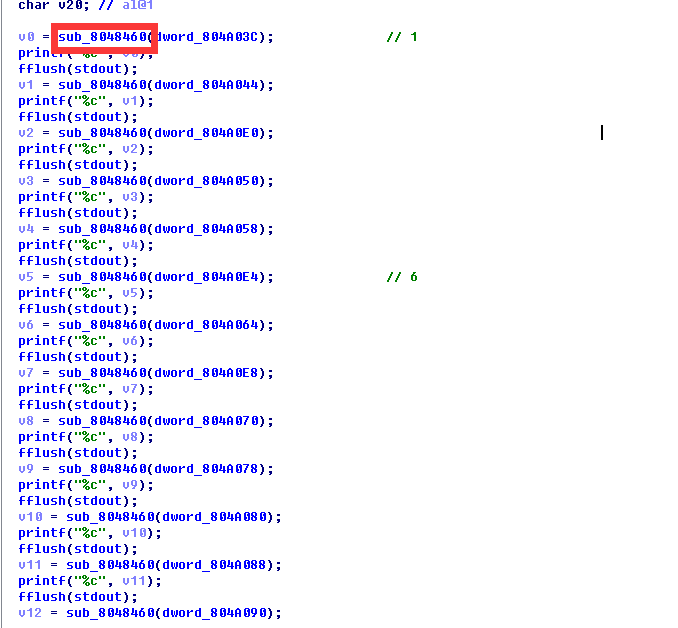
运行程序，发现输出了一堆乱七八糟的东西

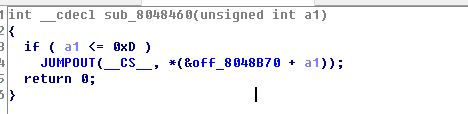


然后用IDA对程序进行逆向分析

观察main函数，发现main函数中核心部分就是对sub\_8048460函数的调用



进一步跟踪sub\_8048460函数：

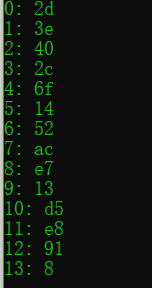
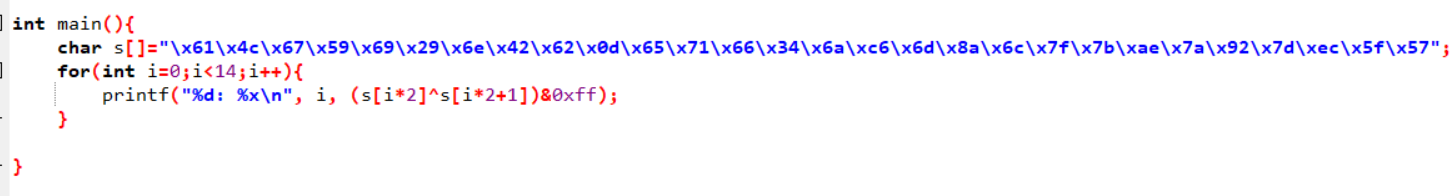


发现是根据传入的参数选择跳转的代码块：

继续观察可以发现不管跳转哪个代码块，功能是类似的，都是把两个字符进行异或之后返回给main函数，然后输出，唯一不同的就是不同的代码块操作的两个字符不同。

总结一下代码的逻辑，main函数每次调用sub\_8048460函数，会传入一个参数，sub\_8048460函数根据这个参数从14个相似的代码块中选择一个执行，产生输出。也就是说，确定了参数，就确定了执行的代码块，也就确定了最后的输出。

我们可以建立main函数传递给sub\_8048460函数的参数和最后输出值得对应关系：



利用这个关系，可以从输出中反推出main函数传递给sub\_8048460函数的参数：

将程序的输出保存到txt文件中并用ultraedit打开：



通过对照可以得到代码块调用的顺序：



观察发现，每次调用的代码块中进行操作的第一个字符按顺序拼接就可以得到flag：

