**随堂测验**

**1.题目：**

|  |
| --- |
| 一、利用Java.util.Comparator<E>对银行定期储户进行降序排列并输出，要求程序使用继承和多态的原理。 |

**完整代码：**

BankAccount.java:

|  |
| --- |
| JavaScript import java.util.ArrayList; import java.util.Collections; import java.util.Comparator; import java.util.List;  // 创建抽象的银行账户类作为父类 abstract class BankAccount {  protected String accountNumber;  protected String accountName;  protected double balance;   public BankAccount(String accountNumber, String accountName, double balance) {  this.accountNumber = accountNumber;  this.accountName = accountName;  this.balance = balance;  }   public abstract void displayInfo(); }  // 定期储户类，继承自银行账户类 class FixedDepositAccount extends BankAccount {  private int depositPeriod; // 存款期限（月）  private double interestRate; // 利率   public FixedDepositAccount(String accountNumber, String accountName, double balance,  int depositPeriod, double interestRate) {  super(accountNumber, accountName, balance);  this.depositPeriod = depositPeriod;  this.interestRate = interestRate;  }   @Override  public void displayInfo() {  System.*out*.printf("账号: %s, 姓名: %s, 余额: %.2f, 存期: %d个月, 利率: %.2f%%\n",  accountNumber, accountName, balance, depositPeriod, interestRate \* 100);  }   // getter方法  public double getBalance() {  return balance;  } }  // 创建比较器类实现Comparator接口 class BalanceComparator implements Comparator<FixedDepositAccount> {  @Override  public int compare(FixedDepositAccount account1, FixedDepositAccount account2) {  // 降序排列，所以用account2减account1  return Double.*compare*(account2.getBalance(), account1.getBalance());  } } |

BankAccountSort.java:

|  |
| --- |
| JavaScript import java.util.ArrayList; import java.util.Collections; import java.util.List;  // 主类 public class BankAccountSort {  public static void main(String[] args) {  // 创建储户列表  List<FixedDepositAccount> accounts = new ArrayList<>();   // 添加一些测试数据  accounts.add(new FixedDepositAccount("001", "张三", 50000.0, 12, 0.03));  accounts.add(new FixedDepositAccount("002", "李四", 100000.0, 24, 0.035));  accounts.add(new FixedDepositAccount("003", "王五", 75000.0, 36, 0.04));  accounts.add(new FixedDepositAccount("004", "赵六", 150000.0, 12, 0.03));  accounts.add(new FixedDepositAccount("005", "钱七", 200000.0, 24, 0.035));   // 输出排序前的列表  System.*out*.println("排序前的储户列表：");  for (FixedDepositAccount account : accounts) {  account.displayInfo();  }   // 使用比较器进行排序  Collections.*sort*(accounts, new BalanceComparator());   // 输出排序后的列表  System.*out*.println("\n按余额降序排序后的储户列表：");  for (FixedDepositAccount account : accounts) {  account.displayInfo();  }  } } |

**2.题目：**

|  |
| --- |
| 二、完成函数pickUp ( )的定义，使得main得以正常执行  import java.util.ArrayList;  import java.util.Random;  public class ListTest {  public static void main(String[] args) {  Random r = new Random();//创建生成随机数  //创建大集合  ArrayList<Integer> blist = new ArrayList<>();  for (int i = 0; i < 20; i++) {  int num = r.nextInt(100) + 1;  blist.add(num); //将随机数添加到大集合  }  System.out.println(blist);  //打印从大集合筛选出的偶数  System.out.println(pickUp(blist));  } |

**完整代码：**

|  |
| --- |
| JavaScript import java.util.ArrayList; import java.util.Random;  public class ListTest {  public static void main(String[] args) {  Random r = new Random(); // 创建生成随机数  // 创建大集合  ArrayList<Integer> blist = new ArrayList<>();  for (int i = 0; i < 20; i++) {  int num = r.nextInt(100) + 1;  blist.add(num); // 将随机数添加到大集合  }  System.*out*.println(blist);  // 打印从大集合筛选出的偶数  System.*out*.println(*pickUp*(blist));  }   // 定义pickUp方法，用于筛选偶数  public static ArrayList<Integer> pickUp(ArrayList<Integer> list) {  // 创建新集合，用于存储偶数  ArrayList<Integer> evenList = new ArrayList<>();   // 遍历原集合，筛选偶数  for (Integer num : list) {  if (num % 2 == 0) { // 判断是否为偶数  evenList.add(num);  }  }   return evenList;  } } |

**3.题目：**

|  |
| --- |
| 编写一个程序，创建一个 HashMap对象，用于实现用户信息的管理(包含用户名和密码)：注册、登录、修改密码等。如果用户名是admin，则可以遍历打印所有用户的信息. |

**完整代码：**

|  |
| --- |
| JavaScript import java.util.HashMap; import java.util.Map; import java.util.Scanner;  public class UserManagementSystem {  private static HashMap<String, String> *userMap* = new HashMap<>();  private static Scanner *scanner* = new Scanner(System.*in*);   public static void main(String[] args) {  // 预设admin账户  *userMap*.put("admin", "admin123");   while (true) {  System.*out*.println("\n=== 用户管理系统 ===");  System.*out*.println("1. 用户注册");  System.*out*.println("2. 用户登录");  System.*out*.println("3. 修改密码");  System.*out*.println("4. 退出系统");  System.*out*.print("请选择操作 (1-4): ");   int choice = *scanner*.nextInt();  *scanner*.nextLine(); // 清除换行符   switch (choice) {  case 1:  *register*();  break;  case 2:  *login*();  break;  case 3:  *changePassword*();  break;  case 4:  System.*out*.println("感谢使用，再见！");  return;  default:  System.*out*.println("无效选择，请重试！");  }  }  }   private static void register() {  System.*out*.print("请输入用户名: ");  String username = *scanner*.nextLine();   if (*userMap*.containsKey(username)) {  System.*out*.println("用户名已存在！");  return;  }   System.*out*.print("请输入密码: ");  String password = *scanner*.nextLine();   *userMap*.put(username, password);  System.*out*.println("注册成功！");  }   private static void login() {  System.*out*.print("请输入用户名: ");  String username = *scanner*.nextLine();  System.*out*.print("请输入密码: ");  String password = *scanner*.nextLine();   if (*userMap*.containsKey(username) && *userMap*.get(username).equals(password)) {  System.*out*.println("登录成功！");   // 如果是admin用户，显示所有用户信息  if (username.equals("admin")) {  System.*out*.println("\n=== 所有用户信息 ===");  for (Map.Entry<String, String> entry : *userMap*.entrySet()) {  System.*out*.println("用户名: " + entry.getKey() + ", 密码: " + entry.getValue());  }  }  } else {  System.*out*.println("用户名或密码错误！");  }  }   private static void changePassword() {  System.*out*.print("请输入用户名: ");  String username = *scanner*.nextLine();  System.*out*.print("请输入原密码: ");  String oldPassword = *scanner*.nextLine();   if (*userMap*.containsKey(username) && *userMap*.get(username).equals(oldPassword)) {  System.*out*.print("请输入新密码: ");  String newPassword = *scanner*.nextLine();  *userMap*.put(username, newPassword);  System.*out*.println("密码修改成功！");  } else {  System.*out*.println("用户名或密码错误！");  }  } } |

**4.题目：**

|  |
| --- |
| 用户输入一个网址，从中读取网络文本数据到程序，每当识别到一个子链接时，就将该链接地址存入一个列表，最终遍历列表，打印该网页所有子链接网址。 |

**完整代码：**

|  |
| --- |
| JavaScript import java.io.BufferedReader; import java.io.InputStreamReader; import java.net.URL; import java.util.ArrayList; import java.util.List; import java.util.Scanner; import java.util.regex.Matcher; import java.util.regex.Pattern;  public class WebLinkExtractor {  public static void main(String[] args) {  try {  Scanner scanner = new Scanner(System.*in*);  System.*out*.print("请输入网址 (例如: https://www.example.com): ");  String websiteUrl = scanner.nextLine();   // 获取网页内容  String content = *getWebContent*(websiteUrl);   // 提取链接  List<String> links = *extractLinks*(content);   // 打印所有链接  System.*out*.println("\n找到的链接：");  for (String link : links) {  System.*out*.println(link);  }  System.*out*.println("\n总共找到 " + links.size() + " 个链接");   } catch (Exception e) {  System.*out*.println("发生错误：" + e.getMessage());  }  }   private static String getWebContent(String websiteUrl) throws Exception {  StringBuilder content = new StringBuilder();  URL url = new URL(websiteUrl);  BufferedReader reader = new BufferedReader(  new InputStreamReader(url.openStream())  );   String line;  while ((line = reader.readLine()) != null) {  content.append(line).append("\n");  }  reader.close();  return content.toString();  }   private static List<String> extractLinks(String content) {  List<String> links = new ArrayList<>();   // 正则表达式匹配href属性  String regex = "href=[\"'](http[s]?://[^\"']+)[\"']";  Pattern pattern = Pattern.*compile*(regex);  Matcher matcher = pattern.matcher(content);   while (matcher.find()) {  String link = matcher.group(1);  if (!links.contains(link)) {  links.add(link);  }  }   return links;  } } |