# 高级语言程序设计大作业实验报告

南开大学 计算机大类

姓名 董朝阳

学号 2411870

班级 0978

2025年5月13日

高级语言程序设计

实验报告

## 目录

- 一、作业题目

- 二、开发软件

- 三、课题要求

- 四、主要流程

- 1. 整体流程

- 2. 算法或公式

- 3. 单元测试

- 五、单元测试

- 六、收获

# 高级语言程序设计大作业实验报告

## 一、作业题目

基于C++的德州扑克图形化游戏开发

## 二、开发软件

[使用的开发软件名称，例如Visual Studio 20XX]

## 三、课题要求

1. 使用C++语言进行开发，构建一个图形化的德州扑克游戏程序。

2. 运用面向对象编程思想，对游戏中的牌、牌堆、玩家手牌、公共牌以及游戏逻辑等进行合理的类设计与实现。

3. 实现基本的游戏流程，包括发牌、下注、判断胜负等功能，并在图形界面中进行展示。

4. 对关键的游戏逻辑和功能进行测试，确保程序的正确性和稳定性。

## 四、主要流程

### 1. 整体流程

游戏初始化时，创建`Deck`对象代表牌堆并进行洗牌操作，同时创建`HoleCards`对象分别表示玩家和电脑的手牌，以及`std::vector<Card>`对象存储公共牌。游戏主循环中，先绘制游戏界面，包含背景、筹码信息、手牌和公共牌、操作按钮等元素。通过监听鼠标点击事件，根据点击位置处理玩家的跟注、加注、弃牌等操作。电脑根据简单AI逻辑进行相应操作，游戏进入下一阶段时，从牌堆中发出公共牌。游戏结束时，根据玩家和电脑的手牌以及公共牌，使用`Evaluator`类判断胜负，并展示结果。

### 2. 算法或公式

- \*\*牌型判断算法\*\*：在`Evaluator`类中，通过`analyzeCombo`函数分析牌型。先对传入的牌按点数从大到小排序，判断是否为同花（`checkFlush`函数）和顺子（`checkStraight`函数），统计每种点数的牌的数量（`countRanks`函数）。根据这些结果，按照皇家同花顺、同花顺、四条、葫芦、同花、顺子、三条、两对、一对、高牌的顺序依次判断牌型。

- \*\*组合生成算法\*\*：使用`generateCombinations`函数生成从给定牌中选取5张牌的所有组合，用于分析不同牌型组合下的牌力。采用`std::vector<bool>`作为掩码，通过`std::prev\_permutation`函数对掩码进行排列，生成不同的组合。

### 3. 单元测试

- 针对牌型判断功能，设计多组测试用例，涵盖各种牌型的组合，如皇家同花顺、同花顺、四条等，验证`Evaluator::evaluateHand`函数是否能正确判断牌型。

- 测试`Deck`类的`reset`、`shuffle`、`deal`等函数，确保牌堆的初始化、洗牌和发牌功能正常。

- 对`HoleCards`类的`receiveCards`、`getCards`、`hasCards`、`clear`等函数进行测试，验证手牌的接收、获取、状态判断和清空功能。

## 五、单元测试

### 测试案例定义

|测试内容|输入|预期输出|目的|

|---|---|---|---|

|牌型判断：皇家同花顺|包含红心10、J、Q、K、A的牌组|HandRank::ROYAL\_FLUSH|验证皇家同花顺牌型判断的正确性|

|牌型判断：同花顺|包含方块5、6、7、8、9的牌组|HandRank::STRAIGHT\_FLUSH|验证同花顺牌型判断的正确性|

|牌型判断：四条|包含4个A和1个其他牌的牌组|HandRank::FOUR\_OF\_A\_KIND|验证四条牌型判断的正确性|

|Deck类测试：reset函数|调用Deck对象的reset函数|牌堆包含52张按顺序排列的牌|验证牌堆重置功能|

|Deck类测试：shuffle函数|调用Deck对象的shuffle函数多次|每次洗牌后牌堆顺序不同|验证牌堆洗牌功能|

|Deck类测试：deal函数|多次调用Deck对象的deal函数|每次返回不同的牌，牌堆数量减少|验证牌堆发牌功能|

|HoleCards类测试：receiveCards函数|调用HoleCards对象的receiveCards函数，传入Deck对象|HoleCards对象包含两张牌，hasCards函数返回true|验证手牌接收功能|

|HoleCards类测试：getCards函数|调用已接收手牌的HoleCards对象的getCards函数|返回包含两张手牌的std::vector<Card>|验证手牌获取功能|

|HoleCards类测试：hasCards函数|调用已接收手牌和未接收手牌的HoleCards对象的hasCards函数|已接收手牌返回true，未接收手牌返回false|验证手牌状态判断功能|

|HoleCards类测试：clear函数|调用已接收手牌的HoleCards对象的clear函数|hasCards函数返回false，手牌数量为0|验证手牌清空功能|

### 测试结果

在测试过程中，发现部分牌型判断边界情况处理不当，如顺子中A-2-3-4-5的特殊情况判断存在遗漏。经过修改`checkStraight`函数，重新测试后所有牌型判断测试用例通过。对于`Deck`类和`HoleCards`类的测试，未发现明显问题，各项功能均能正常运行。

## 六、收获

### 1. 面向对象编程的深入理解

通过设计`Card`、`Deck`、`HoleCards`、`Evaluator`等类，更加深入地理解了面向对象编程中的封装、继承和多态特性。将不同的功能封装在各自的类中，提高了代码的模块化和可维护性。例如，`Card`类封装了牌的花色和点数信息及相关操作，`Deck`类负责牌堆的管理，`Evaluator`类专注于牌型判断，各个类之间职责明确。

### 2. 图形化编程的实践

使用[具体图形库名称，如EasyX]进行图形化界面开发，掌握了图形绘制、图像加载、鼠标事件处理等基本操作。学会了如何将游戏逻辑与图形界面相结合，为用户提供直观的游戏体验。在图形化编程过程中，遇到了图片加载失败、界面闪烁等问题，通过查阅资料和调试，掌握了相应的解决方法，提升了图形化编程能力。

### 3. 算法设计与优化

在实现牌型判断算法和组合生成算法过程中，锻炼了算法设计和优化能力。最初的牌型判断算法逻辑不够清晰，导致部分复杂牌型判断错误，经过多次优化和调整，使算法更加简洁高效。同时，通过学习和运用`std::prev\_permutation`等标准库函数，简化了组合生成算法的实现，提高了代码的可读性和性能。