深 圳 大 学 实 验 报 告

课程名称：**Java 程序设计**

实验项目名称：**课程实验 4：I/O、GUI 和网络编程**

学院：**计算机与软件学院**

专业：

指导教师：**潘微科**

报告人：　　学号：　　班级：

实验时间：**2024 年 11 月 29 日（周五）-2024 年 12 月 18 日（周三）**

实验报告提交时间：**2024 年 12 月 5 日星期四**

教务部制

实验目的与要求：

实验目的：掌握 I/O 程序设计，能够读写本地文件等，初步掌握图形界面程序设计，掌握网络通信协议及相关程序设计。

实验要求：

Part 1（25 分）

(1.1). 数据解析和统计。https://snap.stanford.edu/data/web-Amazon.html 网站上有很多 Amazon 的数据集供研究人员下载使用。本次实验使用 Watches.txt.gz 数据集，请下载后解压。格式说明请看网页上的“Data Format”部分。在报告中附上程序截图、运行结果（如每个输出文件前 10 行的截图等）和简要文字说明。

（i）使用 Java 语言读取解压后的文件（Watches.txt），并得到以下文件（10 分）：

review.txt: 每行 2 列，以分号作为分隔符，第 1 列是 userID，第 2 列是 productID，表示(user, product)二元组。该文件中不同行之间的顺序，按照 userID 从小到大排列，当 userID 相同时按照 productID 从小到大排列。

注：删除 ID 为 unknown 的记录；排序时按照字符串顺序。

（ii）使用 Java 语言根据 review.txt 进行计算，并得到以下文件（15 分）：

productNeighborhood.txt: 每行 4 列，以分号作为分隔符，第 1 列是 productID，第 2-4 列是与该 product 最相似（根据相似度值）的 3 个 product 的 productID，按相似度值从大到小排列（当相似度相同时，按照 productID 的字符串顺序从小到大排序），其中相似度是通过 review.txt 中的两列的信息计算得到的 Jaccard index 值。该文件中不同行之间的顺序，按照第 1 列的 productID 从小到大排列。关于相似度的计算，要求使用多线程实现（5 分）。

注：计算两个商品（product）之间的相似度（即 Jaccard index）时，根据这两个商品所关联的用户的集合来计算——集合的交集的大小除以集合的并集的大小。

Part 2（25 分）

(2.1).使用 JLabel、JTextArea、JButton 等控件实现句子中英互译的 demo，该 demo 包含两个文本框，第一个文本框用于输入中文句子或显示第二个文本框中的英文句子的中文翻译，第二个文本框用于输入英文句子或显示第一个文本框中的中文句子的英文翻译。每个文本框下方各有一个按钮，第一个按钮的名称是“中译英”，第二个按钮的名称是“英译中”，点击按钮表示将该文本框中的内容翻译成另一种语言。要求使用以下三种方式：

A、使用自己事先准备好的中英文翻译（此部分占 5 分）；

B、使用百度翻译 API、有道翻译 API 或其他 API 中的一个 API（此部分占 5 分）；

C、使用腾讯混元大模型、百度文心一言大模型 API、阿里通义千问大模型 API 或其他大模型 API 中的两个 API（此部分占 10 分）；

要求使用以下两个例句：

建校 41 年，深圳大学秉承“自立、自律、自强”的校训，紧随特区，锐意改革、快速发展，为特区发展和国家现代化建设做出了重要贡献。

Sticking to the motto of “self-reliance, self-discipline, self-improvement”, the University is dedicated to serving the Shenzhen Special Economic Zone (SEZ), demonstrating China’s reform and opening up and pioneering change in higher education.

要求使用图形用户界面，界面美观、交互友好。在报告中附上程序截图、运行结果和详细的文字说明。（5 分）

Part 3（30 分）

(3.1). 利用套接字连接（TCP）编写程序，该程序包括三个客户端（ClientA、ClientB、ClientC）和一个服务端（ServerS），三个客户端通过服务端作为桥梁实现相互间的文字交流，例如，ClientA 先发信息给 ServerS，然后 ServerC 再将收到的信息转发给 ClientB 和 ClientC。在报告中附上示意图（三个客户端+一个服务端）、程序截图、完整的运行结果和简要文字说明。（20 分）

(3.2). 利用数据报通信（UDP）实现题(1)中的要求。（10 分）

报告写作。要求：主要思路有明确的说明，重点代码有详细的注释，行文逻辑清晰可读性强，报告整体写作较为专业。（20 分）

说明：

（1）本次实验课作业满分为 100 分，占总成绩的比例 7%。

（2）本次实验课作业截至时间 2024 年 12 月 18 日（周三）21:59。

（3）报告正文：请在指定位置填写，本次实验不需要单独提交源程序文件。

（4）个人信息：WORD 文件名中的“姓名”、“学号”，请改为你的姓名和学号；实验报告的首页，请准确填写“学院”、“专业”、“报告人”、“学号”、“班级”、“实验报告提交时间”等信息。

（5）提交方式：截至时间前，请在 Blackboard 平台中提交。

（6）发现抄袭（包括复制&粘贴整句话、整张图），**抄袭者和被抄袭者的成绩记零分**。

（7）延迟提交，不得分；如有特殊情况，请于截至日期之后的 48 小时内发邮件到 panweike@szu.edu.cn，并在邮件中注明课程名称、作业名称、姓名、学号等信息，以及特殊情况的说明，我收到后会及时回复。

（8）期末考试阶段补交无效。

Part 1（25 分）

(1.1). 数据解析和统计。https://snap.stanford.edu/data/web-Amazon.html 网站上有很多 Amazon 的数据集供研究人员下载使用。本次实验使用 Watches.txt.gz 数据集，请下载后解压。格式说明请看网页上的“Data Format”部分。在报告中附上程序截图、运行结果（如每个输出文件前 10 行的截图等）和简要文字说明。

（i）使用 Java 语言读取解压后的文件（Watches.txt），并得到以下文件（10 分）：

review.txt: 每行 2 列，以分号作为分隔符，第 1 列是 userID，第 2 列是 productID，表示(user, product)二元组。该文件中不同行之间的顺序，按照 userID 从小到大排列，当 userID 相同时按照 productID 从小到大排列。

注：删除 ID 为 unknown 的记录；排序时按照字符串顺序。

（ii）使用 Java 语言根据 review.txt 进行计算，并得到以下文件（15 分）：

productNeighborhood.txt: 每行 4 列，以分号作为分隔符，第 1 列是 productID，第 2-4 列是与该 product 最相似（根据相似度值）的 3 个 product 的 productID，按相似度值从大到小排列（当相似度相同时，按照 productID 的字符串顺序从小到大排序），其中相似度是通过 review.txt 中的两列的信息计算得到的 Jaccard index 值。该文件中不同行之间的顺序，按照第 1 列的 productID 从小到大排列。关于相似度的计算，要求使用多线程实现（5 分）。

注：计算两个商品（product）之间的相似度（即 Jaccard index）时，根据这两个商品所关联的用户的集合来计算——集合的交集的大小除以集合的并集的大小。

（i）定义 main 方法，在其中指定了输入文件 Watches.txt 和输出文件 review.txt 的路径。然后通过循环读取 Watches.txt 文件的每一行，根据行的开头标识来提取 productId 和 userId 信息，并创建 ReviewRecord 对象来存储每条记录。同时删除 ID 为 unknown 的记录。之后按照用户 ID 从小到大排列，当用户 ID 相同时按照产品 ID 从小到大排列的规则对记录进行排序。最后将处理好的记录按照指定格式写入到 review.txt 文件中。

FileWrite.java × Watches.txt ×

```
1  package Part1;
2
3  import java.io.BufferedReader;
4  import java.io.BufferedWriter;
5  import java.io.FileReader;
6  import java.io.FileWriter;
7  import java.io.IOException;
8  import java.util.ArrayList;
9  import java.util.Collections;
10 import java.util.List;
11
12 public class FileWrite {
13
14     public static void main(String[] args) {
15         String watchesFilePath = "E:\\Programs\\JAVA\\Lab4\\src\\Part1\\Watches.txt";
16         String reviewFilePath = "E:\\Programs\\JAVA\\Lab4\\src\\Part1\\review.txt";
17
18         List<ReviewRecord> reviewRecords = new ArrayList<>();
```

```
        try (BufferedReader reader = new BufferedReader(new FileReader(watchesFilePath))) {
            String line;
            ReviewRecord currentRecord = null;
            while ((line = reader.readLine())!= null) {
                if (line.startsWith("product/productId:")) {
                    // 开始处理新的记录，先保存上一个记录（如果有）
                    if (currentRecord!= null) {
                        reviewRecords.add(currentRecord);
                    }
                    currentRecord = new ReviewRecord();
                    currentRecord.productId = line.split( regex: ": ")[1];
                } else if (line.startsWith("review/userId:")) {
                    currentRecord.userId = line.split( regex: ": ")[1];
                }
            }
            // 处理最后一个记录
            if (currentRecord!= null) {
                reviewRecords.add(currentRecord);
            }

            // 删除ID为unknown的记录
            reviewRecords.removeIf(record -> "unknown".equals(record.productId) || "unknown".equals(record.userId));

            // 按照要求排序
            Collections.sort(reviewRecords, (r1, r2) -> {
                int userIdComparison = r1.userId.compareTo(r2.userId);
                if (userIdComparison == 0) {
                    return r1.productId.compareTo(r2.productId);
                }
                return userIdComparison;
            });
```

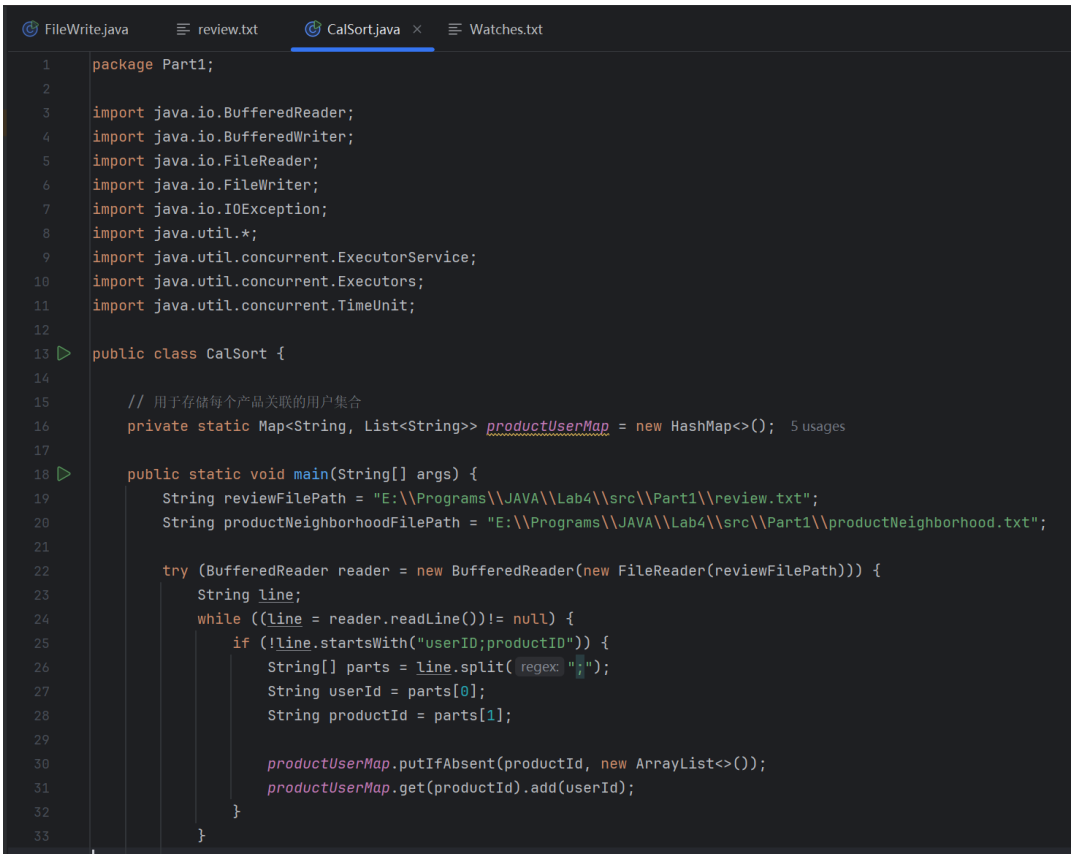
```
            // 写入review.txt文件
            try (BufferedWriter writer = new BufferedWriter(new FileWriter(reviewFilePath))) {
                writer.write( str: "userID;productID");
                writer.newLine();

                for (ReviewRecord record : reviewRecords) {
                    writer.write( str: record.userId + ";" + record.productId);
                    writer.newLine();
                }
            }

        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    // 内部类用于存储每条记录的信息
    private static class ReviewRecord {  4 usages
        String productId;  5 usages
        String userId;  5 usages
    }
}
```

review.txt 内容如下：

FileWrite.java | review.txt | Watches.txt

```
userID;productID
A00219682P7VSC0F8HLCU;B000GB1R7S
A0051720290P2FUYLQ0W3;B000UMK62U
A00859153MLXIP2YVLG0S;B000EQS1D8
A01069191W4W3PK41V3MB;B000AYW0LI
A01660142FKOE1FUIBVU6;B000IK5XTK
A0213583FZ5C5YLZVYII;B000HDJT6Q
A0301520N462Z31RLCLS;B000FVE3BQ
A03371112IP0NDM0XC0O3;B000AYW0M2
A0473911220TP4EMJ6STS;B0006AAS5Q
A048346320A5N6SY520LX;B000JQJS6M
A05590483FAOIB9CFW4A9;B0009352FG
A06330232S6LGGY5KFJDW;B000K3DTCS
A06337672HBV4IHKL6K7P;B0000C9ZD1
A06908902R3PJYHYBAT9;B000SQM5PE
A07312811YZ70HF45UQCI;B000B7MBHM
A0732117lOKZ0952Y225W;B0009P66ZO
A07470053U2K2OV48NFPX;B000JQJS6M
A07826899RN2VF148VJC;B0002948KC
A07918463QKLI6R1DTHXS;B000PSQXXU
A087519335RKPOT19N5ZB;B0000UZ498
A09398482EC892KI6UA5W;B0000JU7J6
A09469841VP90WO0GJMJK;B000GAYQLI
A09653182LQOTLH87Z6HW;B0001HIT1Y
A1000TZ38O9LNS;B000H6AQ0Q
A10014631T8TLLG6QFX0V;B000K3D982
A100355G5MKBEV;B000ESUVRA
A10063PJ5C9WQQ;B0000TII8M
A1006T3B0ZRG8C;B0000UZ4VQ
A1006VQPLKR7MX;B000H28H6U
A100DZOGKXD8J5;B0009P66TK
A100E1UHI8C5H3;B0002UD438
A100ED5J8HBCMS;B000JCN3T4
A100FOCUI19OBS;B000GB1RE6
A100HIQ1HUKJUI;B00095MKRW
```

（ii）首先读取 review.txt 文件，将每个产品关联的用户集合存储到 productUserMap 中。然后创建线程池，使用多线程计算每个产品与其他产品的相似度（Jaccard index），并将结果存储到 similarityResults 中。接着对每个产品的相似度结果进行排序，选取前 3 个最相似的产品 ID，并存储到 neighborhoodResults 中。之后按照第一列 productID 从小到大对 neighborhoodResults 进行排序。最后将处理好的结果按照要求的格式写入到 productNeighborhood.txt 文件中。



```java
        // 获取所有产品ID
        List<String> productIds = new ArrayList<>(productUserMap.keySet());

        // 创建线程池
        ExecutorService executorService = Executors.newFixedThreadPool(Runtime.getRuntime().availableProcessors());

        // 用于存储每个产品与其他产品的相似度结果
        Map<String, List<SimilarityResult>> similarityResults = new HashMap<>();

        // 计算每个产品与其他产品的相似度
        for (String productId : productIds) {
            executorService.submit(() -> {
                List<SimilarityResult> resultList = new ArrayList<>();
                for (String otherProductId : productIds) {
                    if (!productId.equals(otherProductId)) {
                        double similarity = calculateJaccardIndex(productUserMap.get(productId),
                                productUserMap.get(otherProductId));
                        resultList.add(new SimilarityResult(otherProductId, similarity));
                    }
                }
                similarityResults.put(productId, resultList);
            });
        }

        // 关闭线程池
        executorService.shutdown();
        try {
            executorService.awaitTermination(Long.MAX_VALUE, TimeUnit.NANOSECONDS);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
```

```java
        // 对每个产品的相似度结果进行排序，并选取前3个
        List<ProductNeighborhoodResult> neighborhoodResults = new ArrayList<>();
        for (Map.Entry<String, List<SimilarityResult>> entry : similarityResults.entrySet()) {
            String productId = entry.getKey();
            List<SimilarityResult> results = entry.getValue();
            results.sort((r1, r2) -> {
                if (r1.similarity == r2.similarity) {
                    return r1.productId.compareTo(r2.productId);
                }
                return Double.compare(r2.similarity, r1.similarity);
            });
            List<String> topThreeProductIds = new ArrayList<>();
            for (int i = 0; i < Math.min(3, results.size()); i++) {
                topThreeProductIds.add(results.get(i).productId);
            }
            neighborhoodResults.add(new ProductNeighborhoodResult(productId, topThreeProductIds));
        }

        // 按照第一列productID从小到大排序
        neighborhoodResults.sort((r1, r2) -> r1.productId.compareTo(r2.productId));

        // 写入productNeighborhood.txt文件
        try (BufferedWriter writer = new BufferedWriter(new FileWriter(productNeighborhoodFilePath))) {
            writer.write( str: "productID;similarProductID1;similarProductID2;similarProductID3");
            writer.newLine();
            for (ProductNeighborhoodResult result : neighborhoodResults) {
                writer.write( str: result.productId + ";" + result.similarProductIds.get(0) + ";"
                        + result.similarProductIds.get(1) + ";" + result.similarProductIds.get(2));
                writer.newLine();
            }
        }
```

```
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    // 计算Jaccard index
    private static double calculateJaccardIndex(List<String> set1, List<String> set2) {  1 usage
        if (set1 == null || set2 == null) {
            return 0;
        }
        List<String> intersection = new ArrayList<>(set1);
        intersection.retainAll(set2);
        List<String> union = new ArrayList<>(set1);
        union.addAll(set2);
        union = new ArrayList<>(new HashSet<>(union));
        return (double) intersection.size() / union.size();
    }

    // 用于存储相似度结果的内部类
    private static class SimilarityResult {  5 usages
        String productId;  4 usages
        double similarity;  5 usages

        public SimilarityResult(String productId, double similarity) {  1 usage
            this.productId = productId;
            this.similarity = similarity;
        }
    }
```

```
    // 用于存储产品邻域结果的内部类
    private static class ProductNeighborhoodResult {  3 usages
        String productId;  4 usages
        List<String> similarProductIds;  4 usages

        public ProductNeighborhoodResult(String productId, List<String> similarProductIds) {  1 usage
            this.productId = productId;
            this.similarProductIds = similarProductIds;
        }
    }
}
```

productNeighborhood.txt 文件如下：

FileWrite.java | review.txt | CalSort.java | Watches.txt | productNeighborhood.txt

```
productID;similarProductID1;similarProductID2;similarProductID3
4971850491;B000BM7KAU;B000EQVZOU;B000BQVU8Y
4971850788;B0007RTC50;0789208628;1861262337
4971850813;0789208628;1861262337;4971850491
7966780883;B000EQVZ86;B0007IR4U4;B000A7645I
B00004TF4K;0789208628;1861262337;4971850491
B000050T8S;0789208628;1861262337;4971850491
B000050T8V;0789208628;1861262337;4971850491
B000050T8W;0789208628;1861262337;4971850491
B000050T91;0789208628;1861262337;4971850491
B00005QEME;0789208628;1861262337;4971850491
B00005QY08;0789208628;1861262337;4971850491
B00005U7S9;0789208628;1861262337;4971850491
B0000643Q6;B0000C9ZBZ;B000JKNU94;B0000C9ZCX
B0000643Q8;B000JTHAFK;B000EMDAXS;B000MNCH9W
B0000643Q9;B0007KTGM6;B000ETU7MI;B000EU0KB0
B000068X4M;0789208628;1861262337;4971850491
B00006A7DQ;0789208628;1861262337;4971850491
B00006I5XQ;B000GGTWR0;B0000C2TOF;0789208628
B00006I5XR;B0000TIIHS;0789208628;1861262337
B00006I5XZ;B0009WXTES;0789208628;1861262337
B00006I5Y5;0789208628;1861262337;4971850491
B00006I5Y8;0789208628;1861262337;4971850491
B00006I5YC;0789208628;1861262337;4971850491
B00006I5YE;0789208628;1861262337;4971850491
B00006I5YG;B0000U0LRC;B000GB1RAA;B00093CZV0
B00006I5YJ;B0000TIJJ0;0789208628;1861262337
B00006I5YQ;B000B545DM;0789208628;1861262337
B00007E7KA;0789208628;1861262337;4971850491
```

Part 2（25 分）

(2.1).使用 JLabel、JTextArea、JButton 等控件实现句子中英互译的 demo，该 demo 包含两个文本框，第一个文本框用于输入中文句子或显示第二个文本框中的英文句子的中文翻译，第二个文本框用于输入英文句子或显示第一个文本框中的中文句子的英文翻译。每个文本框下方各有一个按钮，第一个按钮的名称是“中译英”，第二个按钮的名称是“英译中”，点击按钮表示将该文本框中的内容翻译成另一种语言。要求使用以下三种方式：

A、使用自己事先准备好的中英文翻译（此部分占 5 分）；

B、使用百度翻译 API、有道翻译 API 或其他 API 中的一个 API（此部分占 5 分）；

C、使用腾讯混元大模型、百度文心一言大模型 API、阿里通义千问大模型 API 或其他大模型 API 中的两个 API（此部分占 10 分）；

要求使用以下两个例句：

建校 41 年，深圳大学秉承“自立、自律、自强”的校训，紧随特区，锐意改革、快速发展，为特区发展和国家现代化建设做出了重要贡献。

Sticking to the motto of “self-reliance, self-discipline, self-improvement”, the University is dedicated to serving the Shenzhen Special Economic Zone (SEZ), demonstrating China’s reform and opening up and pioneering change in higher education.

要求使用图形用户界面，界面美观、交互友好。在报告中附上程序截图、运行结果和详细的文字说明。（5 分）

1. 导入需要使用的库和依赖

pom.xml (Translate) Translate.java ×

```
1  import javax.swing.*;
2  import java.awt.*;
3  import java.awt.event.ActionEvent;
4  import java.awt.event.ActionListener;
5  import com.baidu.translate.demo.TransApi;
6  import com.alibaba.fastjson2.JSON;
7  import java.util.Arrays;
8  import com.alibaba.dashscope.aigc.generation.Generation;
9  import com.alibaba.dashscope.aigc.generation.GenerationParam;
10 import com.alibaba.dashscope.aigc.generation.GenerationResult;
11 import com.alibaba.dashscope.common.Message;
12 import com.alibaba.dashscope.common.Role;
13 import com.alibaba.dashscope.exception.ApiException;
14 import com.alibaba.dashscope.exception.InputRequiredException;
15 import com.alibaba.dashscope.exception.NoApiKeyException;
16 import java.util.HashMap;
17 import java.util.Map;
18 import okhttp3.*;
19 import org.json.JSONObject;
20 import java.io.IOException;
21
```

pom.xml (Translate) × Translate.java

```xml
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <url>http://maven.apache.org</url>
  <dependencies>
    <dependency>
      <groupId>com.alibaba.fastjson2</groupId>
      <artifactId>fastjson2</artifactId>
      <version>2.0.53</version>
    </dependency>
    <dependency>
      <groupId>com.alibaba</groupId>
      <artifactId>dashscope-sdk-java</artifactId>
      <!-- 请将 'the-latest-version' 替换为查询到的最新版本号，https://mvnrepository.com/artifact/com.alibaba/das
      <version>2.16.9</version>
    </dependency>
    <dependency>
      <groupId>com.squareup.okhttp3</groupId>
      <artifactId>okhttp</artifactId>
      <version>4.12.0</version>
    </dependency>
    <dependency>
      <groupId>org.json</groupId>
      <artifactId>json</artifactId>
      <version>20240303</version>
```

2. 定义全局常量（密钥）和一个静态映射表，使模型名称与传入参数连接，初始化客户端。

```java
public class Translate extends JFrame {
    private JTextArea chineseTextArea, englishTextArea;  6 usages
    private JButton translateToEnglish, translateToChinese;  3 usages
    private JComboBox<String> translateMethodComboBox, modelComboBox;  5 usages
    private JLabel chineseLabel, englishLabel;  2 usages

    // 添加一个静态映射用于关联用户界面显示的模型名称和实际的API模型名称
    private static final Map<String, String> MODEL_NAME_MAP = new HashMap<>();  4 usages
    static {
        MODEL_NAME_MAP.put("通义千问", "qwen-plus");
        MODEL_NAME_MAP.put("百度文心一言", "baidu-wenxin"); // 添加百度文心一言模型映射
    }

    private static final String API_KEY = "5mGsGYlMSuSUL6wM1mxOP7sn";  1 usage
    private static final String SECRET_KEY = "8IaRCIfXMvvGZ66YetQpYkFE74WIC1lH";  1 usage

    static final OkHttpClient HTTP_CLIENT = new OkHttpClient().newBuilder().build();  2 usages
```

3. 设计用户使用窗口界面。

```java
    public Translate() {  3 usages
        // 初始化UI组件
        chineseLabel = new JLabel( text: "中文:");
        englishLabel = new JLabel( text: "英文:");

        // 创建文本区域并启用自动换行
        chineseTextArea = new JTextArea( rows: 10,  columns: 40);
        chineseTextArea.setLineWrap(true); // 启用行包裹
        chineseTextArea.setWrapStyleWord(true); // 启用单词包裹

        englishTextArea = new JTextArea( rows: 10,  columns: 40);
        englishTextArea.setLineWrap(true); // 启用行包裹
        englishTextArea.setWrapStyleWord(true); // 启用单词包裹

        // 创建翻译按钮
        translateToEnglish = new JButton( text: "中译英");
        translateToChinese = new JButton( text: "英译中");
```

```
        // 创建翻译方法选择框
        String[] methods = {"例句翻译", "翻译API", "大模型翻译"};
        translateMethodComboBox = new JComboBox<>(methods);

        // 创建大模型选择框，默认为空
        modelComboBox = new JComboBox<>();

        // 添加监听器
        translateToEnglish.addActionListener(new TranslateActionListener("zh", "en"));
        translateToChinese.addActionListener(new TranslateActionListener("en", "zh"));

        // 监听翻译方法变化以更新大模型选择框
        translateMethodComboBox.addActionListener(e -> updateModelOptions());

        // 设置初始大模型选项
        updateModelOptions();
```

```
        // 设置布局
        setLayout(new BorderLayout());

        // 创建面板以容纳文本框和选择框
        JPanel textPanel = new JPanel(new GridLayout( rows: 2,  cols: 2));
        textPanel.add(chineseLabel);
        textPanel.add(new JScrollPane(chineseTextArea));
        textPanel.add(englishLabel);
        textPanel.add(new JScrollPane(englishTextArea));

        JPanel buttonPanel = new JPanel();
        buttonPanel.add(translateMethodComboBox);
        buttonPanel.add(modelComboBox);
        buttonPanel.add(translateToEnglish);
        buttonPanel.add(translateToChinese);

        // 添加面板到主窗口
        add(textPanel, BorderLayout.CENTER);
        add(buttonPanel, BorderLayout.SOUTH);

        // 设置窗口属性
        setTitle("中英互译Demo");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize( width: 600,  height: 400);
        setLocationRelativeTo(null); // 居中显示窗口
        setVisible(true);
    }
```

```
    private void updateModelOptions() {  2 usages
        String selectedMethod = (String) translateMethodComboBox.getSelectedItem();
        modelComboBox.removeAllItems();

        if ("大模型翻译".equals(selectedMethod)) {
            // 使用映射中的键（用户友好的名称）填充下拉菜单
            for (String friendlyName : MODEL_NAME_MAP.keySet()) {
                modelComboBox.addItem(friendlyName);
            }
        }
    }
```

4. 定义监听器，处理按钮事件

```
    // 定义监听器类以处理翻译按钮点击事件
    private class TranslateActionListener implements ActionListener {  2 usages
        private final String fromLang;  7 usages
        private final String toLang;  5 usages

        public TranslateActionListener(String fromLang, String toLang) {  2 usages
            this.fromLang = fromLang;
            this.toLang = toLang;
        }

        @Override
        public void actionPerformed(ActionEvent e) {
            String method = (String) translateMethodComboBox.getSelectedItem();
            String friendlyModelName = (String) modelComboBox.getSelectedItem(); // 获取选定的友好模型名称
            JTextArea sourceTextArea = fromLang.equals("zh") ? chineseTextArea : englishTextArea;
            JTextArea targetTextArea = fromLang.equals("zh") ? englishTextArea : chineseTextArea;

            String sourceText = sourceTextArea.getText().trim();

            if (sourceText.isEmpty()) {
                JOptionPane.showMessageDialog( parentComponent: Translate.this,  message: "请输入要翻译的文本！");
                return;
            }
```

```
            String translatedText = null;
            switch (method) {
                case "例句翻译":
                    translatedText = translateStatic(sourceText, fromLang, toLang);
                    break;
                case "翻译API":
                    translatedText = translateUsingApi(sourceText, fromLang, toLang);
                    break;
                case "大模型翻译":
                    if (friendlyModelName == null || friendlyModelName.isEmpty()) {
                        JOptionPane.showMessageDialog( parentComponent: Translate.this,  message: "请选择一个大模型！");
                        return;
                    }
                    String apiModelName = MODEL_NAME_MAP.get(friendlyModelName);
                    if (apiModelName == null) {
                        JOptionPane.showMessageDialog( parentComponent: Translate.this,  message: "未知的大模型名称！");
                        return;
                    }
                    if ("baidu-wenxin".equals(apiModelName)) {
                        translatedText = translateUsingBaiduWenxin(sourceText, fromLang, toLang); // 调用百度文心一言API
                    } else {
                        translatedText = translateUsingModelApi(sourceText, fromLang, toLang, apiModelName); // 使用其他大模型API
                    }
                    break;
                default:
                    JOptionPane.showMessageDialog( parentComponent: Translate.this,  message: "未知的翻译方法！");
                    return;
            }
```

```

            if (translatedText != null) {
                targetTextArea.setText(translatedText);
            } else {
                JOptionPane.showMessageDialog( parentComponent: Translate.this,  message: "翻译失败，请尝试其他翻译方法！");
            }
        }

        private String translateStatic(String text, String fromLang, String toLang) {  1 usage
            // 根据源语言和目标语言调用静态翻译函数
            if ("zh".equals(fromLang)) {
                return translateStaticChineseToEnglish(text);
            } else if ("en".equals(fromLang)) {
                return translateStaticEnglishToChinese(text);
            }
            return null;
        }
    }
```

5. 分别实现不同的翻译方式。

```java
// A: 实现静态翻译函数
private String translateStaticChineseToEnglish(String text) {  1 usage
    if ("建校41年，深圳大学秉承“自立、自律、自强”的校训，紧随特区，锐意改革、快速发展，为特区发展和国家现代化建设做出了重要贡献。"
            .equals(text)) {
        return "Sticking to the motto of “self-reliance, self-discipline, self-improvement”, the University is " +
                "dedicated to serving the Shenzhen Special Economic Zone (SEZ), demonstrating China’s reform " +
                "and opening up and pioneering change in higher education.";
    }
    return null; // 如果没有匹配项则返回null
}

private String translateStaticEnglishToChinese(String text) {  1 usage
    if (("Sticking to the motto of “self-reliance, self-discipline, self-improvement”, the University is " +
            "dedicated to serving the Shenzhen Special Economic Zone (SEZ), demonstrating China’s reform " +
            "and opening up and pioneering change in higher education.").equals(text)) {
        return "建校41年，深圳大学秉承“自立、自律、自强”的校训，紧随特区，锐意改革、快速发展，为特区发展和国家现代化建设做出了重要贡献。";
    }
    return null; // 如果没有匹配项则返回null
}
```

```java
// B: 调用API进行翻译
private String translateUsingApi(String text, String fromLang, String toLang) {  1 usage
    TransApi api = new TransApi( appid: "20241205002220037",  securityKey: "98Q3xp0lAEjDib9e7vx7");
    String result = api.getTransResult(text, fromLang, toLang);
    com.alibaba.fastjson2.JSONObject jsonObject = JSON.parseObject(result);
    String translatedText = jsonObject.getJSONArray( key: "trans_result").getJSONObject( index: 0).getString( key: "dst");
    return translatedText;
}
```

```java
// C: 使用通义千问API进行翻译
private String translateUsingAliQwen(String text, String fromLang, String toLang, String model) {  1 usage
    try {
        Generation gen = new Generation();
        Message systemMsg = Message.builder()  MessageBuilder<capture of ?, capture of ?>
                .role(Role.SYSTEM.getValue())  capture of ?
                .content("You are a helpful assistant that translates text between languages.")
                .build();
        Message userMsg = Message.builder()  MessageBuilder<capture of ?, capture of ?>
                .role(Role.USER.getValue())  capture of ?
                .content(String.format("Translate this text from %s to %s: %s", fromLang, toLang, text))
                .build();

        GenerationParam param = GenerationParam.builder()  GenerationParamBuilder<capture of ?, capture of ?>
                .apiKey("sk-82b00fbaabd24f7893722ca8d7d9cc01") // 请替换为你的API Key
                .model(model) // 使用传入的模型名称
                .messages(Arrays.asList(systemMsg, userMsg))  capture of ?
                .resultFormat(GenerationParam.ResultFormat.MESSAGE)
                .build();

        GenerationResult result = gen.call(param);
        return result.getOutput().getChoices().get(0).getMessage().getContent();
    } catch (ApiException | NoApiKeyException | InputRequiredException e) {
        JOptionPane.showMessageDialog( parentComponent: this,  message: "翻译失败：" + e.getMessage());
        return null;
    }
}
```

```java
// 使用百度文心一言API进行翻译
private String translateUsingBaiduWenxin(String text, String fromLang, String toLang) {  1 usage
    try {
        // 准备请求体
        JSONObject requestBody = new JSONObject();
        requestBody.put("messages", Arrays.asList(
                new JSONObject().put("role", "user").put("content", String.format("Translate this text from %s to %s: %s", fro
        ));
        requestBody.put("temperature", 0.95);
        requestBody.put("top_p", 0.8);
        requestBody.put("penalty_score", 1);
        requestBody.put("enable_system_memory", false);
        requestBody.put("disable_search", false);
        requestBody.put("enable_citation", false);

        RequestBody body = RequestBody.create(MediaType.parse( $this$parse: "application/json"), requestBody.toString());
        Request request = new Request.Builder()
                .url("https://aip.baidubce.com/rpc/2.0/ai_custom/v1/wenxinworkshop/chat/completions_pro?access_token=" + getAc
                .method( method: "POST", body)
                .addHeader( name: "Content-Type",  value: "application/json")
                .build();
        Response response = HTTP_CLIENT.newCall(request).execute();

        // 解析响应体为 JSONObject 并提取 'result' 字段
        if (!response.isSuccessful()) throw new IOException("Unexpected code " + response);

        JSONObject jsonResponse = new JSONObject(response.body().string());
        return jsonResponse.getString( key: "result");

    } catch (IOException e) {
        JOptionPane.showMessageDialog( parentComponent: this,  message: "翻译失败：" + e.getMessage());
        return null;
    }
}
```

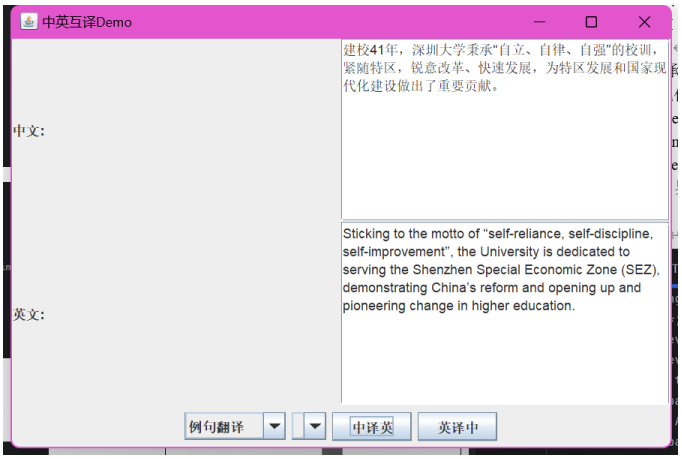
6. 处理返回并获得结果

```java
// 使用百度文心一言API进行翻译
private String translateUsingBaiduWenxin(String text, String fromLang, String toLang) {  1 usage
    try {
        // 准备请求体
        JSONObject requestBody = new JSONObject();
        requestBody.put("messages", Arrays.asList(
                new JSONObject().put("role", "user").put("content", String.format("Translate this text from %s to %s: %s", fro
        ));
        requestBody.put("temperature", 0.95);
        requestBody.put("top_p", 0.8);
        requestBody.put("penalty_score", 1);
        requestBody.put("enable_system_memory", false);
        requestBody.put("disable_search", false);
        requestBody.put("enable_citation", false);

        RequestBody body = RequestBody.create(MediaType.parse( $this$parse: "application/json"), requestBody.toString());
        Request request = new Request.Builder()
                .url("https://aip.baidubce.com/rpc/2.0/ai_custom/v1/wenxinworkshop/chat/completions_pro?access_token=" + getAcc
                .method( method: "POST", body)
                .addHeader( name: "Content-Type",  value: "application/json")
                .build();
        Response response = HTTP_CLIENT.newCall(request).execute();

        // 解析响应体为 JSONObject 并提取 'result' 字段
        if (!response.isSuccessful()) throw new IOException("Unexpected code " + response);

        JSONObject jsonResponse = new JSONObject(response.body().string());
        return jsonResponse.getString( key: "result");

    } catch (IOException e) {
        JOptionPane.showMessageDialog( parentComponent: this,  message: "翻译失败: " + e.getMessage());
        return null;
    }
}
```

7.启动服务。

效果如下：

A. 例句翻译



B.翻译 API

C.大模型翻译


中英互译Demo
中文:
建校41年，深圳大学秉承“自立、自律、自强”的校训，紧随特区，锐意改革、快速发展，为特区发展和国家现代化建设做出了重要贡献。
英文:
For 41 years, Shenzhen University has upheld the motto of "Self-reliance, Self-discipline, and Self-improvement," closely following the special economic zone's lead, boldly reforming, and rapidly developing. It has made significant contributions to the development of the special economic zone and the modernization of the country.
大模型翻译
通义千问
中译英
英译中


中英互译Demo
中文:
建校41年，深圳大学秉承“自立、自律、自强”的校训，紧随特区，锐意改革、快速发展，为特区发展和国家现代化建设做出了重要贡献。
英文:
Over the past 41 years since its establishment, Shenzhen University has been adhering to the motto of “self-reliance, self-discipline and self-improvement”, keeping up with the development of the Special Economic Zone, carrying out determined reforms, developing rapidly, and making important contributions to the development of the Special Economic Zone and the modernization of the country.
大模型翻译
百度文心一言
中译英
英译中

Part 3（30 分）

(3.1). 利用套接字连接（TCP）编写程序，该程序包括三个客户端（ClientA、ClientB、ClientC）和一个服务端（ServerS），三个客户端通过服务端作为桥梁实现相互间的文字交流，例如，ClientA 先发信息给 ServerS，然后 ServerC 再将收到的信息转发给 ClientB 和 ClientC。在报告中附上示意图（三个客户端+一个服务端）、程序截图、完整的运行结果和简要文字说明。（20 分）

1. ServerS 服务器监听端口，等待客户端连接每当有一个新的客户端连接时，服务器创建一个新的线程来处理这个用户的通信。ClientHandler 线程负责读取客户端发送的消息，并将消息广播给所有已连接的客户端，如果客户端发送了 "exit" 命令，或者发生了异常，服务器会关闭与该客户端的连接并移除其输出流。

Client.java | ClientC.java | ClientA.java | ChatRoomLauncher.java | ClientB.java | ServerS.java

```java
package Part3;
import java.io.*;
import java.net.*;
import java.util.*;

public class ServerS {
    private static final int PORT = 12345;  2 usages
    private static Set<PrintWriter> clientWriters = Collections.synchronizedSet(new HashSet<>());  6 usages

    public static void main(String[] args) throws IOException {
        try (ServerSocket serverSocket = new ServerSocket(PORT)) {
            System.out.println("Server is listening on port " + PORT);
            while (true) {
                Socket socket = serverSocket.accept();
                System.out.println("New client connected");
                new ClientHandler(socket).start();
            }
        } catch (IOException e) {
            System.err.println("Server exception: " + e.getMessage());
            e.printStackTrace();
        }
    }
```

```java
    private static class ClientHandler extends Thread {  1 usage
        private Socket socket;  5 usages
        private PrintWriter out;  6 usages
        private BufferedReader in;  4 usages

        public ClientHandler(Socket socket) {  1 usage
            this.socket = socket;
        }

        @Override
        public void run() {
            try {
                in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
                out = new PrintWriter(socket.getOutputStream(), autoFlush: true);

                synchronized (clientWriters) {
                    clientWriters.add(out);
                }

                String message;
                while ((message = in.readLine()) != null) {
                    if ("exit".equalsIgnoreCase(message.trim())) break;
                    broadcastMessage(message);
                }

                closeConnection();
            } catch (IOException e) {
                System.err.println("Exception from chat client: " + e.getMessage());
            } finally {
                closeConnection();
            }
        }
```

```
        private void broadcastMessage(String message) {  1 usage
            synchronized (clientWriters) {
                for (PrintWriter writer : clientWriters) {
                    writer.println(message);
                }
            }
        }

        private void closeConnection() {  2 usages
            if (out != null) {
                synchronized (clientWriters) {
                    clientWriters.remove(out);
                }
            }
            try {
                if (in != null) in.close();
                if (out != null) out.close();
                if (socket != null) socket.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
}
```

2.Client 类，客户端提供 startClient()方法，用于建立与服务器的连接，并开启一个新线程来持续接收服务器发来的消息，主线程则负责从标准输入读取消息，并将其发送给服务器。

Client.java × ClientC.java ClientA.java ChatRoomLauncher.java ClientB.java

```
package Part3;

import java.io.*;
import java.net.*;
import java.util.Scanner;

public abstract class Client {  4 usages  3 related problems
    protected static final String SERVER_HOST = "localhost";  3 usages
    protected static final int SERVER_PORT = 12345;  1 usage
    protected String nickname;  3 usages

    public Client(String nickname) {  no usages  3 related problems
        this.nickname = nickname;
    }
```

```java
public void startClient() {  1 usage  3 related problems
    Scanner scanner = new Scanner(System.in);
    try (Socket socket = new Socket(SERVER_HOST, SERVER_PORT);
         PrintWriter out = new PrintWriter(socket.getOutputStream(), autoFlush: true);
         BufferedReader in = new BufferedReader(new InputStreamReader(socket.getInputStream()))) {

        // Start a thread to read messages from the server
        Thread receiveThread = new Thread(() -> {
            try {
                String serverMessage;
                while ((serverMessage = in.readLine()) != null) {
                    System.out.println(serverMessage);
                }
            } catch (IOException e) {
                System.err.println("Error receiving message from server.");
            }
        });
        receiveThread.start();

        // Send user input to the server
        System.out.println(nickname + " has joined the chat!");
        while (true) {
            String userInput = scanner.nextLine();
            if ("exit".equalsIgnoreCase(userInput.trim())) break;
            out.println("[" + nickname + "] " + userInput);
        }

    } catch (UnknownHostException e) {
        System.err.println("Don't know about host " + SERVER_HOST);
    } catch (IOException e) {
        System.err.println("Couldn't get I/O for the connection to " + SERVER_HOST);
    } finally {
        scanner.close();
    }
}
```

3.具体客户端,ClientA、B、C 继承自 Client 类,每个客户端类都有自己的 main 方法，用于实例化自己并调用 startClient() 方法，从而启动客户端。

Client.java ClientC.java ClientA.java × ChatRoomLauncher.java ClientB.java ServerS

```java
package Part3;


// ClientA.java
public class ClientA extends Client {
    public ClientA() {  2 usages
        super( nickname: "ClientA");
    }

    public static void main(String[] args) {
        new ClientA().startClient();
    }
}
```

Client.java ClientC.java ClientA.java ChatRoomLauncher.java ClientB.java × ServerS

```java
package Part3;

// ClientB.java
public class ClientB extends Client {
    public ClientB() {  1 usage
        super( nickname: "ClientB");
    }

    public static void main(String[] args) {
        new ClientB().startClient();
    }
}
```

Client.java | ClientC.java | ClientA.java | ChatRoomLauncher.java | ClientB.java | ServerS.ja

```java
package Part3;

// ClientC.java
public class ClientC extends Client {
    public ClientC() {  1 usage
        super( nickname: "ClientC");
    }

    public static void main(String[] args) {
        new ClientC().startClient();
    }
}
```

4.创建一个客户端通用的启动器，用于创建相应的客户端实例并启动它。

Client.java | ClientC.java | ClientA.java | ChatRoomLauncher.java | ClientB.java | ServerS.java

```java
package Part3;

// ChatRoomLauncher.java
public class ChatRoomLauncher {
    public static void main(String[] args) {
        if (args.length < 2) {
            System.out.println("Usage: java ChatRoomLauncher <clientType> <nickname>");
            return;
        }

        String clientType = args[0];
        String nickname = args[1];

        Client client;
        switch (clientType.toLowerCase()) {
            case "a":
                client = new ClientA();
                break;
            case "b":
                client = new ClientB();
                break;
            case "c":
                client = new ClientC();
                break;
            default:
                System.out.println("Unknown client type: " + clientType);
                return;
        }

        client.startClient();
    }
}
```

使用效果如下：

服务器

```
PS E:\Programs\JAVA\Lab4\src> java Part3.ServerS
Server is listening on port 12345
New client connected
New client connected
New client connected
```

客户端 A

```
PS E:\Programs\JAVA\Lab4\src> java Part3.ClientA
ClientA has joined the chat!
Hello,I am A
[ClientA] Hello,I am A
[ClientB] I am B
[ClientC] Hi,I am C
```

客户端 B

```
PS E:\Programs\JAVA\Lab4\src> java Part3.ClientB
ClientB has joined the chat!
[ClientA] Hello,I am A
I am B
[ClientB] I am B
[ClientC] Hi,I am C
```

客户端 C

```
PS E:\Programs\JAVA\Lab4\src> java Part3.ClientC
ClientC has joined the chat!
[ClientA] Hello,I am A
[ClientB] I am B
Hi,I am C
[ClientC] Hi,I am C
```

(3.2). 利用数据报通信（UDP）实现题(1)中的要求。（10 分）

1.服务器 UDPServer

ava ClientC.java ClientA.java ChatRoomLauncher.java UDPServer.java × UDPClient.java UDPClientA.java

```java
package UDP;
import java.io.IOException;
import java.net.*;
import java.util.*;

public class UDPServer {
    private static final int PORT = 12345;  2 usages
    private static Set<String> clientAddresses = Collections.synchronizedSet(new HashSet<>());  4 usages
    private static DatagramSocket socket;  6 usages

    public static void main(String[] args) {
        try {
            // Initialize the DatagramSocket as a static member
            socket = new DatagramSocket(PORT);
            System.out.println("Server is listening on port " + PORT);

            byte[] receiveBuffer = new byte[1024];
            while (true) {
                DatagramPacket receivePacket = new DatagramPacket(receiveBuffer, receiveBuffer.length);
                socket.receive(receivePacket);

                String message = new String(receivePacket.getData(),  offset: 0, receivePacket.getLength()).trim();

                InetAddress clientAddress = receivePacket.getAddress();
                int clientPort = receivePacket.getPort();

                // Add the sender's address to the set of known clients
                synchronized (clientAddresses) {
                    clientAddresses.add(clientAddress.getHostAddress() + ":" + clientPort);
                }

                System.out.println("Received: " + message + " from " + clientAddress.getHostAddress() + ":" + clientPort);

                // Broadcast the message to all known clients except the sender
                broadcastMessage(message, clientAddress, clientPort);
            }
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            if (socket != null && !socket.isClosed()) {
                socket.close();
            }
        }
    }

    private static void broadcastMessage(String message, InetAddress senderAddress, int senderPort) throws IOException {  1 usa
        synchronized (clientAddresses) {
            for (String addressPort : clientAddresses) {
                String[] parts = addressPort.split( regex: ":");
                InetAddress address = InetAddress.getByName(parts[0]);
                int port = Integer.parseInt(parts[1]);

                if (!address.equals(senderAddress) || port != senderPort) {
                    byte[] sendBuffer = message.getBytes();
                    DatagramPacket sendPacket = new DatagramPacket(sendBuffer, sendBuffer.length, address, port);
                    socket.send(sendPacket);
                }
            }
        }
    }
}
```

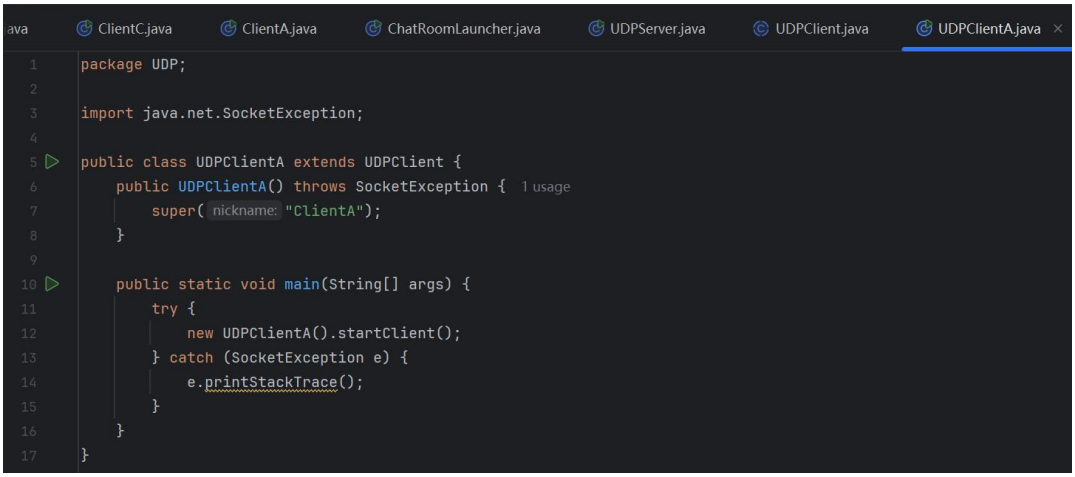
2.UDPClient 类

ClientC.java | ClientA.java | ChatRoomLauncher.java | UDPServer.java | UDPClient.java

```java
package UDP;


import java.io.*;
import java.net.*;
import java.util.Scanner;

public abstract class UDPClient {  3 usages  3 inheritors
    protected static final String SERVER_HOST = "localhost";  1 usage
    protected static final int SERVER_PORT = 12345;  1 usage
    protected String nickname;  3 usages
    protected DatagramSocket socket;  4 usages

    public UDPClient(String nickname) throws SocketException {  3 usages
        this.nickname = nickname;
        this.socket = new DatagramSocket();
    }

    public void startClient() {  3 usages  3 related problems
        Scanner scanner = new Scanner(System.in);
        Thread receiveThread = new Thread(this::receiveMessages);
        receiveThread.start();

        System.out.println(nickname + " has joined the chat!");

        while (true) {
            String userInput = scanner.nextLine();
            if ("exit".equalsIgnoreCase(userInput.trim())) break;
            sendMessage("[" + nickname + "] " + userInput);
        }

        socket.close();
        scanner.close();
    }

```

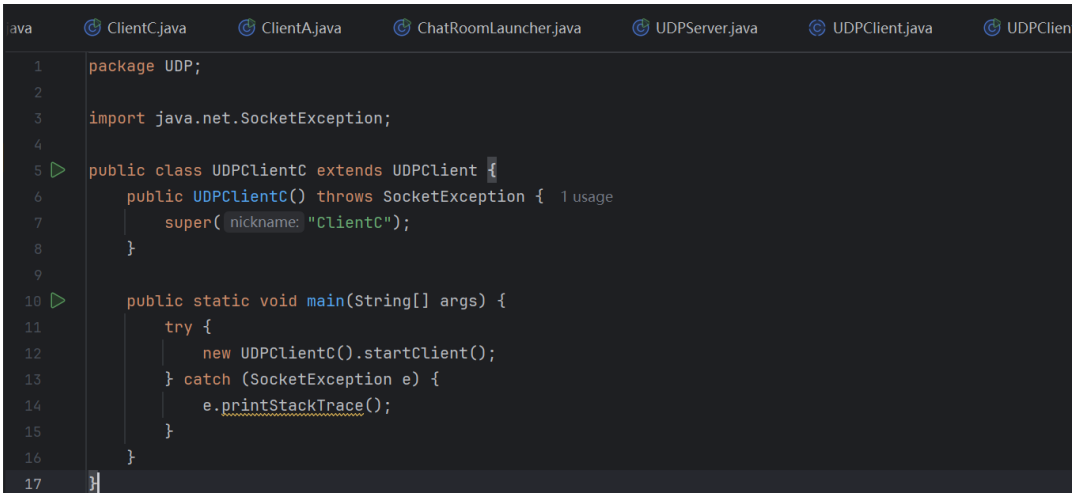
```java
    private void sendMessage(String message) {  1 usage
        try {
            byte[] buffer = message.getBytes();
            DatagramPacket packet = new DatagramPacket(buffer, buffer.length, InetAddress.getByName(SERVER_HOST), SERVER_PORT);
            socket.send(packet);
        } catch (IOException e) {
            System.err.println("Error sending message.");
        }
    }

    private void receiveMessages() {  1 usage
        byte[] receiveBuffer = new byte[1024];
        DatagramPacket receivePacket = new DatagramPacket(receiveBuffer, receiveBuffer.length);

        while (true) {
            try {
                socket.receive(receivePacket);
                String message = new String(receivePacket.getData(),  offset: 0, receivePacket.getLength());
                System.out.println(message);

                // Reset the length of the packet after receiving
                receivePacket.setLength(receiveBuffer.length);
            } catch (IOException e) {
                System.err.println("Error receiving message.");
            }
        }
    }
```

3.客户端 UDPClientA、B、C



```java
package UDP;

import java.net.SocketException;

public class UDPClientA extends UDPClient {
    public UDPClientA() throws SocketException {  1 usage
        super( nickname: "ClientA");
    }

    public static void main(String[] args) {
        try {
            new UDPClientA().startClient();
        } catch (SocketException e) {
            e.printStackTrace();
        }
    }
}
```

```java
package UDP;

import java.net.SocketException;

public class UDPClientB extends UDPClient {
    public UDPClientB() throws SocketException {  1 usage
        super( nickname: "ClientB");
    }

    public static void main(String[] args) {
        try {
            new UDPClientB().startClient();
        } catch (SocketException e) {
            e.printStackTrace();
        }
    }
}
```



```java
package UDP;

import java.net.SocketException;

public class UDPClientC extends UDPClient {
    public UDPClientC() throws SocketException {  1 usage
        super( nickname: "ClientC");
    }

    public static void main(String[] args) {
        try {
            new UDPClientC().startClient();
        } catch (SocketException e) {
            e.printStackTrace();
        }
    }
}
```

使用效果如下：

服务器

```
PS E:\Programs\JAVA\Lab4> cd src
PS E:\Programs\JAVA\Lab4\src> java UDP.UDPServer
Server is listening on port 12345
Received: [ClientA] I am A from 127.0.0.1:63194
Received: [ClientB] Hello,I am B from 127.0.0.1:63195
Received: [ClientC] Here is C from 127.0.0.1:63197
```

客户端 A

```
PS E:\Programs\JAVA\Lab4> cd src
PS E:\Programs\JAVA\Lab4\src> java UDP.UDPClientA
ClientA has joined the chat!
I am A
[ClientB] Hello,I am B
[ClientC] Here is C
```

客户端 B

```
PS E:\Programs\JAVA\Lab4> cd src
PS E:\Programs\JAVA\Lab4\src> java UDP.UDPClientB
ClientB has joined the chat!
Hello,I am B
[ClientC] Here is C
```

客户端 C

```
PS E:\Programs\JAVA\Lab4> cd src
PS E:\Programs\JAVA\Lab4\src> java UDP.UDPClientC
ClientC has joined the chat!
Here is C
```

++

其他（例如感想、建议等等）。

在做 API 调用的时候要引入大量依赖库，接触学习了不同的依赖引入方法和 Maven

指导教师批阅意见：   成绩评定：   指导教师签字： 2024 年　　月　　日
备注：

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。