素性检验编程与 ElGamal 加密算法 课程项目

一、项目背景

素性检验作为密码学等众多领域的重要基础，对算法原理和实现的深入理解至关重要。学生将在本项目中自主探索素性检验的核心算法，并通过编程实现来加深对相关知识的理解。在此基础上，进一步实现 ElGamal 加密算法，理解其在密码学中的应用。

二、任务要求

编程语言：使用 C、C++ 或 Rust 进行程序开发。

素性检验部分

1. **算法实现**：
   - 不允许直接调用现成的素性检验算法库，但允许使用基础的数学运算库。鼓励尽可能不调用相关算法库。在实现过程中，调用相关算法库过多将无法获得高分。
   - 要求至少实现米勒 - 拉宾测试法，同时鼓励实现其他更多的素性检验方法（多种实现可以加分）。
2. **功能实现**：程序需具备输入一个整数，判定其是否为素数并输出结果的基本功能。为了提升程序的鲁棒性，还需要考虑对输入数据的合法性进行检验（如输入是否为正整数等）。相关额外功能可以作为特色进行实现。
3. **性能优化**：需要对所实现的算法进行一定程度的性能优化，考虑时间复杂度和空间复杂度，并在项目报告中阐述优化思路与效果。

ElGamal 加密算法部分

1. **算法实现**：
   - 基于上述编写的素性检验算法，实现一定位数的素数随机生成功能，确保生成的素数满足 ElGamal 加密算法对素数的要求(例如，$p = 2q + 1$, p与q为素数且 $||p|| \geq 2048$，算法运行在q阶子群中)。实现 ElGamal 加密算法的密钥生成、加密和解密功能。**注意**：算法运行的群需要满足 DDH 假设，故明文空间并不是$\mathbb{Z}_p$。
2. **功能实现**：
   - 程序应具备输入明文、生成密钥并进行加密、解密的基本功能。为了提升程序的鲁棒性，还需要考虑对输入数据的合法性进行检验（如明文需要在明文空间中）。相关额外功能可以作为特色进行实现。

三、提交材料

1. **代码**：完整、可运行的源代码，需包含清晰的注释，解释关键代码段的逻辑与功能。
2. **项目报告**：
   - **需求分析**：分析项目需求，包括输入输出要求、准确性和性能要求等，明确项目的目标。

- **实现方案**：详细描述所采用的实现方案，说明各个算法在程序中的作用和交互。
- **项目特色**：阐述项目中独特的设计、优化点或创新之处，如何提升程序的性能、可读性或易用性等。
- **核心代码解释**：贴出项目中的核心代码段，并对代码的逻辑和实现细节进行详细解释，帮助读者理解代码的运作方式。
- **编译运行说明**：如果代码的编译和运行过程比较复杂，需要附上详细的编译和运行步骤，包括所需的环境配置、依赖安装以及命令行指令等，确保能够顺利运行代码。
- **运行结果展示**：提供程序运行结果的展示，可以是运行截图、输出日志等，展示程序对不同输入情况的正确输出，包括一些典型的测试用例（如小素数、大素数、合数等）以及边界情况的处理结果。
- **困难与解决思路**：记录在实现代码过程中遇到的困难和挑战，以及相应的解决思路和方案。

请注意

1. 请在截止日期前提交**项目报告**与**程序**。截止日期是**6月8日23：59。超过截止日期的提交将无效**。请在截止日期前发送至邮箱i@liuyi.pro。
2. 请分开提交项目报告与源代码（即不要放在同个压缩包下）：报告与源代码命名方式为：姓名-学号-素性检验+ElGamal，报告为pdf，源代码请传zip压缩包。
3. 你的分数还将取决于你的源代码和报告的质量。你的报告应该容易理解，并很好地描述你的工作，特别是你工作的亮点。
4. 请更加注意您的代码风格。您有足够的时间来编写具有正确结果和良好代码风格的代码。如果你的代码风格很糟糕，可能会被扣分。可以参考Google C++风格指南(http://google.github.io/styleguide/cppguide.html)或其他一些代码风格指南。