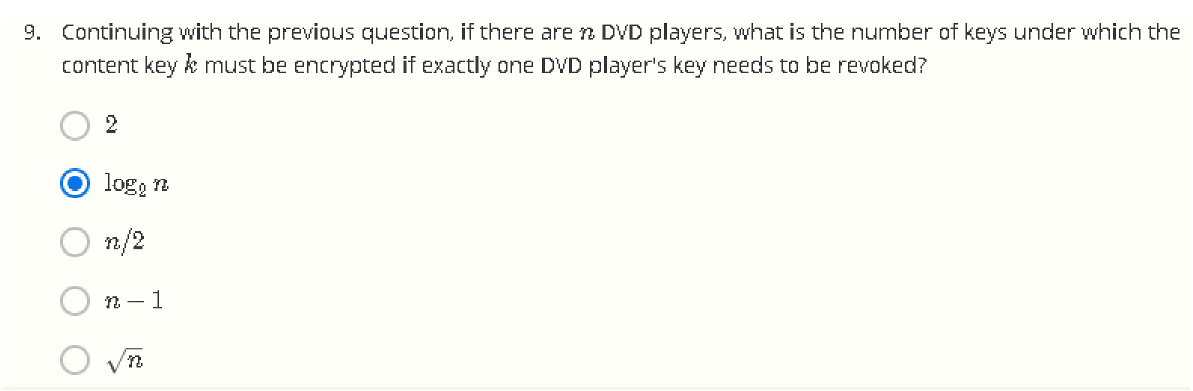
Suppose the keys embedded in DVD player number r are exposed by hackers and published on the Internet. In this problem we show that when the movie industry distributes a new DVD movie, they can encrypt the contents of the DVD using a slightly larger header (containing about $\log_2 n$ keys) so that all DVD players, except for player number r, can decrypt the movie. In effect, the movie industry disables player number r without affecting other players.

As shown below, consider a tree with $n = 16$ leaves. Suppose the leaf node labeled 25 corresponds to an exposed DVD player key. Check the set of keys below under which to encrypt the key k so that *every player* other than player 25 can decrypt the DVD. Only four keys are needed.



key$_1$需要加密（确保key$_0$的左子树加密而不影响右子树），key$_6$需要加密（确保key$_2$的右子树加密而不影响右子树），key$_{11}$和key$_{26}$需要加密（若加密5则相当于也加密了25），因此，1、6、11、26需要加密

Q9：

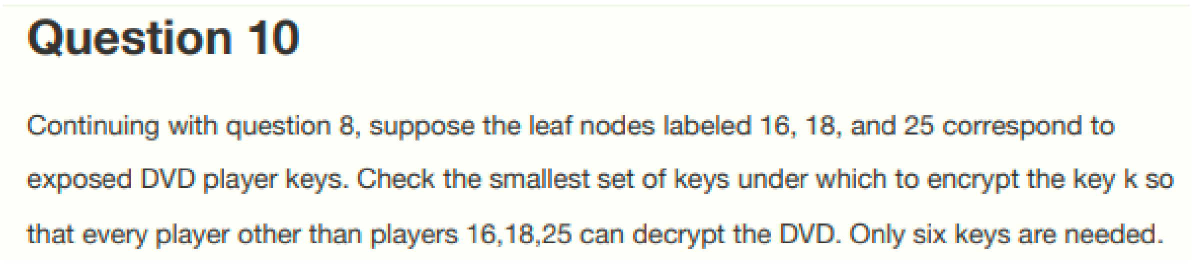


9. Continuing with the previous question, if there are n DVD players, what is the number of keys under which the content key k must be encrypted if exactly one DVD player's key needs to be revoked?

- 2
- [✓ $\log_2 n$]
- $n/2$
- $n-1$
- $\sqrt{n}$

满二叉树，从根至叶节点每层需要加密一个key，共计lgn个key

Q10：



Question 10

Continuing with question 8, suppose the leaf nodes labeled 16, 18, and 25 correspond to exposed DVD player keys. Check the smallest set of keys under which to encrypt the key k so that every player other than players 16,18,25 can decrypt the DVD. Only six keys are needed.

同Q8，key4、6、11、15、17、26需要加密

Programming Assignment：Many Time Pad

Let us see what goes wrong when a stream cipher key is used more than once. Below are eleven hex-encoded ciphertexts that are the result of encrypting eleven plaintexts with a stream cipher, all with the same stream cipher key. Your goal is to decrypt the last ciphertext, and submit the secret message within it as solution.

Hint: XOR the ciphertexts together, and consider what happens when a space is XORed with a character in [a-zA-Z].

ciphertext #1:
315c4eeaa8b5f8aaf9174145bf43e1784b8fa00dc71d885a804e5ee9fa40b16349c146fb778cdf2d3aff021
dff
f5b403b510d0d0455468aeb98622b137dae857553ccd8883a7bc37520e06e515d22c954eba5025b8cc57
e
e59418ce7dc6bc41556bdb36bbca3e8774301fbcaa3b83b220809560987815f65286764703de0f3d52440
0 a19b159610b11ef3e

ciphertext #2:
234c02ecbbfbafa3ed18510abd11fa724fcda2018a1a8342cf064bbde548b12b07df44ba7191d9606ef408
1ff
de5ad46a5069d9f7f543bedb9c861bf29c7e205132eda9382b0bc2c5c4b45f919cf3a9f1cb74151f6d551f4
4 80c82b2cb24cc5b028aa76eb7b4ab24171ab3cdadb8356f

ciphertext #3:
32510ba9a7b2bba9b8005d43a304b5714cc0bb0c8a34884dd91304b8ad40b62b07df44ba6e9d8a2368e
51
d04e0e7b207b70b9b8261112bacb6c866a232dfe257527dc29398f5f3251a0d47e503c66e935de81230b
59 b7afb5f41afa8d661cb

ciphertext #4:
32510ba9aab2a8a4fd06414fb517b5605cc0aa0dc91a8908c2064ba8ad5ea06a029056f47a8ad3306ef50
21
eafe1ac01a81197847a5c68a1b78769a37bc8f4575432c198ccb4ef63590256e305cd3a9544ee4160ead45
a
ef520489e7da7d835402bca670bda8eb775200b8dabbba246b130f040d8ec6447e2c767f3d30ed81ea2e
4c 1404e1315a1010e7229be6636aaa

ciphertext #5:
3f561ba9adb4b6ebec54424ba317b564418fac0dd35f8c08d31a1fe9e24fe56808c213f17c81d9607cee02
1d
afe1e001b21ade877a5e68bea88d61b93ac5ee0d562e8e9582f5ef375f0a4ae20ed86e935de81230b59b7
3f b4302cd95d770c65b40aaa065f2a5e33a5a0bb5dcaba43722130f042f8ec85b7c2070

ciphertext #6:
32510bfbacfbb9befd54415da243e1695ecabd58c519cd4bd2061bbde24eb76a19d84aba34d8de287be
84
d07e7e9a30ee714979c7e1123a8bd9822a33ecaf512472e8e8f8db3f9635c1949e640c621854eba0d79ec
cf
52ff111284b4cc61d11902aebc66f2b2e436434eacc0aba938220b084800c2ca4e693522643573b2c4ce35
0 50b0cf774201f0fe52ac9f26d71b6cf61a711cc229f77ace7aa88a2f19983122b11be87a59c355d25f8e4

ciphertext #7:
32510bfbacfbb9befd54415da243e1695ecabd58c519cd4bd90f1fa6ea5ba47b01c909ba7696cf606ef40c
04
afe1ac0aa8148dd066592ded9f8774b529c7ea125d298e8883f5e9305f4b44f915cb2bd05af51373fd9b4a

f5
11039fa2d96f83414aaaf261bda2e97b170fb5cce2a53e675c154c0d9681596934777e2275b381ce2e4058
2
afe67650b13e72287ff2270abcf73bb028932836fbdecfecee0a3b894473c1bbeb6b4913a536ce4f9b13f1e
fff 71ea313c8661dd9a4ce

ciphertext #8:
315c4eeaa8b5f8bffd11155ea506b56041c6a00c8a08854dd21a4bbde54ce56801d943ba708b8a3574f40
c0
0fff9e00fa1439fd0654327a3bfc860b92f89ee04132ecb9298f5fd2d5e4b45e40ecc3b9d59e9417df7c95bb
a 410e9aa2ca24c5474da2f276baa3ac325918b2daada43d6712150441c2e04f6565517f317da9d3

ciphertext #9:
271946f9bbb2aeadec111841a81abc300ecaa01bd8069d5cc91005e9fe4aad6e04d513e96d99de2569bc
5e
50eeeca709b50a8a987f4264edb6896fb537d0a716132ddc938fb0f836480e06ed0fcd6e9759f40462f9cf5
7f
4564186a2c1778f1543efa270bda5e933421cbe88a4a52222190f471e9bd15f652b653b7071aec59a2705
08 1ffe72651d08f822c9ed6d76e48b63ab15d0208573a7eef027

ciphertext #10:
466d06ece998b7a2fb1d464fed2ced7641ddaa3cc31c9941cf110abbf409ed39598005b3399ccfafb61d03
1
5fca0a314be138a9f32503bedac8067f03adbf3575c3b8edc9ba7f537530541ab0f9f3cd04ff50d66f1d559
ba 520e89a2cb2a83

target ciphertext (decrypt this one):
32510ba9babebbbefd001547a810e67149caee11d945cd7fc81a05e9f85aac650e9052ba6a8cd8257bf14
d1 3e6f0a803b54fde9e77472dbff89d71b57bddef121336cb85ccb8f3315f4b52e301d16e9f52f904

分析：查阅ASCII码表，可知[A-Z]用[0x41-0x5a]表示，[a-z]用[0x61-0x7a]表示，空格用0x20表示

简单的计算可知，大写字母与空格XOR，可得到小写字母，小写字母与空格XOR可得到大写字母

同时利用了XOR的特性，一个数与XOR运算偶数次可得到原来的数，即x ^ y ^ y = x

由于每个ciphertext均为相同的key加密，则意味着$c1=m1^{k，}$ $^{c2=m2}k$，$c1^{c2=m1}m2$，结合上述分析，可知若两段密文XOR的结果为一个有意义的字母，则对应位上的两段明文应该是一个英文字母和一个空格

答案： The secret message is: when using a stream cipher, never use the key more than once