Java基础练习3

1. 建立一个实体类Student类，属性：姓名，年龄，成绩，班级

建立一个list1，包含 “张三，18岁，80分，1班”，“李四，19岁，100分，1班”，“王五，17岁，59分，1班”。

建立一个list2，包含 “赵六，18岁，85分，2班”，“刘七，19岁，93分，2班”，“孙八，17岁，55分，2班”。

1. 整合两个list学生信息成一个新的list
2. 按照分数给出学生信息排名
3. 输出不及格的学生信息
4. 查找张三的信息
5. 从list剔除年龄大于18岁的学生信息
6. 使用Map 完成练习1的习题。

3.仿照手机淘宝，设计订单和商品的实体类。

1. Student类：

|  |
| --- |
| package com.neuteah.homework3;  /\*\*  \* 学生实体类  \*/  public class Student {  /\*\*  \* 学生姓名  \*/  private String name;  /\*\*  \* 学生年龄  \*/  private int age;  /\*\*  \* 学生成绩  \*/  private float score;  /\*\*  \* 学生班级  \*/  private String clazz;  public Student() {  }  public Student(String name, int age, float score, String clazz) {  this.name = name;  this.age = age;  this.score = score;  this.clazz = clazz;  }  @Override  public String toString() {  return "Student{" +  "name='" + name + '\'' +  ", age=" + age +  ", score=" + score +  ", clazz='" + clazz + '\'' +  '}';  }  public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  public int getAge() {  return age;  }  public void setAge(int age) {  this.age = age;  }  public float getScore() {  return score;  }  public void setScore(float score) {  this.score = score;  }  public String getClazz() {  return clazz;  }  public void setClazz(String clazz) {  this.clazz = clazz;  }  } |

（1）整合两个list学生信息成一个新的list

（2）按照分数给出学生信息排名

（3）输出不及格的学生信息

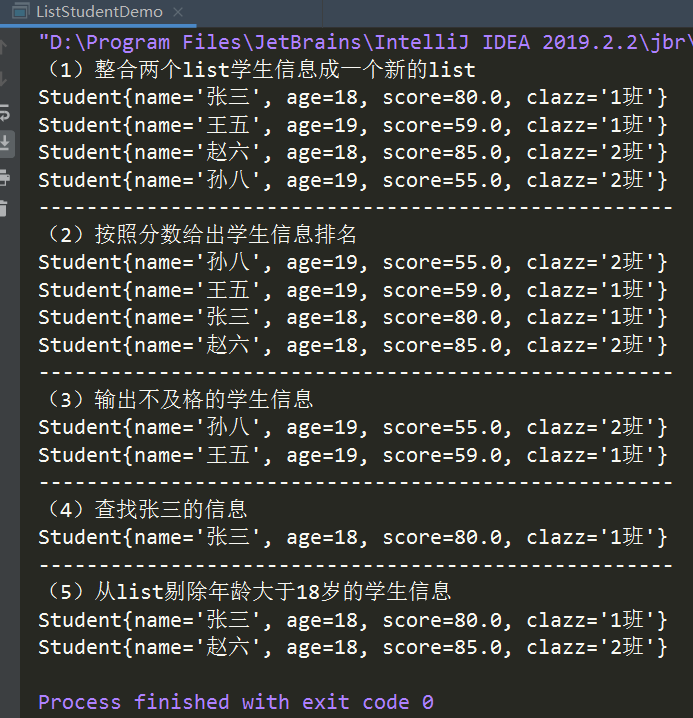
（4）查找张三的信息

（5）从list剔除年龄大于18岁的学生信息

代码整合如下：

|  |
| --- |
| package com.neuteah.homework3;  import java.util.ArrayList;  import java.util.Collections;  import java.util.Comparator;  import java.util.List;  /\*\*  \* 使用List集合操作Student对象  \*/  public class ListStudentDemo {  public static void main(String[] args) {  List<Student> list1 = new ArrayList<>();  List<Student> list2 = new ArrayList<>();  //添加Student对象  list1.add(new Student("张三", 18, 80, "1班"));  list1.add(new Student("王五", 19, 59, "1班"));  list2.add(new Student("赵六", 18, 85, "2班"));  list2.add(new Student("孙八", 19, 55, "2班"));  //（1）整合两个list学生信息成一个新的list  List<Student> listAll = functhon1(list1, list2);  //（2）按照分数给出学生信息排名  functhon2(listAll);  //（3）输出不及格的学生信息  functhon3(listAll);  //（4）查找张三的信息  functhon4(listAll);  //（5）从list剔除年龄大于18岁的学生信息  functhon5(listAll);  }  /\*\*  \*（1）整合两个list学生信息成一个新的list  \* @param list1 学生信息列表1  \* @param list2 学生信息列表2  \* @return 返回新的list  \*/  public static List<Student> functhon1(List<Student> list1, List<Student> list2){  System.out.println("（1）整合两个list学生信息成一个新的list");  List<Student> listAll = new ArrayList<>();  listAll.addAll(list1);  listAll.addAll(list2);  for (Student student : listAll) {  System.out.println(student.toString());  }  System.out.println("-----------------------------------------------------");  return listAll;  }  /\*\*  \*（2）按照分数给出学生信息排名  \* @param listAll 总的学生信息列表  \*/  public static void functhon2(List<Student> listAll){  System.out.println("（2）按照分数给出学生信息排名");  Collections.sort(listAll, new Comparator<Student>() {  @Override  public int compare(Student student1, Student student2) {  float score1 = student1.getScore();  float score2 = student2.getScore();  if (score1 > score2) {  return 1;  } else if (score1 == score2) {  return 0;  } else {  return -1;  }  }  });  for (Student student : listAll) {  System.out.println(student.toString());  }  System.out.println("-----------------------------------------------------");  }  /\*\*  \*（3）输出不及格的学生信息  \* @param listAll 总的学生信息列表  \*/  public static void functhon3(List<Student> listAll){  System.out.println("（3）输出不及格的学生信息");  for (Student student : listAll) {  if (student != null && student.getScore() < 60) {  System.out.println(student.toString());  }  }  System.out.println("-----------------------------------------------------");  }  /\*\*  \*（4）查找张三的信息  \* @param listAll 总的学生信息列表  \*/  public static void functhon4(List<Student> listAll){  System.out.println("（4）查找张三的信息");  for (Student student : listAll) {  if ("张三".equals(student.getName())) {  System.out.println(student.toString());  }  }  System.out.println("-----------------------------------------------------");  }  /\*\*  \*（5）从list剔除年龄大于18岁的学生信息  \* @param listAll 总的学生信息列表  \*/  public static void functhon5(List<Student> listAll){  System.out.println("（5）从list剔除年龄大于18岁的学生信息");  List<Student> removeList = new ArrayList<>();  for (Student student : listAll) {  if (student != null && student.getAge() > 18) {  removeList.add(student);  }  }  listAll.removeAll(removeList);  for (Student student : listAll) {  System.out.println(student.toString());  }  }  } |

截图：



2．Student类在1.中

（1）整合两个list学生信息成一个新的list

（2）按照分数给出学生信息排名

（3）输出不及格的学生信息

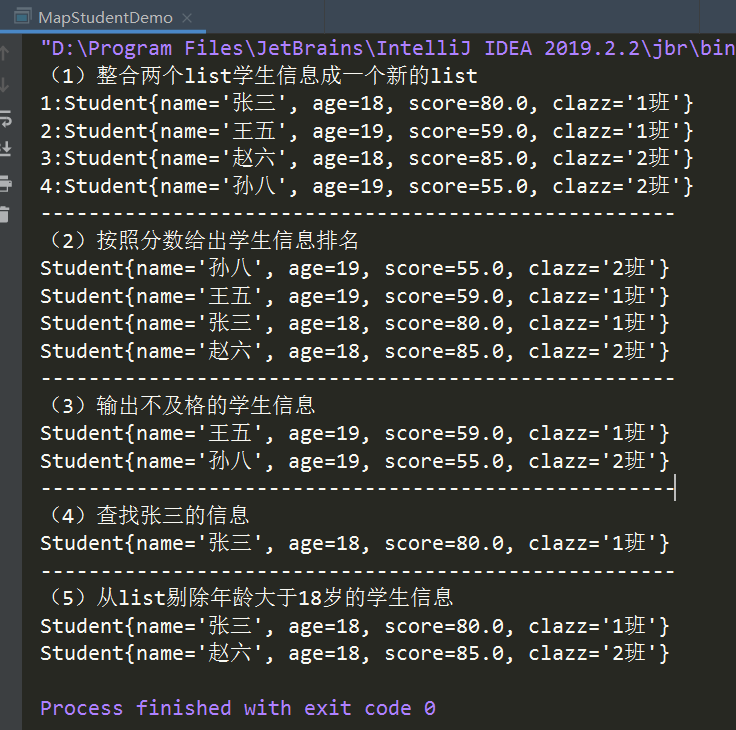
（4）查找张三的信息

（5）从list剔除年龄大于18岁的学生信息

代码整合如下：

|  |
| --- |
| package com.neuteah.homework3;  import java.util.\*;  /\*\*  \* 使用Map集合操作Student对象  \*/  public class MapStudentDemo {  public static void main(String[] args) {  Map<Integer, Student> map1 = new HashMap<>();  Map<Integer, Student> map2 = new HashMap<>();  //添加Student对象  map1.put(1, new Student("张三", 18, 80, "1班"));  map1.put(2, new Student("王五", 19, 59, "1班"));  map2.put(3, new Student("赵六", 18, 85, "2班"));  map2.put(4, new Student("孙八", 19, 55, "2班"));  //（1）整合两个list学生信息成一个新的list  Map<Integer, Student> mapAll = function1(map1, map2);  //（2）按照分数给出学生信息排名  functhon2(mapAll);  //（3）输出不及格的学生信息  functhon3(mapAll);  //（4）查找张三的信息  functhon4(mapAll);  //（5）从list剔除年龄大于18岁的学生信息  functhon5(mapAll);  }  /\*\*  \* （1）整合两个list学生信息成一个新的list  \*  \* @param map1 学生信息列表1  \* @param map2 学生信息列表2  \* @return  \*/  public static Map<Integer, Student> function1(Map<Integer, Student> map1, Map<Integer, Student> map2) {  System.out.println("（1）整合两个list学生信息成一个新的list");  Map<Integer, Student> mapAll = new HashMap<>();  mapAll.putAll(map1);  mapAll.putAll(map2);  for (Map.Entry<Integer, Student> entry : mapAll.entrySet()) {  System.out.println(entry.getKey() + ":" + entry.getValue().toString());  }  System.out.println("-----------------------------------------------------");  return mapAll;  }  /\*\*  \* （2）按照分数给出学生信息排名  \*  \* @param mapAll 总的学生信息集合  \*/  public static void functhon2(Map<Integer, Student> mapAll) {  System.out.println("（2）按照分数给出学生信息排名");  List<Map.Entry<Integer, Student>> entryList = new ArrayList<Map.Entry<Integer, Student>>(mapAll.entrySet());  Collections.sort(entryList, new Comparator<Map.Entry<Integer, Student>>() {  @Override  public int compare(Map.Entry<Integer, Student> integerStudentEntry1, Map.Entry<Integer, Student> integerStudentEntry2) {  float score1 = integerStudentEntry1.getValue().getScore();  float score2 = integerStudentEntry2.getValue().getScore();  if (score1 > score2) {  return 1;  } else if (score1 == score2) {  return 0;  } else {  return -1;  }  }  });  for (Map.Entry<Integer, Student> entry : entryList) {  System.out.println(entry.getValue().toString());  }  System.out.println("-----------------------------------------------------");  }  /\*\*  \* （3）输出不及格的学生信息  \*  \* @param mapAll 总的学生信息集合  \*/  public static void functhon3(Map<Integer, Student> mapAll) {  System.out.println("（3）输出不及格的学生信息");  for (Map.Entry<Integer, Student> entry : mapAll.entrySet()) {  if (entry != null && entry.getValue().getScore() < 60) {  System.out.println(entry.getValue().toString());  }  }  System.out.println("-----------------------------------------------------");  }  /\*\*  \* （4）查找张三的信息  \*  \* @param mapAll 总的学生信息集合  \*/  public static void functhon4(Map<Integer, Student> mapAll) {  System.out.println("（4）查找张三的信息");  for (Map.Entry<Integer, Student> entry : mapAll.entrySet()) {  if (entry != null && "张三".equals(entry.getValue().getName())) {  System.out.println(entry.getValue().toString());  }  }  System.out.println("-----------------------------------------------------");  }  /\*\*  \* （5）从list剔除年龄大于18岁的学生信息  \*  \* @param mapAll 总的学生信息集合  \*/  public static void functhon5(Map<Integer, Student> mapAll) {  System.out.println("（5）从list剔除年龄大于18岁的学生信息");  //把map中的Student对象转换成Iterator迭代器进行迭代  Iterator<Student> iterator = mapAll.values().iterator();  //当迭代器中还有元素时，继续循环遍历  while (iterator.hasNext()) {  //迭代器中的元素都是Student对象  Student student = iterator.next();  if (student != null && student.getAge() > 18) {  iterator.remove();  }  }  for (Map.Entry<Integer, Student> entry : mapAll.entrySet()) {  System.out.println(entry.getValue().toString());  }  }  } |

截图：



3.

·订单类

|  |
| --- |
| package com.neuteah.homework3;  import java.util.Date;  import java.util.List;  /\*\*  \* 淘宝订单实体类  \*/  public class Order {  /\*\*  \* 订单编号  \*/  private String id;  /\*\*  \* 支付宝交易号  \*/  private String aliPayNum;  /\*\*  \* 订单里的商品编号  \* 一个订单里可以有多个商品编号  \*/  private List<String> goodsId;  /\*\*  \* 创建订单的用户  \*/  private String userId;  /\*\*  \* 订单总价  \*/  private float orderPrice;  /\*\*  \* 商品总价  \*/  private float goodsPrice;  /\*\*  \* 实际付款价格  \*/  private float realPrice;  /\*\*  \* 邮费  \*/  private float expressPrice;  /\*\*  \* 收件人地址  \*/  private String address;  /\*\*  \* 创建时间  \*/  private Date createTime;  /\*\*  \* 付款时间  \*/  private Date payTime;  /\*\*  \* 获得的天猫积分  \*/  private int integral;  /\*\*  \* 发票  \*/  private String invoice;  public Order() {  }  public Order(String id, String aliPayNum, List<String> goodsId, String userId, float orderPrice, float goodsPrice, float realPrice, float expressPrice, String address, Date createTime, Date payTime, int integral, String invoice) {  this.id = id;  this.aliPayNum = aliPayNum;  this.goodsId = goodsId;  this.userId = userId;  this.orderPrice = orderPrice;  this.goodsPrice = goodsPrice;  this.realPrice = realPrice;  this.expressPrice = expressPrice;  this.address = address;  this.createTime = createTime;  this.payTime = payTime;  this.integral = integral;  this.invoice = invoice;  }  public String getId() {  return id;  }  public void setId(String id) {  this.id = id;  }  public String getAliPayNum() {  return aliPayNum;  }  public void setAliPayNum(String aliPayNum) {  this.aliPayNum = aliPayNum;  }  public List<String> getGoodsId() {  return goodsId;  }  public void setGoodsId(List<String> goodsId) {  this.goodsId = goodsId;  }  public String getUserId() {  return userId;  }  public void setUserId(String userId) {  this.userId = userId;  }  public float getOrderPrice() {  return orderPrice;  }  public void setOrderPrice(float orderPrice) {  this.orderPrice = orderPrice;  }  public float getGoodsPrice() {  return goodsPrice;  }  public void setGoodsPrice(float goodsPrice) {  this.goodsPrice = goodsPrice;  }  public float getRealPrice() {  return realPrice;  }  public void setRealPrice(float realPrice) {  this.realPrice = realPrice;  }  public float getExpressPrice() {  return expressPrice;  }  public void setExpressPrice(float expressPrice) {  this.expressPrice = expressPrice;  }  public String getAddress() {  return address;  }  public void setAddress(String address) {  this.address = address;  }  public Date getCreateTime() {  return createTime;  }  public void setCreateTime(Date createTime) {  this.createTime = createTime;  }  public Date getPayTime() {  return payTime;  }  public void setPayTime(Date payTime) {  this.payTime = payTime;  }  public int getIntegral() {  return integral;  }  public void setIntegral(int integral) {  this.integral = integral;  }  public String getInvoice() {  return invoice;  }  public void setInvoice(String invoice) {  this.invoice = invoice;  }  } |

商品类：

|  |
| --- |
| package com.neuteah.homework3;  import java.util.Date;  import java.util.List;  /\*\*  \* 淘宝商品实体类  \*/  public class Goods {  /\*\*  \* 商品编号  \*/  private String goodsId;  /\*\*  \* 商品名称  \*/  private String goodsName;  /\*\*  \* 促销价格  \*/  private float salePrice;  /\*\*  \* 原价  \*/  private float originalPrice;  /\*\*  \* 评价  \* 一件商品可以有很多评价  \*/  private List<String> goodsComments;  /\*\*  \* 图片保存路径  \* 一件商品可以有多张图片  \*/  private List<String> goodsImgUrl;  /\*\*  \* 商品数量  \*/  private int goodsSum;  /\*\*  \* 已销售数量  \*/  private int saleSum;  /\*\*  \* 上架时间  \*/  private Date releaseTime;  /\*\*  \* 下架时间  \*/  private Date withdrawalTime;  public Goods() {  }  public Goods(String goodsId, String goodsName, float salePrice, float originalPrice, List<String> goodsComments, List<String> goodsImgUrl, int goodsSum, int saleSum, Date releaseTime, Date withdrawalTime) {  this.goodsId = goodsId;  this.goodsName = goodsName;  this.salePrice = salePrice;  this.originalPrice = originalPrice;  this.goodsComments = goodsComments;  this.goodsImgUrl = goodsImgUrl;  this.goodsSum = goodsSum;  this.saleSum = saleSum;  this.releaseTime = releaseTime;  this.withdrawalTime = withdrawalTime;  }  @Override  public String toString() {  return "Goods{" +  "goodsId='" + goodsId + '\'' +  ", goodsName='" + goodsName + '\'' +  ", salePrice=" + salePrice +  ", originalPrice=" + originalPrice +  ", goodsComments=" + goodsComments +  ", goodsImgUrl=" + goodsImgUrl +  ", goodsSum=" + goodsSum +  ", saleSum=" + saleSum +  ", releaseTime=" + releaseTime +  ", withdrawalTime=" + withdrawalTime +  '}';  }  public String getGoodsName() {  return goodsName;  }  public void setGoodsName(String goodsName) {  this.goodsName = goodsName;  }  public float getSalePrice() {  return salePrice;  }  public void setSalePrice(float salePrice) {  this.salePrice = salePrice;  }  public float getOriginalPrice() {  return originalPrice;  }  public void setOriginalPrice(float originalPrice) {  this.originalPrice = originalPrice;  }  public List<String> getGoodsComments() {  return goodsComments;  }  public void setGoodsComments(List<String> goodsComments) {  this.goodsComments = goodsComments;  }  public List<String> getGoodsImgUrl() {  return goodsImgUrl;  }  public void setGoodsImgUrl(List<String> goodsImgUrl) {  this.goodsImgUrl = goodsImgUrl;  }  public int getGoodsSum() {  return goodsSum;  }  public void setGoodsSum(int goodsSum) {  this.goodsSum = goodsSum;  }  public int getSaleSum() {  return saleSum;  }  public void setSaleSum(int saleSum) {  this.saleSum = saleSum;  }  public Date getReleaseTime() {  return releaseTime;  }  public void setReleaseTime(Date releaseTime) {  this.releaseTime = releaseTime;  }  public Date getWithdrawalTime() {  return withdrawalTime;  }  public void setWithdrawalTime(Date withdrawalTime) {  this.withdrawalTime = withdrawalTime;  }  public String getGoodsId() {  return goodsId;  }  public void setGoodsId(String goodsId) {  this.goodsId = goodsId;  }  } |