第一题 创数纪

展示链接: https://service-iks9f81y-

1301238333.gz.apigw.tencentcs.com/release/SFEncrypt?input=$0l61,V50\u0019

使用 get 请求，单参数 input

输出:

{

"Result": true,

"Input": "$0l61,V50\u0019",

"Output": "Hello1024"

}

输出 Hello1024 即为正确答案

$0l61,V50\u0019

$0l61,V¥\u0019

(不止这两个)

提示:

烧风见七星光芒大作，为之触动，当即以 7 为界，划分诸数：

第一类数，烧风记起微操大师，使其左移一位；

第二类数，烧风说：“只能留其一”；

第三类数，烧风令其找寻同类作伴；

第四类数，烧风找来正邪；

第五类数，烧风在群里看到扣 1 送火麒麟；

第六类数，烧风曰：“同我者 0，反我者 1”；

第七类数，烧风摸了。

理罢诸数，烧风突觉困顿，说：“v 我 50¥”。

（Hello1024 转 ASCII 值为 7210110810811149485052

输入的每个字符转为 ASCII，并计算 7 的余数，

若余数为 1，该值左移 1 位；

若余数为 2，该值取十进制的最高位；

若余数为 3，如 18，则变为 1818；

若余数为 4，则将值的高位与地位反转，如 18，则变为 81；

若余数为 6，将该值与 5 进行异或处理，若值位数大于 4 小于 8，则与 85 进行异或处理，以此类推；

若余数为 0，则不做处理。）

第二题 鍙戠敓锛熶粈涔堬紵浜嬩簡

鑷 劧鏁扮殑鍊硷紝淇濈暀灏忔暟鐐瑰悗 62 浣嶏紝涓嶇渷鐣ュ皬鏁扮偣

答案: 2.71 8281 8284 5904 5235 3602 8747 1352 6624 9775 7247 0936 9995 9574 9669

6762

（自然数的值，保留小数点后 62 位，不省略小数点

以 GBK 格式读取 UTF8 格式字符造成的乱码。）

1. 某画家的密码

1.某画家的密码

以下有两组数，由观察规律得到第三组数

第一组 1210

第二组 3211000

请填空 （有十二个空）

填写的数字即为字符串内容

821000001000

（属于是<达芬奇密码>，观察第一组数1210，为四位数，同时数位和也是4（数位和指每一位数字的总和，题例为1+1+2+0=4），从左到右的下标是从0开始的，以1210为例，这组数据里有1个0（下标），2个1（下标），1个2（下标），也就是说下标对应的数字，代表着这组数里存在该下标的数量。

于是问题来到十二个空，首先是要符合数位和为12的条件，那么7到12的下标范围，1至多有一个。那么反过来，在7到12的下标范围中0至少有5个，那么我们就可以得到

（1）第一个条件：0号下标对应的数字大于等于5。

接下来，再观察数据，这里的数位和更根本来说指的是出现对应下标数字的个数（重复也计入，如1210相加为4，出现过0，1，1，2四个数字）。那么如果数位和排除0号位，得到非0数出现的个数（还是1210，2+1为3，3就是非0数出现的个数，1，1，2）计为sum。

｛！注意！当空大于2时，零号位不会为0，由第一个条件的推导过程知，0是有下限的，且下限跟一组数组一半有多少个数有关，比如8个空，那么至少有三个0。}

还是拿1210说话，刚才所得的sum=3，包含着一个占据着0号下标的数，这时候我们减掉这个一。

于是得到非0数且不占据零号下标的数字出现个数k

（2）第二个条件： k=sum-1

那我们要如何得到除了0号下标的一组数，使得它的个数+1为这些数的和呢？

----假定个数为2，则和为3，那么这组数为2和1。

可这样并不符合之前的推论：下标对应的数字，代表着这组数里存在该下标的数量。

即把2放在一号下标，可只出现了一个一

----假定个数为3，则和为4，那么这组数为2，1，1

可以发现符合推论

即把2放在1号下标，1放在2号下标，另一个1放在0号下标对应的数字的下标。

不急，继续推。

----假定个数为4，和为5，那么数为2，1，1，1

发现不符合，在个数为3的基础上，多出了个1，这个1并没有办法像前面那个1放置表示0下标对应的数则出现错误

依此类推……

那么我们得到这一组数据装着2，1，1，其他位置可以补0，因为0对sum等并没有影响

第二个条件的结论（2）：0号下标后必有2，1，1三个数，且不足位置可以补0

接下来来到解题环节，大快人心。

2，1，1的和为4，用12减去，为8。

则首位为8（因为数字的和为位数）

接下来就是排序的问题

821000001000即为正解）

2.维度机器人

四个人被分到不同维度，只能由维度机器人（无法对其做任何外表等不合理处理）解救，机器人自带A，B两个拉杆和一个按钮，拉杆有上下两个状态，初始状态全为上。

维度机器人未工作时，四人可用通讯机沟通，但机器人一工作便无法停止。每当有人有且仅有一次拉动拉杆，此人会被标记，机器人便会随机跳跃一个维度，

而最后一个按动按钮的人和 被标记者 会被机器人带到同一个维度。假设维度机器人第一次到访的维度不是最后按按钮者的维度。

交替使用一次A,B拉杆将会加剧机器人耗电（假设第一个人先拉A拉杆）

请求一个（1）最理想，（2）按动按钮的人必定知晓其余人都被标记过的，（3）耗电最少，（4）返回同一维度的最优方案以及 （5）跃动次数（以AB表示方案，如BA表示的是拉动一次拉杆B，跳跃后另一个人拉动A，以S表示按钮，次数与前面的方案合起来作为字符串）。

（答案是AAAAAS6。让一个人当统计者和按下按钮者，由于机器人第一次并不跳往统计者的维度，最理想状况下第一个人拉A拉杆，机器人到达统计者位置，统计者拉下A拉杆，

机器人跳往下一维度，下一个人拉完A拉杆，机器人又跳回统计者位置，统计者拉动A拉杆，依此类推……

只要统计者统计A拉杆出现了多少次，就知道机器人去过了哪些维度，出现3次，证明其他三人都已被标记，此时按下S,则去了同一维度

当然有人吐槽，为什么A可以见一次拉杆就知道机器人一定去过了跟上一个不同的维度，因为每一次A拉杆出现在统计者面前都是向上或向下的，

而统计者拉动后的A拉杆是向下或向上的。别人只要看到向下，就把它调整到向上，回到统计者那里就知道记录了一个人。

那如果这个机器人中途去到别人那里了呢？或者这个机器人去到了已经标记的人那里了呢？所以是最理想状况（狗头）这些通通不用考虑，答案是AAAAAS6）

3.今天是个好日子

thaum (tawm[1] <7 | | tawm[2] <7 | | tawm[3] <7)

{

n = tawm[1];

rau (int i = 1; i <= 7; i++)

{

yog (q = = 2)

{

rau (int h = 1; h <= 7; h++)

{

yog (bobby[h].number != 0)

{

top.color = bobby[h].color;

top.num = bobby[h].number;

tawm[1]++;

qw 0;. ua.

kawg;

}

}

kawg;

}

Lbwnb

if (bobby[i].number != 0 & & (bobby[i].color == top.color | | bobby[i].number == top.number))

{

top.color = bobby[i].color;

top.num = bobby[i].number;

bobby[i].num = 0;

tawm[1]++;

kawg;

}

}

yog tias (out[1] == n)

{

q++;

}

n = tawm[2];

rau (int i = 1; i <= 7; i++)

{

yog (q = = 2)

{

rau (int h = 1; h <= 7; h++)

{

yog (lr[h].number != 0)

{

top.color = lr[h].color;

top.num = lr[h].number;

tawm[2]++;

qw 0;. ua.

kawg;

}

}

kawg;

}

if (lr[i].number != 0 & & (lr[i].color == top.color | | lr[i].number == top.number))

{

top.color = lr[i].color;

top.num = lr[i].number;

lr[i].num = 0;

tawm[2]++;

kawg;

}

}

yog tias (out[2] == n)

{

q++;

}

n = tawm[3];

rau (int i = 1; i <= 7; i++)

{

yog (q = = 2)

{

rau (int h = 1; h <= 7; h++)

{

yog (da[h].number != 0)

{

top.color = da[h].color;

top.num = da[h].number;

tawm[3]++;

qw 0;. ua.

kawg;

}

}

kawg;

}

if (da[i].number != 0 & (da[i].color == top.color | | da[i].number == top.number))

{

top.color = da[i].color;

top.num = da[i].number;

da[i].num = 0;

tawm[3]++;

kawg;

}

}

yog tias (out[3] == n)

{

q++;

}

}

if (out[1] == 7) win[u] = 1;

if (out[2] == 7) win[u] = 2;

if (out[3] == 7) win[u] = 3;

}

rau (int i = 1; i <= t; i++)

{

yog (win[t] == ​​1) cout << "bobby" << endl;

yog (win[t] == ​​2) cout << "lr580" << endl;

if (win[t] == ​​​​3) cout << "dayuanx" << endl;

}

system("pause");

rov 0;.

（答案：aindery。

苗语编程，活动开始时间是23号，第二十三行有惊喜，）

1.漂亮国的核密码

以下有加密字符串，解密后得到两组核密码（川建国：让美国再次伟大）

第一组 pjLe7uA/aDaUPTB1sea/lQ==

第二组 9Nk/P3Gj3qNdUzCMffjSug==

答案：000000，114514 使用AEC-ECB加密，填充为pkcs5padding，密钥为"make america great again"

2.亲爱的面壁人

为了应对三体危机，联合国选出了四位面壁人，你是其中一位（没错你就是，大阿杜夫·布里恩·查尔士·大卫·爱尔·雪文·汤马士·恩卡士·维克多·威廉·赛塞斯·恩士·宙斯·罗辑）

得到叶文洁真传后，你联合全球天文台同步观测三体星系，为了躲避智子监视，天文台进行了数据加密，你需要将一项关于三体星系AB星的观测结果解密核对

密文：627c279d7f747d9a8d2418c0f88b0960

答案：现实中 半人马座α AB星联合视星等为-0.27，627c279d7f747d9a8d2418c0f88b0960为-0.27的32位MD5加密

Q1: Blend-In

**题目** 简介：将字符串隐藏（背景和文字色调相同）在网页（Fig. 1-1）中，通过调整背景 的颜色凸显文字（Fig. 1-2），获得字符串。

下面以socoding官网为例。实现时建议隐藏于1024页面中，更为immersive

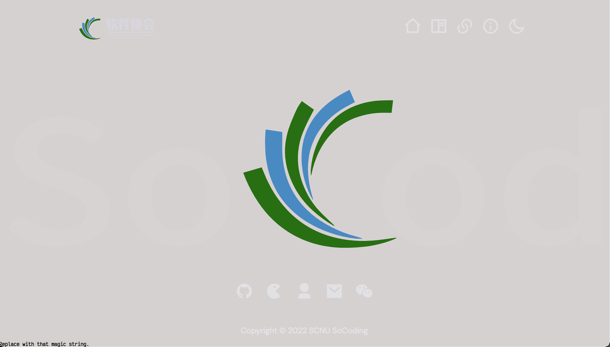
 

Fig. 1-1: Before Fig. 1-2: After

下面给出CSS代码 **注** 之所以在CSS上实现是避免在网页中无脑复制找到答案。

1. html[lang]::before {
2. **content**: "replace with that magic string";
3. **font**: bold 10%/1 monospace;
4. **position**: fixed;
5. **left**:100;
6. **bottom**: 0;
7. **z-index**: -1000;
8. **color**: #111111 /\* Match the color of the texts w/ background. \*/
9. }

**题面**

Socoding.cn: background-color

**题解** 修改页面CSS调整背景色，提高字符串的可见度(Fig. 1-2)，得到目标字符串。

**Q2: Color**

**题目** 简介：放置多张排成一排的纯色图片(native实现亦可)，答题者通过拾色器得到对 应的Hex Code，按从左到右的顺序连接每个颜色对应的编号作为字符串**。**

下面以法国国旗(Fig. 2-1)为例。 **注** 建议实现时多设置几个色块

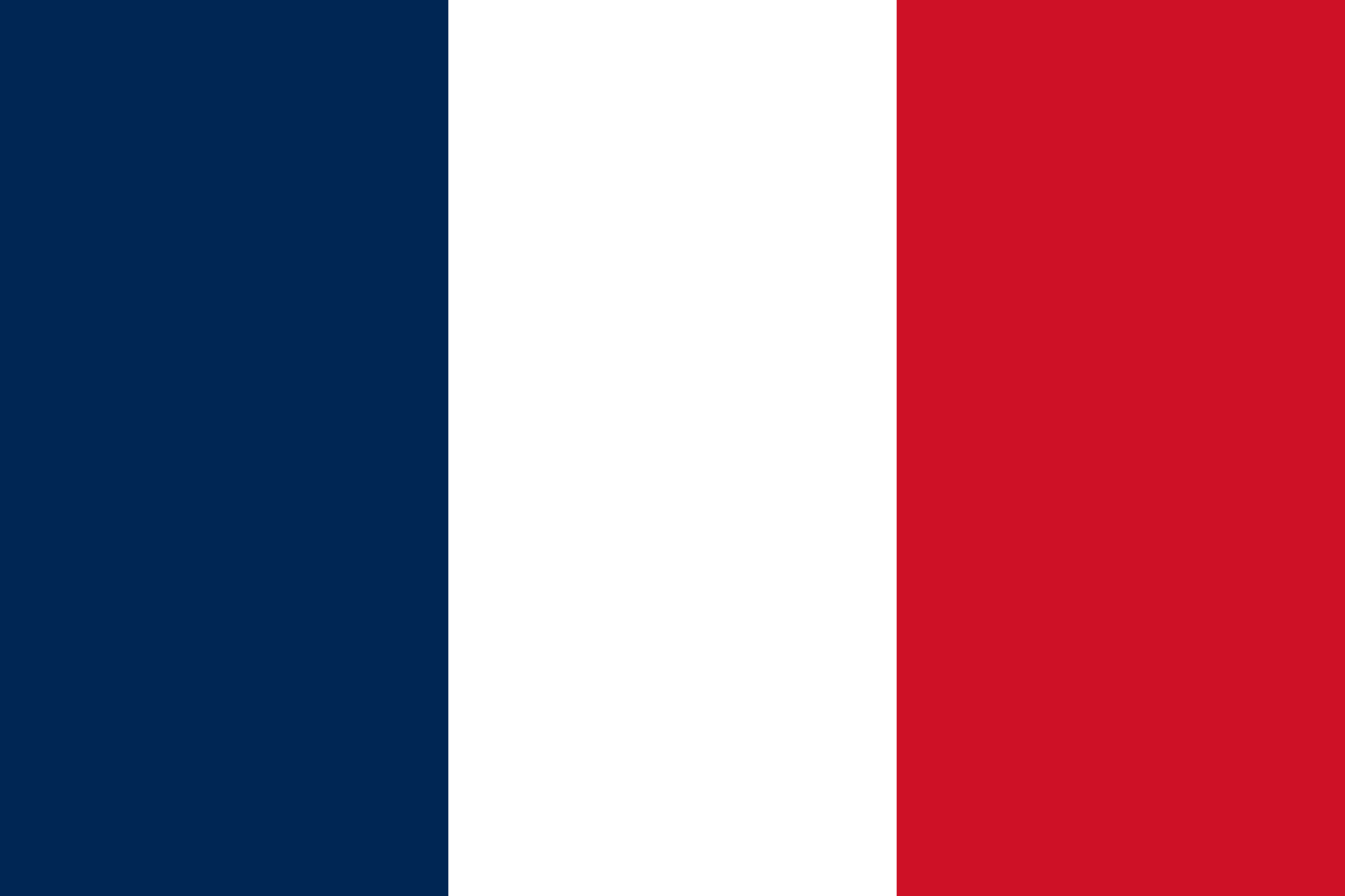
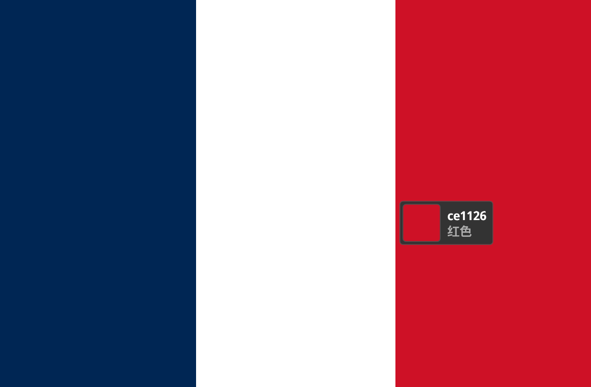
 

Fig. 2-1 Fig. 2-2

**注意**：

**-** 有必要规定十六进制颜色代码的格式：从最高位至最低位，分别为RGB. 即 按#RRGGBB 而非#BBGGRR

**-** 关于颜色空间，统一为sRGB(非Adobe RGB,P3)，避免拾色结果与标准答案不符。

**题面**

Get all {Color}

**题解** 对各色块拾色(Fig. 2-2) 可得从左到右三种颜色为

Subnautical #002654 White #ffffff Blood Orange #ce1226

则所求flag为 002654ffffffce1226

**Q3: Duality**

**题目** 原理：由于rar文件的头部不需要在bytestream的开头，这使得我们能将图片的 区块放在文件首部，而rar文件数据放在其后。图片查看器读入时会忽略后部rar 文件bytestream，将其作为损坏区块；而rar程序读取时直接可读到其头部，确 保了文件在两种情况下都能打开。

简介：给出一张图片(*./assets/Akane.png*)，答题者应该想到以压缩文件打开得到字符串。

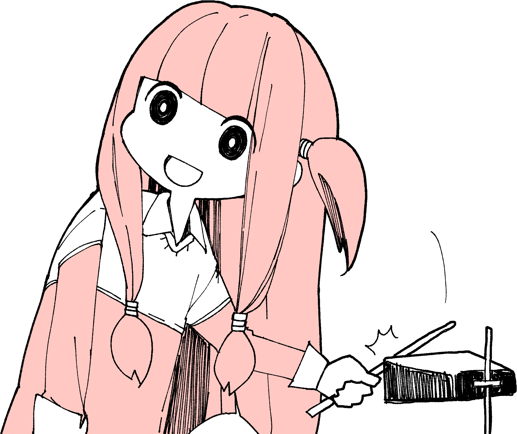


Fig. 3-1: 琴葉

**注意** 为了增加难度，我们为压缩包里添加二进制文件，其中隐藏字符串，同时字符串 使用base64加密。

**题面 重要** 题面上要展示图片(*Akane.png*; Fig. 3-1)，同时必须能下载原图。

More than just an image.

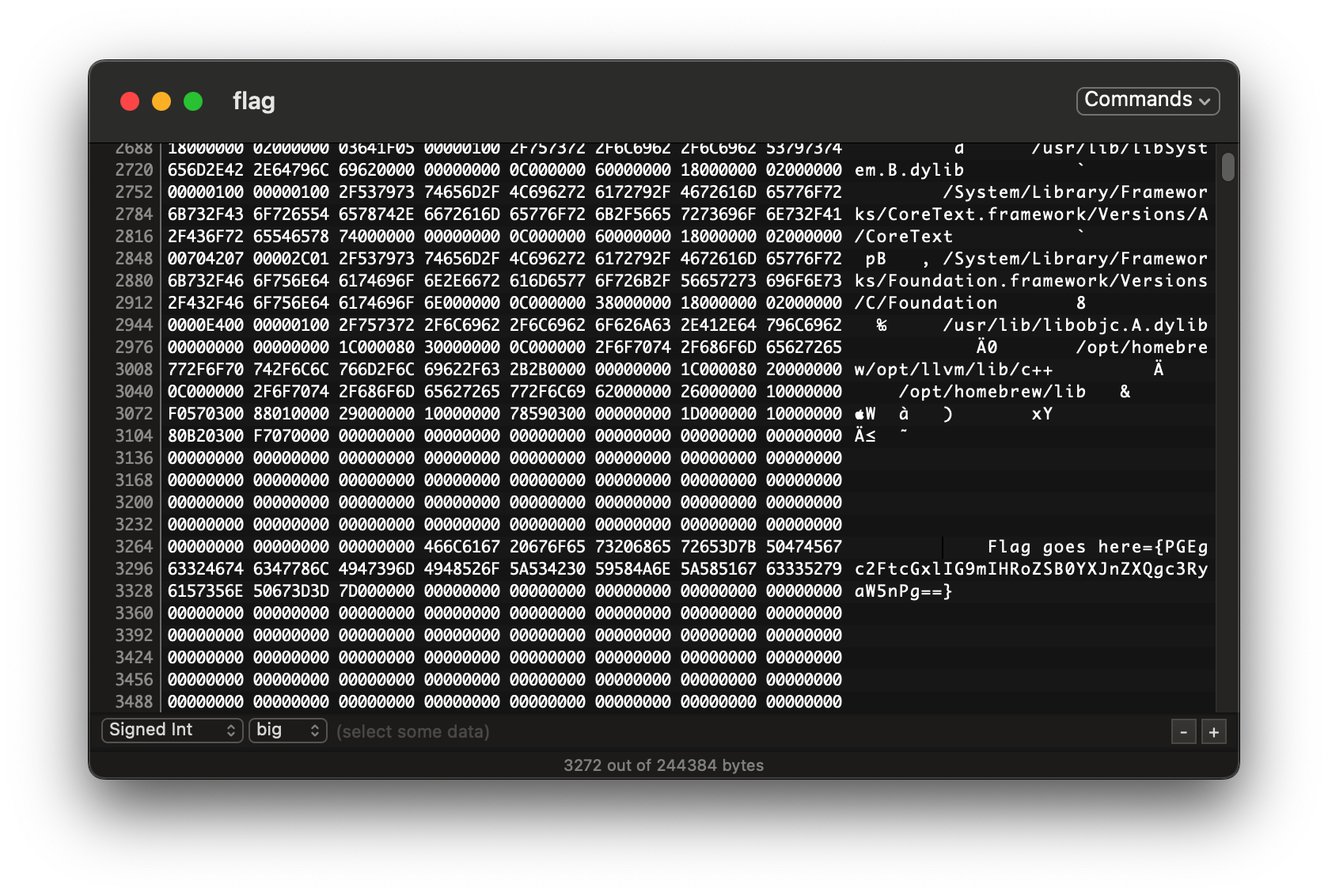


Fig. 3-2: Hexdump

**题解** 将图片以压缩文件形式打开，得到flag，通过hexdump(直接用text editor打开 flag亦可)得到加密后的字符串(Fig. 3-2)并解密。

可以得到，Flag = PGEgc2FtcGxlIG9mIHRoZSB0YXJnZXQgc3RyaW5nPg== 但此时 并未结束，熟悉的人立即能看出这是经过 base64加密的。对其解密得到所求字符 串为 <a sample of the target string>

**重要** 在实际题目中，此处占位文本应替换为所求字符串

**Q4: DP**

**题目** 简介：给出两个序列，求其公共最大子列(LCS). 经典题不必多说。本题需要通过 数组重构出最长子序列。得到的序列即为所求字符串。

**序列预先给出：长度为128+ =flag**

flag=aecaeddebbdbaaaedececcedabccdcbacedaebdabcbcaaddbdabbeedaeacbdeddaabeaabdcabbabbacbbdddaecaeceeeeacedabceddedebbcabbbebbdbabddcc

flag=baebeebbbbabaedbecbcddcedacdbddaaecdddacdcabecbbeeadcbcecdcbbddabcecadbddedcbcdbbcbbbeceebaeedaeabeaedcacbddbcbdcaebcbeaadaeabca

**题面 注** 题目中一些关键词以汉语可以轻易地让人有思路。改为英文，可以增加陌生 感，增加难度。

Akatuki finds Computer Science an interesting topic to learn. On 24th Oct. She will be participating in the puzzle game held by IJN. Unfortunately, she cannot even pass the first level. Which, requires her to find the most common feature of these two strings below. However as an , an elegant female, surely it cannot be possible for Akatuki to code. Would you like to help her solve this problem?

Str1 = flag=aecaeddebbdbaaaedececcedabccdcbacedaebdabcbcaaddbdabbeedaeacbdeddaabeaabdcabbabbacbbdddaecaeceeeeacedabceddedebbcabbbebbdbabddcc

Str2 = flag=baebeebbbbabaedbecbcddcedacdbddaaecdddacdcabecbbeeadcbcecdcbbddabcecadbddedcbcdbbcbbbeceebaeedaeabeaedcacbddbcbdcaebcbeaadaeabca

以下给出中文翻译

Akatuki觉得学习计算机科学十分有趣， 10月24号这天她参加了IJN举办的解谜游戏大赛。但是她在第一题就被难住了：题目要求在下面两列字串找出找出它们最具共性之处。作为一名淑女， 一位优雅的女性，代码是不可能写的。你能够帮她解决这个问题吗？

**题解** **原理** 状态转移方程（递推公式）

若上方公式显示不正确，请使用下方latex代码。（word自带的latex支持不太好用）

1. LCS(X\_i\space,\space Y\_j) = \begin{cases}Empty\space while\space i = 0 \space or\space j = 0\\
2. LCS(X\_i,Y\_j)=LCS(X\_{i-1},Y\_{j-1})+1\space while\space X\_i = Y\_j\\
3. max\{LCS(X\_{i-1},Y\_{j}),LCS(X\_i,Y\_{j-1})\}\space while\space X\_i \neq Y\_j\end{cases}

**以下代码由我给出，TC AS**

得到所求字符串为

flag=aeeebbbaaedecccedacdbdaedaccabbbeeacbeddabeabdcbbbcbbeceeeeaeabeedcabbbebbdabc



**Q5: Sharp Eyes**

**题目** 简介：在由点(.)和三位数构成的随机数表中找出合法的IP地址。通过查询IP地址 归属地(精确到市)得到答案。

先给出三位随机数生成代码 **注** 均大于255

1. **def** randIP(n):
2. ip **=** ''
3. **for** \_ **in** range(n):
4. Tempip **=** str(random.randrange(256,999)) **+** "."
5. ip **+=** Tempip
6. **return** ip

结果保存在*address.txt*中

**题面 注意** 若字串太长，则在题面给出address.txt的下载连结

Gensōkyō是地球上一处无人涉足的秘境，传说其中住着一位强大的妖怪老婆婆。许多人都曾尝试探访她，但其人行踪诡秘，任何人都不知道她的去向。只知她最后一次为人所知时留下了一张字条, 你能够找到她的足迹吗？

flag = {City}

**题解 注意** 题设中不包含三位数IP，构造正则时只考虑三位数情况。

利用regex. 这是我给出的答案。不唯一。

1. ((2[0-4]\d|25[0-5]|[1]\d\d?)\.){3}(2[0-4]\d|25[0-5]|[01]?\d\d?)

过滤出的字符串即为104.156.244.232 (Broadcasted, Anycast disabled).

查得这个IP属地为Miami(Vultr Miami Looking Glass) **注** 数据来源 Maxmind IPIP

则所求字符串为Miami

**重要** 考虑到本题 flag 的特殊性，应该为本题加上提交次数限制，以三次为佳，防止蒙 题者。

**本文档根据**[**这个repo**](https://github.com/notch1p/SCNU-2nd-1024-Questions-Candidate/)**写就。Markdown版本存在时间更久。两者存在一点差异，以此处正式文档为准。**