**实验四**

**1.题目：**

|  |
| --- |
| 1、改写程序清单10-6的Course类：  改写getStudents()方法，返回一个数组，该数组长度和课程中的选课学生数一样（提示：创建一个新的数组并将学生拷贝进去）  改进addStudent()方法，在数组没有更多空间添加更多学生时，可以自动增加数组大小。可以通过创建一个新的更大的数组并复制当前数组的内容来实现。  实现dropStudent().  添加一个clear()，用于清除选某门课程的所有学生。  编写一个测试程序，创建一门课程，添加三个学生，删除一个学生，然后显示该课程的选课学生。 |

**完整代码：**

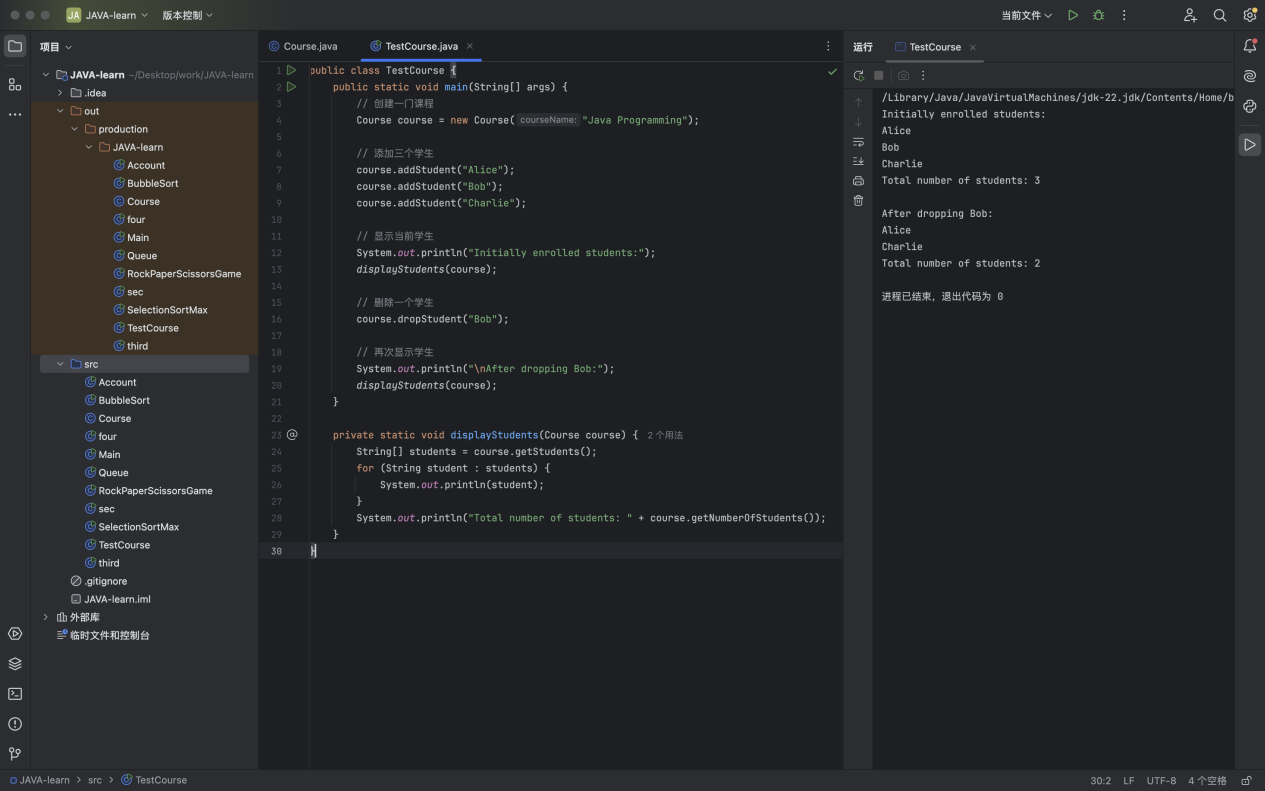
Course：

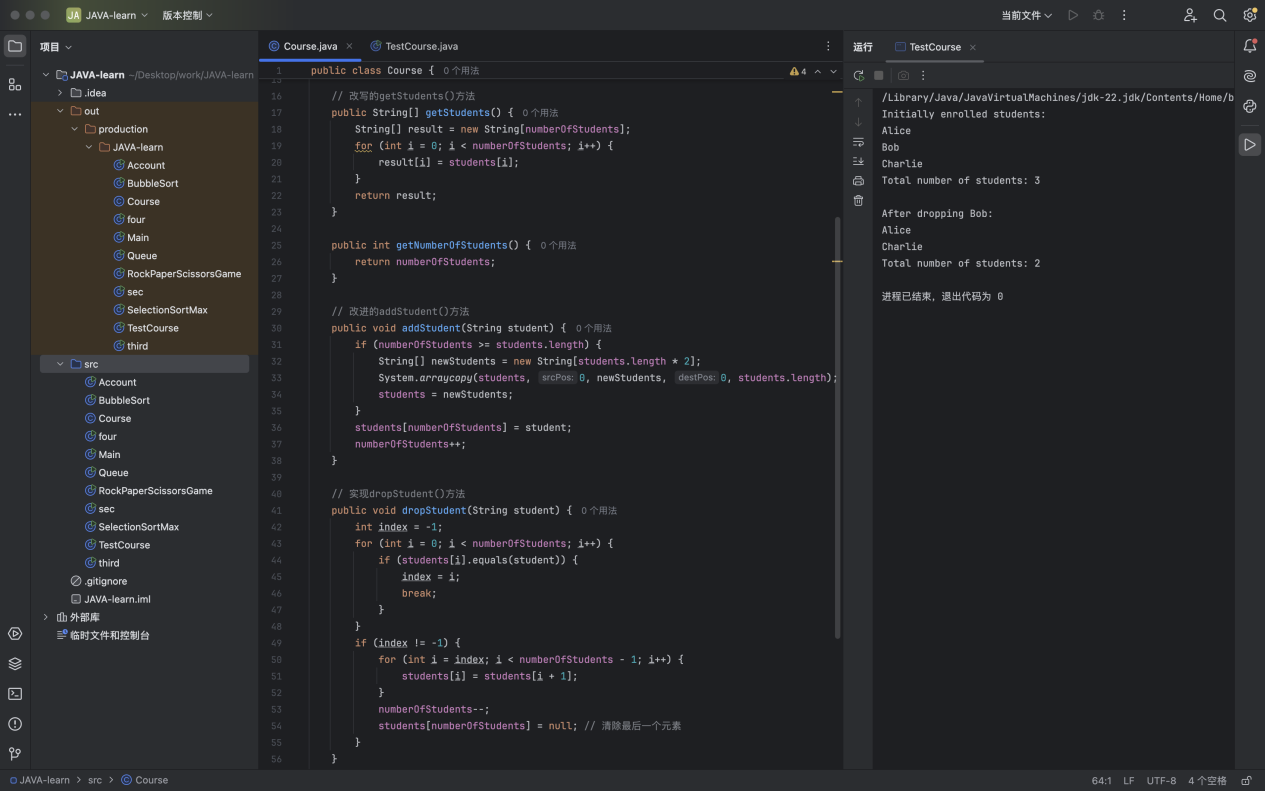
|  |
| --- |
| Java public class Course {  private String courseName;  private String[] students;  private int numberOfStudents;   public Course(String courseName) {  this.courseName = courseName;  this.students = new String[10]; // 初始容量为10  this.numberOfStudents = 0;  }   public String getCourseName() {  return courseName;  }   // 改写的getStudents()方法  public String[] getStudents() {  String[] result = new String[numberOfStudents];  for (int i = 0; i < numberOfStudents; i++) {  result[i] = students[i];  }  return result;  }   public int getNumberOfStudents() {  return numberOfStudents;  }   // 改进的addStudent()方法  public void addStudent(String student) {  if (numberOfStudents >= students.length) {  String[] newStudents = new String[students.length \* 2];  System.*arraycopy*(students, 0, newStudents, 0, students.length);  students = newStudents;  }  students[numberOfStudents] = student;  numberOfStudents++;  }   // 实现dropStudent()方法  public void dropStudent(String student) {  int index = -1;  for (int i = 0; i < numberOfStudents; i++) {  if (students[i].equals(student)) {  index = i;  break;  }  }  if (index != -1) {  for (int i = index; i < numberOfStudents - 1; i++) {  students[i] = students[i + 1];  }  numberOfStudents--;  students[numberOfStudents] = null; // 清除最后一个元素  }  }   // 实现clear()方法  public void clear() {  students = new String[10]; // 重置为初始容量  numberOfStudents = 0;  } } |

Test：

|  |
| --- |
| Java public class TestCourse {  public static void main(String[] args) {  // 创建一门课程  Course course = new Course("Java Programming");   // 添加三个学生  course.addStudent("Alice");  course.addStudent("Bob");  course.addStudent("Charlie");   // 显示当前学生  System.*out*.println("Initially enrolled students:");  *displayStudents*(course);   // 删除一个学生  course.dropStudent("Bob");   // 再次显示学生  System.*out*.println("\nAfter dropping Bob:");  *displayStudents*(course);  }   private static void displayStudents(Course course) {  String[] students = course.getStudents();  for (String student : students) {  System.*out*.println(student);  }  System.*out*.println("Total number of students: " + course.getNumberOfStudents());  } } |

**代码运行截图：**





**2.题目：**

|  |
| --- |
| 2、石头剪刀布游戏：  计算机和玩家各出一个，判断输赢，支持多次连续操作，直到玩家选择退出。 |

**完整代码：**

|  |
| --- |
| JavaScript import java.util.Random; import java.util.Scanner;  public class RockPaperScissorsGame {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.*in*);  Random random = new Random();   int playerWins = 0;  int computerWins = 0;  int ties = 0;   System.*out*.println("欢迎来到石头剪刀布游戏!");  System.*out*.println("规则: 0 = 石头, 1 = 剪刀, 2 = 布");   while (true) {  // 获取玩家的选择  System.*out*.print("请输入你的选择 (0-2), 或输入-1退出游戏: ");  int playerChoice = scanner.nextInt();   // 检查是否退出游戏  if (playerChoice == -1) {  break;  }   // 验证玩家输入  if (playerChoice < 0 || playerChoice > 2) {  System.*out*.println("无效输入,请重试!");  continue;  }   // 生成计算机的选择  int computerChoice = random.nextInt(3);   // 显示双方的选择  System.*out*.println("你的选择: " + *choiceToString*(playerChoice));  System.*out*.println("计算机的选择: " + *choiceToString*(computerChoice));   // 判断胜负  int result = (playerChoice - computerChoice + 3) % 3;   switch (result) {  case 0:  System.*out*.println("平局!");  ties++;  break;  case 1:  System.*out*.println("你赢了!");  playerWins++;  break;  case 2:  System.*out*.println("计算机赢了!");  computerWins++;  break;  }   // 显示当前比分  System.*out*.println("当前比分 - 你: " + playerWins + ", 计算机: " + computerWins + ", 平局: " + ties);  System.*out*.println();  }   // 显示最终比分  System.*out*.println("\n游戏结束!");  System.*out*.println("最终比分 - 你: " + playerWins + ", 计算机: " + computerWins + ", 平局: " + ties);   scanner.close();  }   // 将数字选择转换为对应的字符串  private static String choiceToString(int choice) {  switch (choice) {  case 0:  return "石头";  case 1:  return "剪刀";  case 2:  return "布";  default:  return "未知";  }  } } |

**代码运行截图：**

