1. 分析和设计

1.1 基于模型-视图的整体结构设计

本系统基于模型-视图的结构进行设计。其中，模型层负责处理数据和业务逻辑，它独立于用户界面和业务流程，提供对数据的操作和管理。模型层可以负责与本地文件进行交互，进行数据的存储和传输。

视图层负责用户界面的呈现和用户交互。它包含用户界面组件、样式、布局和交互逻辑。视图层通过模型层提供的数据和服务来展示数据、接收用户输入，并将用户的操作传递给控制层进行处理。

控制层负责协调模型层和视图层之间的交互和数据流动。它接收用户的输入、处理用户操作，并将操作传递给模型层进行数据处理和业务逻辑处理。控制层充当模型和视图之间的中介，负责处理用户事件、更新视图以及与模型层的交互。

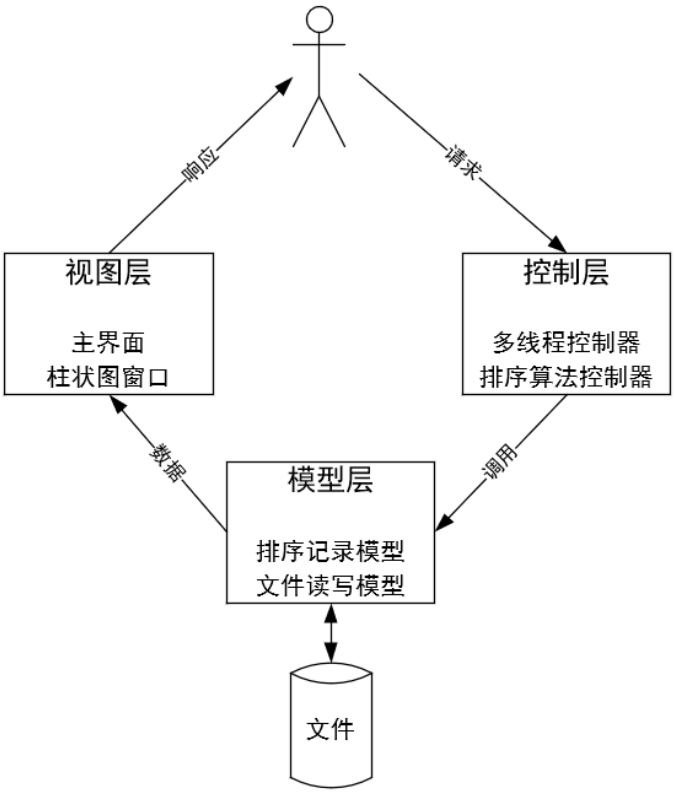


图 1.2 基于模型-视图的整体设计图

1.2 功能模块划分

如图 1.2 所示，系统功能分为三个模块，分别是数字输入及排序种类选择模块、排序过程显示模块和历史记录查看模块。其中，排序过程显示功能又可以分

为表格显示和柱状图绘制，而历史记录查看功能可以分为历史输入字符串、历史排序表格以及历史排序柱状图的查看。

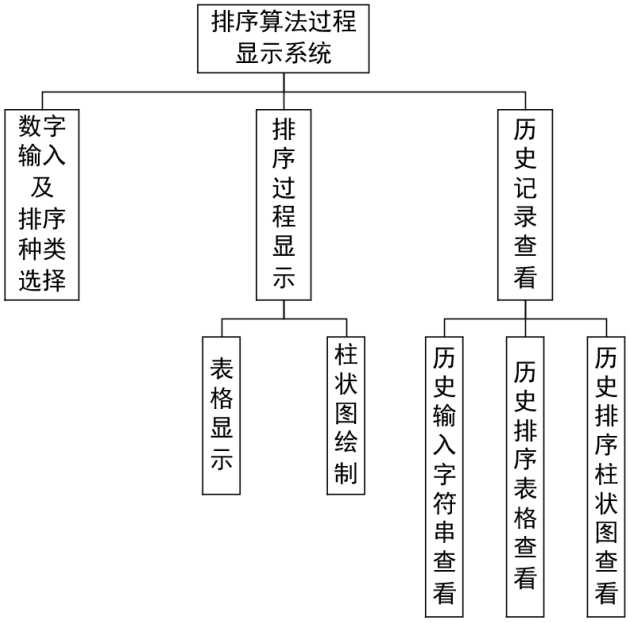


图 1.2 系统模块图

在数字输入及排序种类选择功能中，用户可以在文本框中输入一串想要排序的数字，然后从冒泡排序、选择排序和插入排序中选择自己想要使用的排序方式，点击相应的按钮。

在排序过程显示功能中，系统将通过算法处理用户输入的数字，将排序过程中每一次交换数字后的数组都记录进列表。用户可以查看系统排序过程中所对应的表格，也可以点击“显示数组变化图”按钮，查看排序过程所对应的柱状图。

在历史记录查看功能在，用户可以回放历史记录，查看之前在系统中输入过的所有记录。选中某一条记录后，可以直接查看之前排序过程所对应的表格和柱状图。

1.3 类和对象分析与设计

系统的类图如图 1.3 所示。

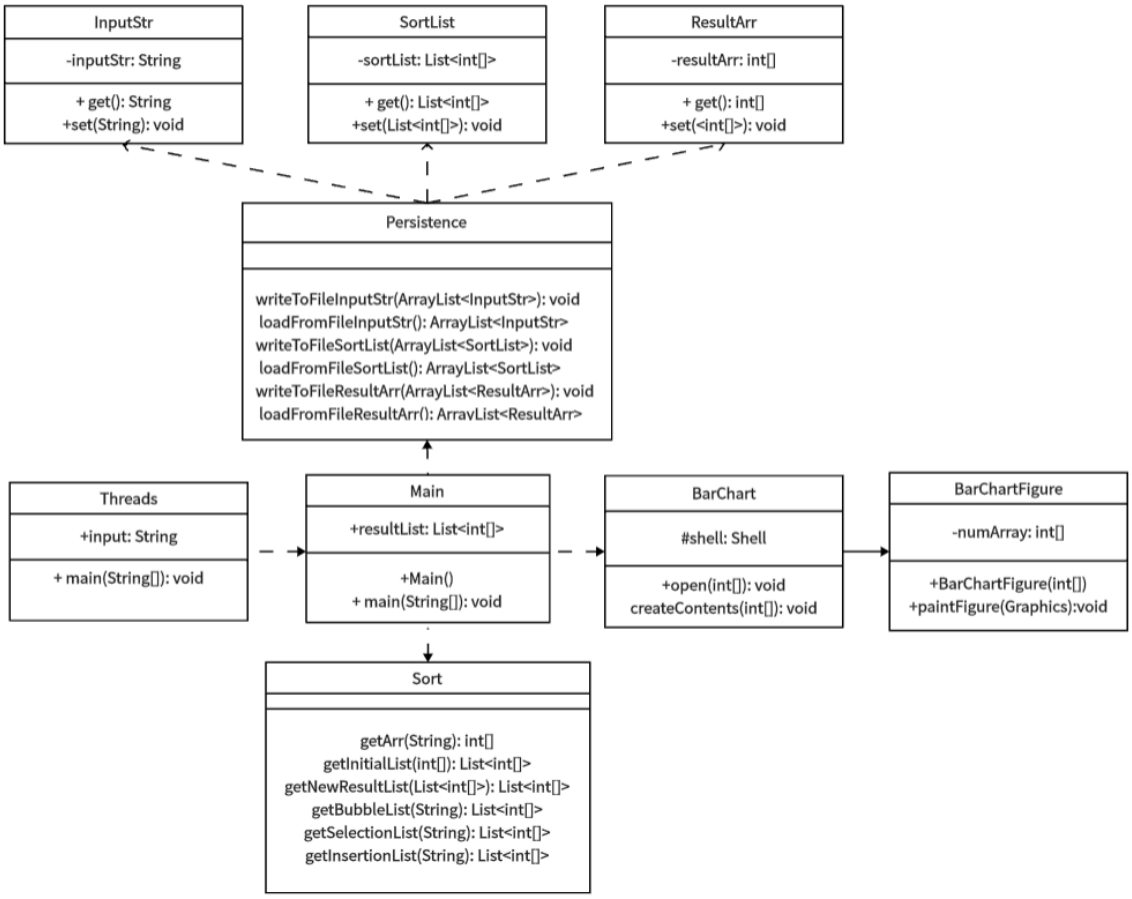


图 1.3 系统类图

1.3.1 模型层设计

（1）InputStr 类

InputStr 类用于保存算法输入参数，定义用户每次排序输入的字符串 str，并设置 get()和 set()方法，因此可在其它类中修改、读取该类的参数。该类实现了 Serializable 接口，表示其对象可以在网络或文件之间进行序列化和反序列化操作，以便进行对象的持久化或传输。

（2）SortList 类

SortList 类用于保存算法执行过程。定义排序过程列表 list 并设置 get()和 set()方法，将一次排序过程中每进行一次数字交换后所更新的数组的都保存到列表中。该类也实现了 Serializable 接口。

（3）ResultArr 类

ResultArr 类用于保存算法执行结果。定义排序结果数组 arr 并设置 get()和 set()方法，描述排序结果数组。该类也实现了 Serializable 接口。

1.3.2 控制层设计

（1）Sort 类

Sort 类是一个用于排序的业务类，主要实现了三种排序算法：冒泡排序（getBubbleList）、选择排序（getSelectionList）和插入排序（getInsertionList）。该类还包含了分割字符串、添加初始序号、删除排序交换序号等方法，并返回处理结果。

（2） Threads 类

Threads 类用于实现多线程功能。运行 Threads 类后，按照指示在控制台输入启动信息，就可以开启一个新的 Main 线程。

（3） Persistence 类

Persistence 类实现了对象持久化功能，可以将 ArrayList<InputStr>、ArrayList<SortList>和 ArrayList<ResultArr>写入文件并从文件中读取。类中的 writeToFileInputStr(ArrayList<InputStr>)方法可以将给定的 ArrayList<InputStr>对象写入文件，loadFromFileInputStr ()方法从文件中读取 ArrayList<InputStr>对象并返回。其他两种对象列表同理。

1.3.3 视图层设计

（1）Main 类

系统主界面创建了 JTextField 类型的文本框和 JButton 类型的按钮，创建 Jtable 并设置模型，然后将它们都加入 JPanel 类型的面板中，将面板加入 Jframe 类型的主窗口。Jtable 初始时不设置表头和表格内容，后续通过点击事件监听器来设置其显示内容。

如图 1.4 所示，主界面一共有五个按钮，将点击事件监听器注册到不同的按钮上，然后在事件监听器中实现相应的事件处理方法，以实现不同的功能。

图 1.4 系统主界面

三种排序方式的按钮对应的事件监听处理程序类似，此处以冒泡排序对应的按钮 bubbleSortButton 为例进行介绍。在程序中，首先将 Sort 类实例化，调用相关方法以获取原始数组、包含排序交换过程的列表 resultList 和排序结果数组 numArray。然后，根据 numArray 的大小，使用 addColumn 方法来为表格添加表头，告知每一列各展示什么内容。接着，根据 resultList，使用 addRow 方法来向表格中逐行添加数字。在展示完排序过程中数组变化过程之后，还需要将 InputStr 类、SortList 类和 ResultArr 类实例化，根据相关方法来将本次排序记录存储到本地文件。

在“显示数组变化图”按钮所对应的事件监听处理程序中，需要调用 BarChart 类，绘制排序过程中数组的变化过程对应的柱状图，该类将在下文详细介绍。

在“查看历史纪录”按钮所对应的事件监听处理程序中，首先创建 Persistence 类型的对象 per，然后从 inputStr.txt、sortList.txt 和 resultArr.txt 文件中分别获取存储着 InputStr 类、SortList 类和 ResultArr 类对象的列表。然后读取每个对象中的字符串属性，获得用户所输入的字符串历史记录，创建一个弹窗并将历史记录显示在下拉框中。下拉框通过创建 JComboBox<String>类来实现，可以将历史记录字符串逐个显示在其中。

（2）BarChart 类

BarChart 类使用 SWT 库来创建图形用户界面，包含了两个可以被调用的方法。在 createContents(int[] nowArray)方法中，创建了一个新的 Shell 类型的窗口 shell。然后创建一个 FigureCanvas 对象，添加到 shell 窗口中。FigureCanvas 也是 SWT 的一个部件，用于在图形界面中显示图形。

在 open (int[] nowArr)方法中，首先调用 createContents 方法，然后设置窗口位置，让其显示在界面上。

（3）BarChartFigure 类

BarChartFigure 类是一个继承自 Eclipse Draw2D 的 Figure 类的自定义图形类，用于绘制柱状图。它接收一个整型数组 numArray 作为构造函数参数，然后根据数组内数字大小绘制相应高度的柱状图。每个柱子的宽度为 50，柱子之间有一个间隔为 20。柱子的高度通过将数组中的元素乘以 10 来计算，并以蓝色背景和黑色边框进行绘制。然后在每个柱子上添加标签，显示该柱子对应的数字大小。

1.4 系统过程分析与设计

在用户进入系统后，主界面会显示初始的文本输入框和交互按钮。用户输入想要进行排序的一串数字后，可以选择需要使用的排序种类。系统会根据用户所点击的按钮来判定要进行哪种排序方式，进入按钮所对应的事件监听处理程序。

在处理程序中，系统会将排序过程中每一次交换后改变的数组保存到一个列表中，并保存排序结果，然后将列表中的数字显示成表格。最后，会将本次排序相关数据序列化，存入文件。

若用户接下来点击“显示变化图按钮”，则系统开启新窗口，绘制数组对应的柱状图；若点击“查看历史记录”按钮，则系统访问文件，弹出历史记录窗口，展示过去输入过的所有字符串，然后在用户点击某个字符串记录后加载其对应的表格和变化图。

系统流程图如图 1.5 所示。

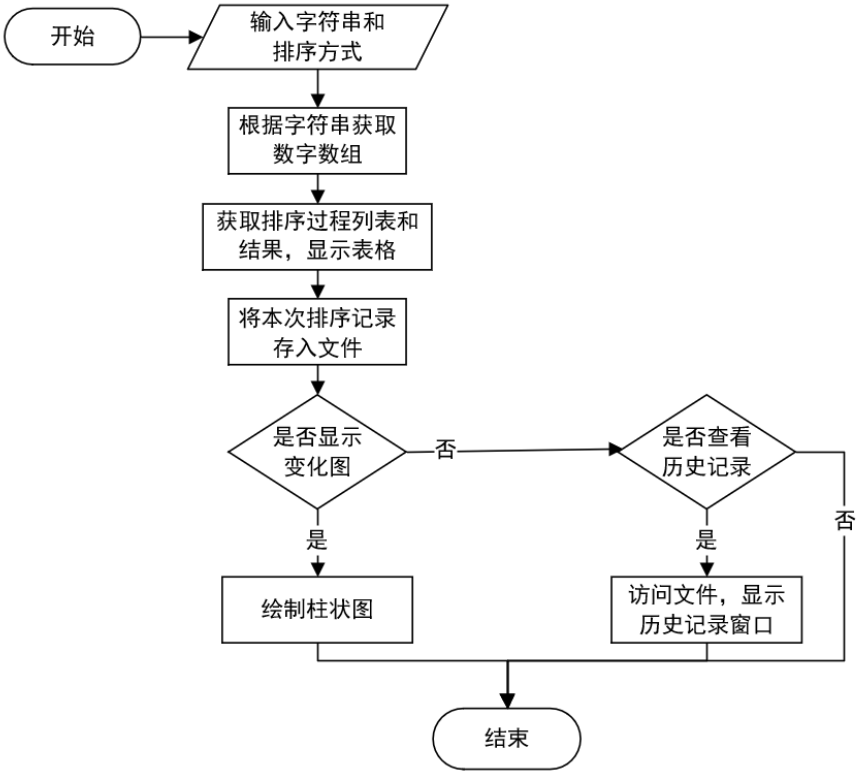

开始
输入字符串和
排序方式
根据字符串获取
数字数组
获取排序过程列表和
结果，显示表格
将本次排序记录
存入文件
是否显示
变化图
否
是否查看
历史记录
是
是
否
绘制柱状图
访问文件，显示
历史记录窗口
结束

图 1.5 系统流程图

2. 系统实现

2.1 系统应用的关键技术说明

2.1.1 Swing

Swing 是 Java 的默认图形用户界面（GUI）工具包，它提供了一套丰富的组件，用于创建跨平台的桌面应用程序。Swing 组件使用 Java 代码编写，并通过 Java 虚拟机渲染。Swing 提供了许多常见的用户界面元素，如按钮、文本框、下拉框、列表和表格等，它具有灵活性和可定制性，可以通过添加监听器和自定义绘制进行扩展和个性化。其中，JTable 是 Swing 提供的一个表格组件，用于展示和编辑数据，可以显示数据的多行和多列。

本系统运用了 Swing 图形组件来显示数字输入框和交互按钮，通过为按钮添加点击事件监听器的方式来实现排序功能的触发。使用 JTable 图形组件显示算法执行过程中每一次排序交换后数组的变化过程。将 Draw 2D 库和 SWT 技术结合使用，根据数组中数字大小来绘制柱状图，将变化过程进行可视化展示。接下来对着三大技术进行简要介绍。

2.1.2 Draw 2D 和 SWT

SWT 是另一个用于构建 Java 桌面应用程序的图形用户界面（GUI）工具包。SWT 是由 Eclipse 基金会开发的，被广泛用于构建 Eclipse IDE 及其插件。SWT 使用本地操作系统的原生组件，因此在一些方面比 Swing 更加高效和快速。

Draw 2D 是 Java 平台的绘图库，它具有丰富的功能和灵活性，可以与其他 Java GUI 技术（例如 SWT）结合使用，为应用程序提供出色的可视化效果。它提供了一套强大的工具和类，用于在 2D 平面上进行图形和图像的绘制。Draw 2D 库允许创建和绘制各种形状，如直线、矩形、圆形、椭圆等，可以通过指定坐标、尺寸、颜色和样式来控制绘制的图形；具有可扩展的渲染器架构，可以使用不同的渲染器实现来绘制图形，还支持颜色管理，以确保图形在不同设备和显示器上的颜色一致性。

2.1.3 对象持久化

除此之外，系统还运用了对象持久化技术来保存排序历史记录。对象持久化是将对象转换为字节流的过程，也称为对象序列化，是指将对象的状态转换为可存储或传输的形式，以便在需要时重新创建或恢复该对象。对象持久化可以用于

数据存储，能将对象保存在磁盘或数据库等持久性存储介质中。这样，即使程序结束或计算机重新启动，对象的状态仍然可以被保留，而不会丢失。

2.2 关键功能设计

（1）排序功能

用户进入系统，在输入框内输入一串数字并以逗号间隔，然后系统要对字符串进行处理，获取初始待排序数组，然后分别编写三种排序算法的对应代码，将每一次进行两数交换后的数组都保存在一个列表中，返回排序过程列表和排序结果。

（2）表格显示功能

系统要根据排序过程列表，使用 JTable 图形组件显示排序算法执行过程中数组的变化，将数字以二维表格的形式显示在界面。

（3）变化图绘制功能

用户点击“查看数组变化图”按钮后，系统要根据排序算法中获得的排序过程列表中读取所有数组，根据数字大小绘制柱状图。数字越大，则对应的柱子高度越高。根据列表中数组的数量，绘制相应数量的柱状图，展示数组的变化过程。

（4）对象持久化功能

用户每次排序后，系统都要保存算法输入参数、算法执行过程和算法执行结果。系统通过使用 ObjectOutputStream 和 ObjectInputStream 来实现对象的序列化和反序列化，将 ArrayList<InputStr>、ArrayList<SortList>和 ArrayList<ResultArr>以二进制形式分别写入文件。这样可以实现数据的持久化存储和加载，将数据保存在本地文件中，以便后续的读取和使用。

（5）多线程功能

系统要能同时开启多个线程，当用户在控制台输入规定的字符串后，系统会启动一个新线程，并在该线程中创建并启动一个 Main 对象，这样就实现了多线程调用 Main 类的功能。若用户同时开启两个线程，在一个线程中执行一次排序之后，立刻在另一个线程中查看历史记录，就能够看到刚才输入的记录。

2.3 关键用户界面设计

进入系统后，主界面如图所示。用户可以在左上角文本框输入想要排序的一串数字，以中文逗号间隔，然后在右侧的三个按钮中选择需要进行哪种排序方式。

如图 2.1 所示，输入“5，7，6，4，8”，然后点击冒泡排序，就可在界面中间看到能展示排序交换过程的表格。

图 2.1 输入数字举例

用户点击界面下方的“显示数组变化图”按钮，就可以看到刚才那串数组对应的柱状图，柱子的高度能够反映数字的大小。如图 2.2 所示。

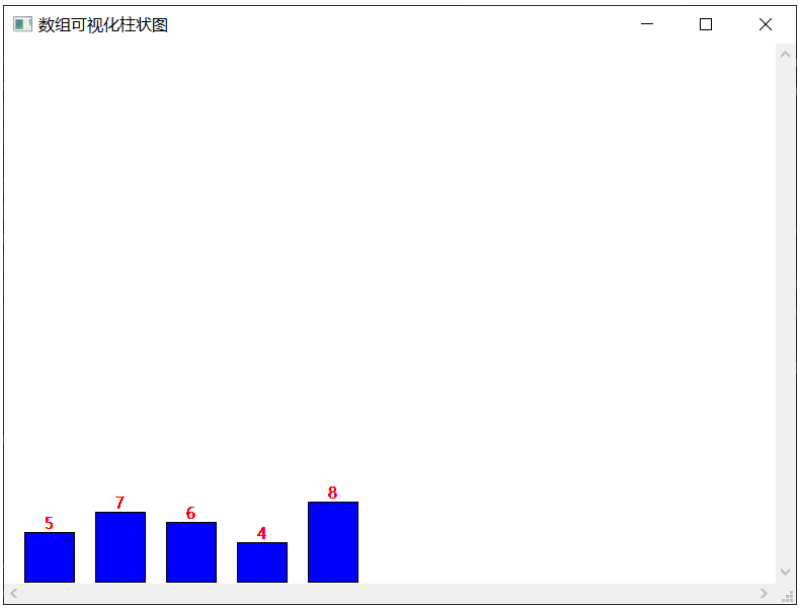


图 2.2 柱状图展示

用户点击一下弹窗右上角的叉号，则可以看到数组下一次交换后所对应的柱状图。交换四次后，排序完成。四次交换过程中的柱状图如图 2.3 所示。

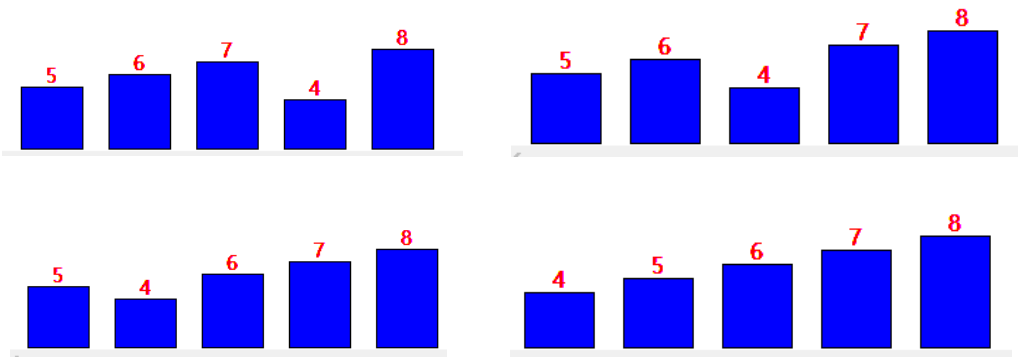


图 2.3 柱状图变化图

用户点击“查看历史纪录”按钮，将弹出一个历史记录窗口。点击下拉框，可以看到之前输入过的所有记录。如图 2.4 所示。

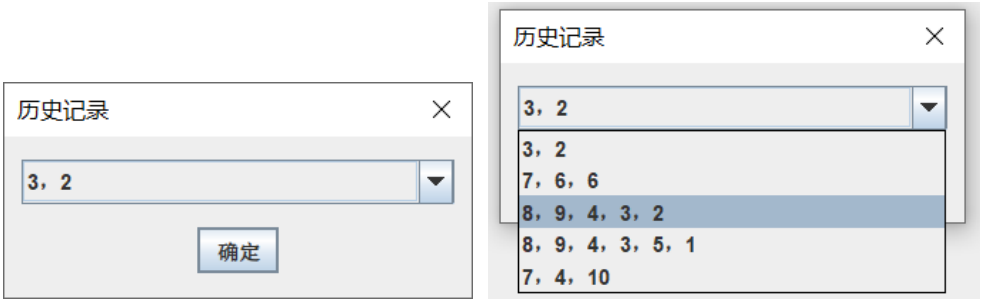


图 2.4 历史记录查看

选择其中一条，就可以查看与之对应的排序交换表格和柱状图。系统将根据选中的字符串，查找对象列表中与之匹配的对象，获取其对应的排序过程列表和结果数组。例如选择“7，6，6”，则界面变成如图 2.5 所示。此时若点击“查看历史记录”按钮，依然可以查看所选择排序的数组对应的柱状图。

图 2.5 历史记录选中展示

此时可以再运行 Threads 类来打开一个线程，在其中一个线程中输入一串新数字并点击冒泡排序，然后立刻在另一个线程中查看历史记录，可以发现已经出现了刚刚输入的字符串。如图 2.6 所示。

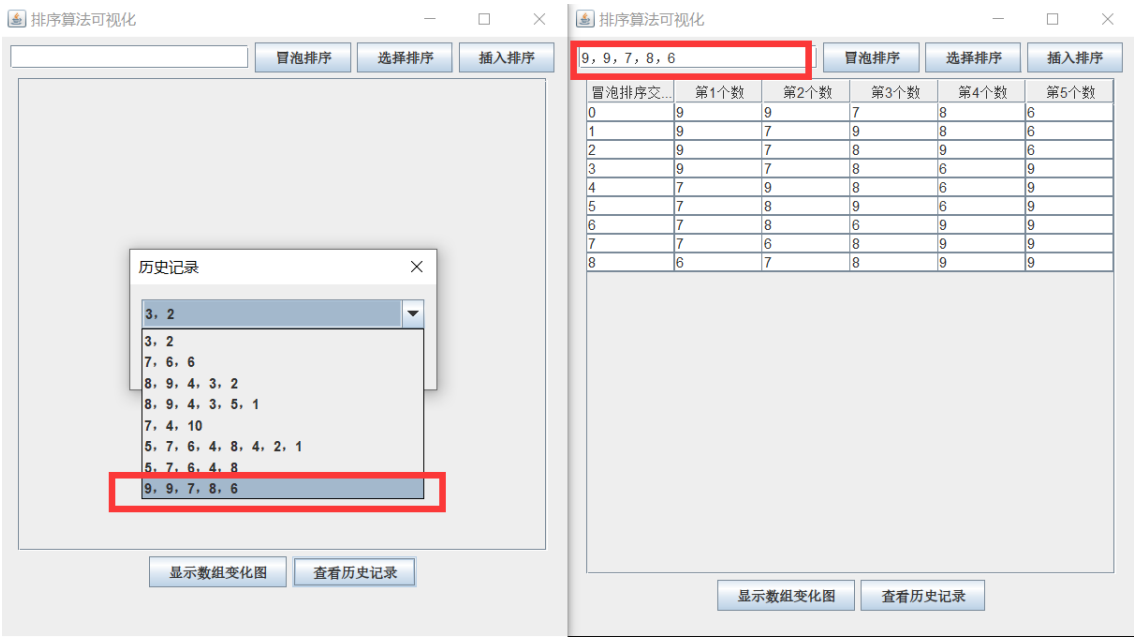


图 2.6 多线程功能展示

2.4 关键业务类或者业务过程实现说明

（1）数组排序

Sort 类中的 getArr 方法可以处理用户输入的字符串，用逗号来分割字符串，将其转换为原始数组 numArray。定义结果列表 resultList，将 numArray 添加序号 0 然后加入其中。

在排序过程中，定义 sortTime 表示交换次数序号，设定初始值为 1，在每一次交换两数之后，将 sortTime 插入当前数组的第 0 个位置，将得到的数组添加进 resultList，然后让 sortTime 加一，开启下一轮排序。当排序完成后，返回最终的 resultList 列表，其中就包含了排序过程中的每一步结果。

（2）表格显示

系统使用 model.addColumn 方法为表格添加表头。首先，添加一个 "冒泡排序交换次数" 的表头。然后，使用 for 循环遍历 numArray 数组的长度，在表头中添加 "第 n 个数" 的列名，其中 n 表示对应的索引加 1。

在排序过程中，使用 for 循环遍历 resultList 列表中的每个数组 arr。创建一个 Object 数组 row，长度与当前数组 arr 相同，用于存储表格的一行数据。将当前数组的元素逐个赋值给 row 数组的对应位置，然后调用 model.addRow(row) 方法向表格添加一行数据，将 row 数组作为参数传递给该方法。

（3）数组变化图绘制

BarChart 类在 open 方法中，通过获取默认的 Display 对象，创建界面所在的显示设备，然后调用 createContents 方法创建界面内容。 在 createContents 方法中，创建一个 Shell 对象作为界面的容器和 FigureCanvas 对象，用于显示绘制的柱状图。然后创建一个 BarChartFigure 对象，传入柱状图需要的数据，并将 BarChartFigure 对象设置为 FigureCanvas 的内容，使其显示在界面上。

BarChartFigure 类的构造函数接受一个整型数组 numArray，该数组包含了每个柱子的高度数据。paintFigure 函数使用循环遍历 numArray 数组中的每个元素，代表每个柱子的高度。在循环中，计算柱子的高度为 numArray[i] * 10，然后创建一个矩形区域 rect，指定柱子的位置和大小。接着创建一个 RectangleFigure 对象 rectangleFigure，代表一个矩形图形，将矩形图形添加到当前 BarChartFigure 对象中。然后在每个柱子上添加标签，显示该柱子对应的数字大小，标签的位置

在柱子正上方。最后更新柱子的横坐标 x，以便绘制下一个柱子。

（4）对象持久化

Persistence 类分别对 InputStr、SortList 和 ResultArr 这三种对象进行持久化，这里以 InputStr 为例进行阐述。

类中的 writeToFileInputStr 方法用于将 ArrayList<InputStr>对象写入到文件中。首先，它创建一个 FileOutputStream 对象，并指定文件路径（"inputStr.txt"）用于将数据写入。然后，创建 ObjectOutputStream 对象，并将 ArrayList<InputStr>对象通 writeObject 方法写入到文件中。最后，关闭 ObjectOutputStream 和 FileOutputStream，完成数据的写入。

loadFromFileInputStr 方法用于从文件中读取数据，并返回一个 ArrayList<InputStr>对象。首先，它创建一个 FileInputStream 对象，并指定文件路径（"inputStr.txt"）用于从文件中读取数据。然后，创建一个 ObjectInputStream 对象，并使用 readObject 方法从文件中读取数据，并将其强制转换为 ArrayList<InputStr>对象。最后，关闭 ObjectInputStream 和 FileInputStream，并返回读取到的 ArrayList<InputStr>对象。

（5）多线程

当用户在控制台输入"aaa"时，创建一个新的线程 t，并将一个实现了 Runnable 接口的匿名类对象作为参数传递给 Thread 的构造函数。该匿名类实现了 run()方法，定义了线程的执行逻辑。在 run()方法中，创建一个新的 Main 对象，即创建一个新的界面。调用线程对象 t 的 start()方法，启动新线程。

3. 系统实施和部署

3.1 应用程序如何部署

将 sortVisualization.zip 解压后，用 idea 打开文件夹。首先在 Project Structure 中引入 swt.jar 和 draw2d.jar。如图 3.1 所示。

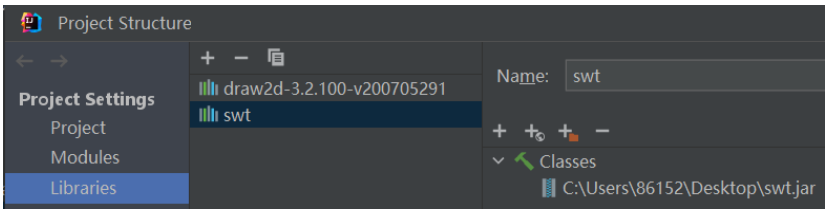
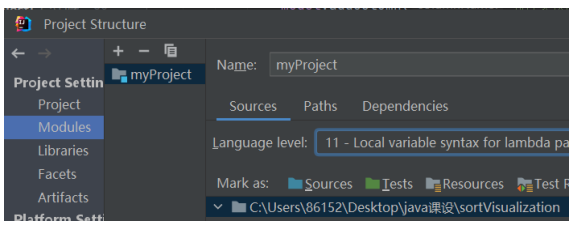
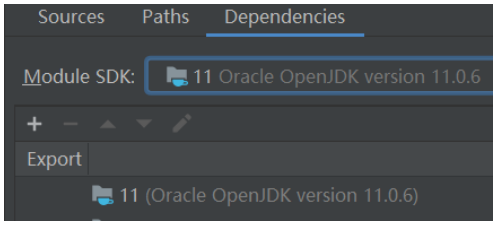


图 3.1 引入两个包

然后在在 Project Structure 的 Sources 和 Dependencies 中设置项目使用 JDK 11，并 Settings 中设置 JDK 11。如图 3.2 所示。





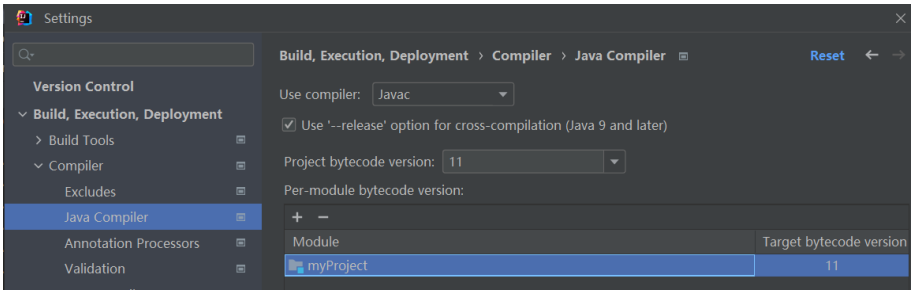


图 3.2 设置 JDK 版本为 11

3.2 应用程序如何运行

如图 3.3，找到 Main.java 文件，点击运行项目，则可进入系统。

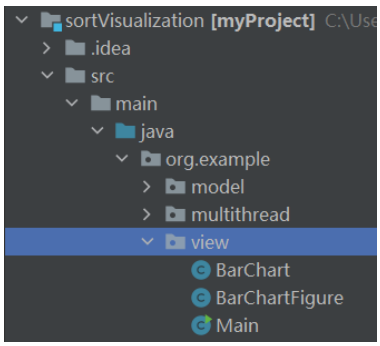


图 3.3 项目目录