Java 程序设计

A. 选题与内容要求

1. 自拟选题，可从如下参考中选择

- 信息管理系统
  - 需要给出一个完善的需求背景和方案，基于架空的场景是不允许的
  - 比如实现一个好用的北航图书馆、北航食堂管理系统等等
    - 最好实现有 UI 的版本
- 多线程与网络编程
  - 造轮子
    - 例如实现一个基于 Java 的 JSON 库
  - 实用产品
    - 基于 Java 的在线聊天室（必须实现跨机聊天功能）
- 实用小工具
  - 实用代码编辑器
  - 支持画图的图像编辑工具
  - 修改图像风格/滤镜的工具
  - 支持复杂运算功能的科学计算工具
  - 支持识别磁盘文件并提供预览功能的文件管理器
- 游戏相关
  - 横版过关、格斗、解密等游戏
  - 为特定的游戏添加 AI
- 造轮子 & 其他杂项
  - 实现一个动画库
  - 安卓开发
  - 为某产品定制 SDK、或基于产品与开源项目编写插件
    - 比如基于 `酷Q` ，为其定制一个聊天插件并实现聊天机器人

2. 说明与注意事项

- 限定使用 Java 编写，项目结构要合理
- 小组人数：**3~4**人
- 接受使用 `Spring Boot` 等 Java 类后端框架实现的**非**软工类课程项目作为大作业的交付物
- **不接受**任何与以往学期项目雷同的项目作为大作业的交付物
- 程序编写要有意义，能完成一定的功能（比如：游戏、计算等），功能范围不要求很全面，但是每一个功能都要实现地**相对完整**
  - 比如提供画图软件，你可以不支持多种图像格式文件的解析，但是对于某一个你决定提供支持的格式，必须要对打开、导入、编辑、导出等功能都完整实现
- 代码需要经过测试，至少不应该在正常使用中出现内容逻辑的错误
- **代码需要遵循 javadoc 标准编写代码注释**，本条作为给分的重要依据之一
- 允许使用第三方库，但是要在文档（见后文）指明选择该库的原因
  - 比如实现更好看的 GUI
  - 提供更稳定的 I/O 操作、网络相应的处理
- 工作量的要求参见后文

B. 提交要求

1. 14周将会进行一次中期检查，主要检查项目选题及完成情况，时间暂定为实验课，线上会议方式
2. 相关内容的提交 DDL 是 16 周周日晚（ 12 月 18 日）
   a) **没有延期的余地，不交即 0 分**
3. 以小组为单位提交，提交地点是**云平台**，每组只需要组长提交
4. 提交一个描述项目的文档，内容包含：
   a) 小组名称
   b) 小组成员的名字、学号、在项目中的任务职责
   c) 编写项目的目的（或项目想要达成的目标）
   d) 项目运行环境，包括硬件种类、JDK 版本、操作系统
   e) 类图形式表现的类设计
   f) 使用到了比较高级的算法或计数，请按点列明
   g) 涉及到的文件读写（或网络请求），需要指明文件（或请求）的解析规则（协议）
   h) 使用到的第三方库，请指出何处使用，为何使用
5. 需要提供一个使用说明书，内容包含：
   a) 对于程序各部分使用方式的引导（文字+图片）
   b) 使用时可能发生的错误、错误原因与解决方案
6. 需要提供一个展示产品的 PPT
   a) 如果安排展示，会选择在 16 周上机时展示项目的 DEMO
   b) 不一定会安排展示，但是 PPT 必须要有
   c) 如果是具有较好展示性的项目，请提供项目实机视频，对展示性好的组考虑加分
7. 对于**基于现实需求而编写的项目**，需要提供一份说明项目的灵感来源的文档（可以百度软件工程的**需求文档**），无字数要求
8. 重点 ：提交一份生成的 JavaDoc（html）

C. 评分依据

> 以原创为前提，提倡结合实际进行开发，**杜绝拷贝和代做程序，一旦发现，整组记 0**
>
> - 请尊重他人的代码成果，使用第三方库必须要指明来源

以下是具体评分标准

1. 工作量（约 40%）

    - 根据小组人数粗略裁定，建议在 2k~3k 行代码，安卓开发以及一些框架的使用不适用此标准
        - 框架环境代码或与 Java 项目不相关的代码（如前端Html、CSS、JavaScript代码）不属于代码统计的范畴
    - 请不要刻意使用大量空行、大量注释、大量冗余代码来充数（如果发现，会进行一定的惩罚）
2. 代码结构的设计和使用技术的水平（约 20%）
3. 产品的使用体验（约 25%）
    - 易用性（包括清晰的操作引导和符合常理的操作方式）
    - 错误处理与反馈（鲁棒性）
4. 各个文档内容的完整性（约 10%）
5. 小组成员分工的合理度（约 5%）

D. 一些建议

这一部分不要求必须实现，但是为了分数着想建议根据自己的项目进行酌情考虑。

- 功能实现
    - 对于给出的使用说明书，用户至少能够在只根据界面上的内容和提示，以及阅读使用说明书的情况下就能够正常使用所有功能。**不应该让用户去猜测下一步要如何操作，也不应该报出让用户摸不着头脑的错误和异常**。通俗的说，找几个其他同学，让他们试用，试用者体验时不会给出较差的评价，就基本达到要求了
- 错误处理
    - 对于各种情况，应该提供正确的错误处理（构建新文件、超时重试、反馈错误后退出等）
        - 在任何情况下，程序不应该因为 `Exception` 而爆炸
        - 对于使用菜单式 `CLI` 的同学， `Ctrl+Z` 与 `Ctrl+C` 这种强制炸程序的操作可以不考虑
        - 对于使用 `GUI` 的同学，对于任务管理器和 `kill` 这种强制杀程序的操作可以不考虑
        - 对于使用 `GUI` 的同学，当执行窗体关闭时，应该能够对用户操作进行确认，根据设计对于部分数据进行存储
- 美观
    - 这是一个很主观的标准，因此不会有很多约束，但是至少别继续使用十几年前动感地带的 UI 风格
- 通过连接数据库或文件读写来实现持久化
- 项目结构合理
- 注释
    - 所有源文件，按照 javadoc 的标准格式打上注释。
- 测试
    - 按照未来的软工课的标准，给出完整的测试计划，至少包括单元测试和集成测试。对于单元测试，还应该给出测试代码并指明测试框架
- 设计
    - 给出一个类图（或设计文档）描述你的代码设计。设计上的东西很难评价，遵循面向对象设计准则、高内聚低耦合等说法很难映射到具体的代码上