浙江傳媒學院
COMMUNICATION
UNIVERSITY
OF ZHEJIANG

短学期实践实训（总结）报告

学 院 名 称：媒体工程学院

课程（项目）名称：程序能力训练

学　　号：_____

姓　　名：_____

班　　级：_____

同 组 同 学：_____

_____

任 课 教 师：_____

学　　期：2025-2026 学年短学期

《明日方舟》六星干员管理信息系统及图形化显示

1 选题与分工

A) 意义：选题为《明日方舟》六星干员查询管理系统意义在于供以游戏用户查询了解六星干员的信息；

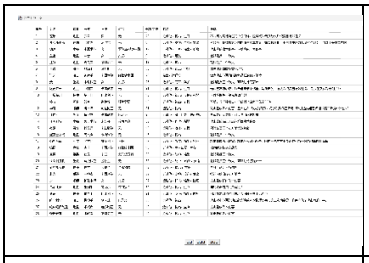
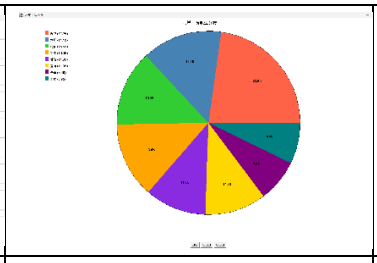
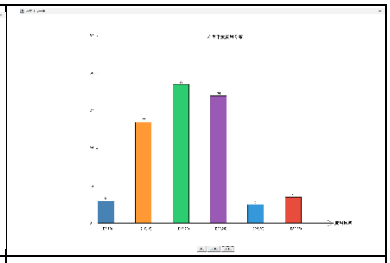
B) 系统任务：针对现有干员信息构成的 CSV 文件数据进行增删改查并对该数据以图标和柱状图等统计形式展示在可视化界面；

C) 分工：本人负责 Java 部分，实现数据可视化；**负责 C 语言部分，实现数据增删改查。

2 系统的设计

Java 部分：

通过按钮一、二、三，我们可以在“表格/饼图/柱状图”之间自由切换，全程无弹窗、无命令行，适合快速浏览《明日方舟》六星干员分布情况。

		
点击按键一	点击按键二	点击按键三

3 系统的实现

Java 部分：

A) 数据结构

本部分采用 Java 标准集合框架中的 ArrayList<DataRow> 作为内存数据集。DataRow 为不可变包装类，内部仅持有一个 String[]，用于保存 CSV 中的一行原始字段，既保证轻量又避免外部修改。全局静态容器 CsvReader.DATA 被所有模块共享，承担“数据总线”角色，后续的分析、绘图均直接读取该列表，逻辑清晰且节省复制开销。

```java
package ark;
import java.io.*;
import java.util.ArrayList;
import java.util.List;
public class CsvReader {
    public static final List<DataRow> DATA = new ArrayList<>();
```

B) 核心代码

① CSV 读取（CsvReader.java）

采用 BufferedReader + InputStreamReader("GBK") 显式指定编码，防止中文乱码。

按行循环 readLine()，跳过空行；对每行直接执行 split(",") 得到字段数组并包装成 DataRow 后装入 DATA。

异常捕获仅打印简短提示，不中断主流程，符合图形化程序“启动即展示”的需求。

```java
public class CsvReader {
    public static final List<DataRow> DATA = new ArrayList<>();


        public static void readCsvFile() {
            try (BufferedReader br = new BufferedReader(
                    new InputStreamReader(new FileInputStream("Officers.csv"), "GBK"))) {
                String line;
                while ((line = br.readLine()) != null) {
                    if (line.trim().isEmpty()) continue;
                    DATA.add(new DataRow(line.split(",")));
                }
            } catch (IOException e) {
                System.out.println("文件读取错误: " + e.getMessage());
            }
        }
    }
```

② 数据分析（DataAnalyzer.java）

单次顺序扫描即可完成两项统计：

职业分布：Map<String,Integer> OCCUPATION_COUNT 累加 arr[2]。

费用分布：将 arr[6] 解析为 int，调用 getCostInterval() 划入 6 个左闭右开区间，结果存入 COST_DISTRIBUTION。

两段逻辑均从 i=1 开始，跳过表头；解析失败时捕获 NumberFormatException 并忽略，保证健壮性。

```java
package ark;
import java.util.*;
public class DataAnalyzer {
    public static final Map<String, Integer> OCCUPATION_COUNT = new HashMap<>();
    public static final Map<String, Integer> COST_DISTRIBUTION = new HashMap<>();
    public static final String[] COST_INTERVALS = {
            "【5,10)", "【10,15)", "【15,20)", "【20,25)", "【25,30)", "【30,35)"
    };
    public static void analyze() {
        if (CsvReader.DATA.isEmpty()) return;
        for (int i = 1; i < CsvReader.DATA.size(); i++) {// 职业
            String[] arr = CsvReader.DATA.get(i).getData();
            if (arr.length >= 3) {
                String occ = arr[2];
                OCCUPATION_COUNT.put(occ, OCCUPATION_COUNT.getOrDefault(occ, 0) + 1);
            }
        }
        for (int i = 1; i < CsvReader.DATA.size(); i++) {// 费用
            String[] arr = CsvReader.DATA.get(i).getData();
            if (arr.length >= 7) {
                try {
                    int cost = Integer.parseInt(arr[6]);
                    String interval = getCostInterval(cost);
                    COST_DISTRIBUTION.put(interval, COST_DISTRIBUTION.getOrDefault(interval, 0) + 1);
                } catch (NumberFormatException ignore) {
                }
            }
        }
    }
    private static String getCostInterval(int cost) {
        if (cost >= 5 && cost < 10) return "【5,10)";
        if (cost >= 10 && cost < 15) return "【10,15)";
        if (cost >= 15 && cost < 20) return "【15,20)";
        if (cost >= 20 && cost < 25) return "【20,25)";
        if (cost >= 25 && cost < 30) return "【25,30)";
        if (cost >= 30 && cost < 35) return "【30,35)";
        return "【其他】";
    }
}
```

③ 图形化展示（DrawCanvas.java）

继承 Canvas，全程手写绘制，不依赖第三方图表库。

双缓冲机制：在 paint() 中维护一张 offScreen 图像，所有元素先画到内存，再一次性 drawImage 到屏幕，杜绝闪烁。

三种视图通过 currentMode 切换，同一组件复用，降低资源消耗。

```
switch (GuiBuilder.currentMode) {
    case GuiBuilder.MODE_TABLE -> drawTableMode(g);
    case GuiBuilder.MODE_PIE   -> drawPieChartMode(g);
    case GuiBuilder.MODE_BAR   -> drawBarChartMode(g);
}
```

表格模式

动态计算列宽：遍历所有行取最大字符串像素，加 CELL_PADDING，确保内容不截断。

自主实现垂直滚动：右侧留 16 px 滚动条，支持鼠标拖拽、点击翻页与滚轮事件；

scrollOffset 与 thumbHeight 按数据行数比例映射，体验与原生控件一致。

```
private void drawTableMode(Graphics g) {
    calculateColumnWidths(g);
    int visibleRows = getVisibleRows();
    int startRow = scrollOffset / ROW_HEIGHT;
    int endRow   = Math.min(startRow + visibleRows, CsvReader.DATA.size());
    g.setFont(g.getFont().deriveFont(Font.BOLD, 12));
    drawTableRow(g, CsvReader.DATA.get(0), TABLE_X, TABLE_Y, true);
    g.setFont(g.getFont().deriveFont(Font.PLAIN, 12));
    for (int i = startRow; i < endRow; i++) {
        if (i == 0) continue;
        int rowY = TABLE_Y   + (i - startRow +1 ) * ROW_HEIGHT;
        drawTableRow(g, CsvReader.DATA.get(i), TABLE_X, rowY, false);
    }
    drawScrollbar(g);
}
```

饼图模式

基于 OCCUPATION_COUNT 排序后顺时针绘制，

预设 8 组柔和配色循环使用，图例置于左侧，标题置顶，布局均衡。

```
List<Map.Entry<String,Integer>> sorted =
        new ArrayList<>(DataAnalyzer.OCCUPATION_COUNT.entrySet());
sorted.sort(Map.Entry.<String,Integer>comparingByValue().reversed());
```

```
g.setFont(g.getFont().deriveFont(Font.BOLD, 16));
g.drawString("六星干员职业分布", centerX - 100, CHART_PADDING / 2);
```

```
g.fillRect(100, legendY, 15, 15);          // 左侧色块
g.drawString(String.format("%s (%.1f%%)", occ, pct), 120, legendY + 15);
legendY += 25;                            // 把他们竖排
```

柱状图模式

六段费用区间固定，Y 轴采用“友善刻度”算法 calculateNiceNumber()，自动向上取整并

均分 5 格，刻度线、数值标签齐全。

```
public static final String[] COST_INTERVALS = {
        "【5,10)", "【10,15)", "【15,20)", "【20,25)", "【25,30)", "【30,35)"
};
```

每组柱子使用不同颜色填充并加黑框，顶部标注具体人数，直观对比各区间人数差异。

```
final Color[] BAR_COLORS = { ... 6 组柔和颜色 };
...
g.setColor(BAR_COLORS[i % BAR_COLORS.length]);
g.fillRect(barX, barY, BAR_WIDTH, barH);   // 先填色
g.setColor(Color.BLACK);
g.drawRect(barX, barY, BAR_WIDTH, barH);   // 再加黑框
```

④ 界面框架（GuiBuilder.java）

使用 AWT 而非 Swing/JavaFX，减少依赖、启动更快。

```
import java.awt.*;
import java.awt.event.*;
```

顶层 Frame 采用 BorderLayout：中心放 DrawCanvas，南部放按钮面板。

```
Frame frame = new Frame("六星干员一览");
frame.setLayout(new BorderLayout());
frame.add(canvas, BorderLayout.CENTER);
frame.add(btnPanel, BorderLayout.SOUTH);
```

三个按钮切换模式后仅调用 canvas.repaint()，不重建对象，性能开销极小。

```
b1.addActionListener(e -> { currentMode = MODE_TABLE; canvas.repaint(); });
b2.addActionListener(e -> { currentMode = MODE_PIE;  canvas.repaint(); });
b3.addActionListener(e -> { currentMode = MODE_BAR;  canvas. Repaint(); });
```

4 项目总结

项目虽然相对简陋但是基本要求功能已然全部实现，通过这次项目实践对面向对象的编程思想认知更加深刻，对 Java 程序的信息交互了解更深尤其是在编码方面理解更加深入，做了一个看似简单的项目但是各种方法面收获颇丰，坚定了我们对学完知识就要做项目实践的学习方法，实践才能快速将知识融会贯通。