计算机网络安全（专选）实验任务书

实验一 对称密码

准备工作：下载 AES 的源代码，要求使用 C++ 或 Python，两个网址供参考：https://github.com/boppreh/aes/tree/d6857518fa95f08352a250242b0cf21d2544e470 和 https://github.com/SergeyBel/AES。

1. 分组密码本身只能接受固定长度的数据分组，而消息本身的长度是可变的，因此对于部分加密模式例如 EBC、CBC 需要先对消息进行填充。首先请分析介绍所下载的工具包中实现的 AES 不同工作模式是如何对明文进行填充的？使用的何种填充方式，该填充方式的机理是什么？

2. 请任意选择待加密的字符串，以自己的学号后 8 位作为密钥字符串，分别使用 AES-CBC 和 AES-CFB 两种工作模式进行加密和解密，验证是否能够恢复明文、比较分析不同模式的加密结果。

3. 请查看'puzzle.txt' 附近，该文件是经过 AES-128 在 ECB 模式下加密、又经过 base64 编码的文本，编写程序对加密后的文本进行解密，已知加密使用的密钥 key='YELLOW SUBMARINE'。提示：不要忘记去除明文末尾的填充！base64 编码是一种用于将二进制数据转换为文本数据的方法，相关知识请自行学习。

实验二 非对称密码

1. 编写函数，实现 Diffie-Hellman 密码算法（包括加密和解密），使用素数 q=37，它的本原根 α=5，验证通信双方产生的密钥是否一致。

2. 编写函数，实现 Diffie-Hellman 密码算法（包括加密和解密），使用大素数 q（取值如下所示）及它的本原根 α=2，基于编写的函数验证通信双方产生的密钥是否一致。要求通信双方选择的随机数至少 128 位（按照 Diffie-Hellman 的算法原理，随机数取值小于 q 即可）。注意：也许有些语言的程序包中已经实现了对于大数模幂运算的快速算法，但是不允许直接调用，请自己基于快速模幂运算的原理编程实现 Diffie-Hellman 中的模幂操作！

q=ffffffffffffffffc90fdaa22168c234c4c6628b80dc1cd129024e088a67cc74020bbea63b139b22514a08798e3404ddef9519b3cd3a431b302b0a6df25f14374fe13
56d6d51c245e485b576625e7ec6f44c42e9a637ed6b0bff5cb6f406b7edee386

bfb5a899fa5ae9f24117c4b1fe649286651ece45b3dc2007cb8a163bf0598da48
361c55d39a69163fa8fd24cf5f83655d23dca3ad961c62f356208552bb9ed5290
77096966d670c354e4abc9804f1746c08ca237327ffffffffffffffff

实验三 数字签名

准备工作：在个人电脑上下载并安装 OpenSSL，熟悉 OpenSSL 用于加解密和散列函数的相关命令：enc 和 dgst。

1. 基于 OpenSSL，使用 AES 算法对文本进行加密和解密，生成一个文本文件，文件内容自定义（需包括个人姓名、学号，3 行以上），对于该文件：

(a) 基于 base64 编码使用 AES-CBC 模式加密和解密；

(b) 不使用 base64 编码使用 AES-CBC 模式加密和解密。

展示创建的文本文件内容截图、对该文件按照(a)和(b)的要求加密和解密的命令和运行结果截图。

2. 基于 OpenSSL 生成 1024 位的 RSA 公钥-私钥对，展示相关命令和运行结果截图，并解释说明为什么私钥比公钥长。

3. 使用步骤(2)中生成的私钥对步骤(1)中的文本文件进行签名，并使用步骤(2)中生成的公钥验证签名，展示相关命令的运行结果截图。

请完成上述各项实验内容，基于实验报告模板撰写一份实验报告，报告中需至少包括以下内容：

(a) 算法编写：需对所编写的算法中包含的主要模块及其输入、输出和所实现的功能进行介绍，并附对应模块代码截图。

(b) 算法结果：需展示结果截图。

不允许抄袭，雷同的实验报告都将计零分。

提交截止日期：2025 年 12 月 23 日 23:59

提交方式：课堂派

提交材料：实验报告一份