中北大学软件学院

**实验报告**

|  |  |
| --- | --- |
| 专 业： | 软件工程 |
| 课程名称： | 面向对象程序设计 |
| 班 级： | 22130403 |
| 学 号： | 2213040340 |
| 姓 名： | 张恒瑜 |
| 辅导教师： | 李华玲 |

2021年09月制

成绩：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 实验时间 | 2023年 10 月22 日 19 时至 21时 | 学时数 | 4学时 | |
| 1.实验名称  抽象类与接口 | | | | |
| 2.实验目的  （1）熟练掌握抽象类及接口的定义及使用；  （2）掌握多态的含义，学会在实际应用中使用多态；  （3）理解接口回调的含义，学会在实际应用中使用。 | | | | |
| 3.训练知识点集群  （1）抽象类（定义，使用，方法重写，多态的实现等）  （2）接口（定义，使用，方法重写，多态的实现，接口回调） | | | | |
| 4.实验内容  **内容1：利用抽象类和多态的知识完成宠物商店宠物的“自我介绍”程序。**  要求：设计一个模拟发声器，可以根据输入的类型输出对应的宠物的信息。  输出格式为：我是一只狗。我叫乐乐，是一只黑色的牧羊犬。我会汪汪汪…。  我是一只猫。我叫咪咪，是一只白色的波斯猫。我会喵喵喵…。  **内容2：首先为线性表定义接口及抽象类，然后使用数组实现线性表。**  （1）定义一个接口MyList，里面包含插入、查找、删除、替换、获取线性表长度、判断线性表是否为空、清空线性表等抽象方法；  （2）定义一个抽象类MyAbstractList实现接口MyList，该抽象类拥有唯一的属性size（protected修饰），表示当前线性表的长度，并提供无参和有参构造方法完成线性表的初始化，重写接口中添加、删除、判断线性表是否为空及返回线性表长度的方法；  （3）利用数组实现线性表：定义一个MyArrayList类，继承自抽象类MyAbstractList，并利用数组实现所有操作。  （4）给出测试类  **内容3：**完善电商购物平台项目，在已有类（商品类目类、商品类和商品数据库类）的基础上，增加商品信息查询接口BookDao类和接口的实现类BookDaoaImpl，完成商品信息录入，修改，查询等功能。  **内容4：**（学生自由创造）  根据你理解的接口的含义，自己写一个有关接口的案例，并测试运行。请在下面描述你的案例： | | | | |
| 5. 根据问题抽象出类图   |  | | --- | | 《interface》  Mylist | |  | | *+insertElem(int,Object):void*  *+insertElem(Object):void*  *+indexOf(Object):int*  *+lastIndexOf(Object):int*  *+remove(int):boolean*  *+remove(Object):boolean*  *+change(int,Object):void*  *+isEmpty():boolean*  *+getLength():int*  *+clear():void* |  |  | | --- | | MyAbstractList | | #size:int | | +MyAbstractList()  +MyAbstractList(Object[])  +insertElem(Object):void  +remove(Object):boolean  +isEmpty():boolean  +getLength():int |   箭头是空心的  继承关系   |  | | --- | | MyList | | -data:Object[]  +CAPACITY:int | | +indexOf(Object):int  +lastIndexOf(Object):int  +remove(int):boolean  +change(int,Object):void  +clear():void  +insertElem(int,Object):void  +toString():String | | | | | |
| 1. 实验源代码     **public** **interface** Eatable {  **public** **abstract** **void** eat();  }  --------------------------------------------  **public** **abstract** **class** Pet **implements** Eatable{  **protected** String name;  **protected** String color;  **protected** String breed;  **public** Pet() {}  **public** Pet(String name, String color, String breed) {  **super**();  **this**.name = name;  **this**.color = color;  **this**.breed = breed;  }  **public** **abstract** **void** cry();  **public** **abstract** **void** eat(); // 在抽象类中可以不实现接口中的方法  **public** **void** speak() {  // 我是一只狗。我叫乐乐，是一只黑色的牧羊犬。我会汪汪汪…  System.***out***.print("我是一只" + toString() + "。我叫" + name + ",是一只" + color + "的" + breed + "。" + "我会");  cry(); // 上转型对象实现多态  eat(); // 接口回调实现多态  }  }  --------------------------------------------  **public** **class** Cat **extends** Pet{  **public** Cat(String name, String color, String breed) {  **super**(name,color,breed);  }  **public** **void** cry() {  System.***out***.println("喵喵喵....");  }  **public** **void** eat() {  System.***out***.println("吃鱼");  }  **public** String toString() {  **return** "猫";  }  }  **public** **class** Dog **extends** Pet{  **public** Dog(String name, String color, String breed) {  **super**(name,color,breed);  }  **public** **void** cry() {  System.***out***.println("汪汪汪....");  }  **public** **void** eat() {  System.***out***.println("吃骨头");  }  **public** String toString() {  **return** "狗";  }  }  **public** **class** TestSimulator {  **public** **static** **void** main(String[] args) {  // **TODO** Auto-generated method stub  Simulator s = **new** Simulator();  s.playSound(**new** Dog("Dog","白色","牧羊犬"));  s.playSound(**new** Cat("Cat","黑色","橘猫"));  }  }  (2)    **public** **abstract** **class** MyAbstractList **implements** MyList{  **protected** **int** size = 0;  **public** MyAbstractList() {  **super**();  }  **public** MyAbstractList (Object[] objects) {  **for**(**int** i=0; i<objects.length; i++) {  insertElem(objects[i]);  }  size = objects.length;  }  **public** **void** insertElem(Object e) {  // **TODO** Auto-generated method stub  insertElem(size,e);  }  **public** **boolean** remove(Object e) {  **if** (indexOf(e) >= 0) {  remove(indexOf(e));  **return** **true**;  }  **return** **false**;  }  **public** **boolean** isEmpty() {  // **TODO** Auto-generated method stub  **return** size == 0;  }  **public** **int** getLength() {  // **TODO** Auto-generated method stub  **return** size;  }  **public** **interface** MyList {  **public** **void** insertElem(**int** index, Object e);  **public** **void** insertElem(Object e);  **public** **int** indexOf(Object e);  **public** **int** lastIndexOf(Object e);  **public** **boolean** remove(**int** index);  **public** **boolean** remove(Object e);  **public** **void** change(**int** index, Object e);  **public** **boolean** isEmpty();  **public** **int** getLength();  **public** **void** clear();  }  **public** **class** MyArrayList **extends** MyAbstractList{  **private** Object[] data = **new** Object[***CAPACITY***];  **public** **static** **final** **int** ***CAPACITY*** = 100;  **public** MyArrayList() {}  **public** MyArrayList(Object[] data) {  **super**(data);  }  **public** **int** indexOf(Object e) {  **for**(**int** i = 0; i < size; i++) {  **if**(data[i].equals(e)) **return** i+1;  }  **return** -1;  }  **public** **boolean** remove(**int** index) {  **for**(**int** i = index-1; i < size; i++) {  data[i] = data[i+1];  }  size--;  **return** **true**;  }  **public** **void** change(**int** index, Object e) {  data[index - 1] = e;  }  **public** **void** clear() {  size = 0;  }  **public** **void** insertElem(**int** index, Object e) {  **for**(**int** i = size-1; i >= index; i--) {  data[i+1] = data[i];  }  data[index] = e;  size++;  }  **public** String toString() {  StringBuffer result = **new** StringBuffer("[");  **for**(**int** i = 0; i < size; i++) {  result.append(data[i]);  **if**(i<size-1) {  result.append(",");  }  }  StringBuffer result1 = **new** StringBuffer("]");  **return** result.toString() + result1.toString();  }  **public** **int** lastIndexOf(Object e) {  **for** (**int** i = size-1; i > 0; i--) {  **if**(data[i].equals(e)) {  **return** i+1;  }  }  **return** -1;  }  }  **import** java.util.Scanner;  **public** **class** TestAbstractList {  **public** **static** **void** main(String[] args) {  // **TODO** Auto-generated method stub  MyArrayList list = **new** MyArrayList();  **while**(**true**) {  System.***out***.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*\*\*\*\*请输入你想进行的操作\*\*\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*\*\*\*\*1.初始化线性表\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*\*\*\*\*2.插入元素到线性表\*\*\*\*\*\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*\*\*\*\*3.陈列所创建的线性表\*\*\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*\*\*\*\*4.删除线性表中的元素\*\*\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*\*\*\*\*5.显示线性表的大小\*\*\*\*\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*\*\*\*\*6.判断线性表是否为空\*\*\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*\*\*\*\*7.替换线性表中的元素\*\*\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*\*\*\*\*8.清除线性表中所有元素\*\*\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*\*\*\*\*9.退出主程序\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  Scanner sc = **new** Scanner(System.***in***);  **int** choice = sc.nextInt();  **switch** (choice) {  **case** 1:  System.***out***.println("请输入你想要创建线性表的大小：");  **int** size = sc.nextInt();  **for**(**int** i = 1; i <= size ; i++ ) {  System.***out***.println("请输入第" + i + "个元素");  String elem = sc.next();  list.insertElem(elem);  }  **break**;  **case** 2:  System.***out***.println("请输入你要插入的位置：");  **int** location = sc.nextInt();  System.***out***.println("请输入你要插入的元素：");  String elem = sc.next();  list.insertElem(location,elem);  **break**;  **case** 3:  System.***out***.println(list.toString());  **break**;  **case** 4:  System.***out***.println("请输入你要删除元素的位置：");  **int** localtion1 = sc.nextInt();  System.***out***.println("该元素是否被删除" + list.remove(localtion1));  **break**;  **case** 5:  System.***out***.println(list.getLength());  **break**;  **case** 6:  System.***out***.println(list.isEmpty());  **break**;  **case** 7:  System.***out***.println("请输入你要替换元素的位置：");  **int** location2 = sc.nextInt();  System.***out***.println("请输入你要替换的元素：");  Object e = sc.next();  list.change(location2,e);  **break**;  **case** 8:  list.clear();  **break**;  **case** 9:  **return**;  }}}}  (3)  **import** nuc\_zhy0340\_shopping\_entity.Book;  **public** **interface** IBookDao {  **public** **void** insertBook();  **public** Book[] quertBooks();  **public** Book queryBookById(**int** bid);  **public** Book queryBookByName(String name);  **public** Book queryBookByAuthor(String author);  **public** Book queryBookByPrice(**int** price);  **public** **void** modifyBook();  }  import nuc\_zhy0340\_shopping\_entity.Book;  import nuc\_zhy0340\_shopping\_entity.Category;  import nuc.sw.shopping.db.\*;  import java.util.Scanner;  public class BookDaoImpl implements IBookDao{  private int index = 0;  private int datasize = 0;  final int DateBaseSize = 100;  private Book[] books = new Book[DateBaseSize];  final Category category1 = new Category(1, "计算机", "java");  final Category category2 = new Category(2, "小说", "名著");    Scanner sc = new Scanner(System.in);  public void insertBook() {  System.out.println("你要添加几本书？");  int amount= sc.nextInt();  datasize = amount;  for(int i = 0; i < datasize; i++) {  System.out.println("请输入书本的名字:");  String name = sc.next();  System.out.println("请输入书本的作者:");  String author = sc.next();  System.out.println("请输入书本的价格:");  float price = sc.nextFloat();  System.out.println("请输入书本的数量:");  int num = sc.nextInt();  Category selectCate = null;  System.out.println(1 + " 计算机 " + "Java");  System.out.println(2 + " 小说 " + "名著");  System.out.println("你想添加1或2哪个种类的书籍？");  int select = sc.nextInt();  switch (select) {  case 1 : {  selectCate = category1;  System.out.println("书籍添加成功!");  break;  }  case 2 : {  selectCate = category2;  System.out.println("书籍添加成功!");  break;  }  }  books[index] = new Book(index, name, author, price, num, selectCate);  index++;  }  }  public Book[] quertBooks() {  return books;  }      public Book queryBookById(int bid) {  Book book = null;  for(int i = 0; i < datasize; i++) {  if(books[i].getBid() == bid) {  return books[i];  }  }  return null;  }  public Book queryBookByName(String name) {  Book book = null;  for(int i = 0; i < datasize; i++) {  if(books[i].getName().equals(name)) {  return books[i];  }  }  return null;  }  public Book queryBookByAuthor(String author) {  Book book = null;  for(int i = 0; i < datasize; i++) {  if(books[i].getAuthor().equals(author)) {  return books[i];  }  }  return null;  }  public Book queryBookByPrice(int price) {  Book book = null;  for(int i = 0; i < datasize; i++) {  if(books[i].getPrice() == price) {  return books[i];  }  }  return null;  }  public void modifyBook() {  System.out.println("请输入你要修改图书的编号：");  int ser = sc.nextInt();  System.out.println("该图书的数据是：" + books[ser]);  while (true) {  System.out.println("1.名字");  System.out.println("2.作者");  System.out.println("3.价格");  System.out.println("4.数量");  System.out.println("5.退出");  System.out.println("请选择你要修改的数据：");  int choice = sc.nextInt();  switch (choice) {  case 1 : {  System.out.println("你要将名字修改成什么？");  String name = sc.next();  books[ser].setName(name);  System.out.println("修改完成");  break;  }  case 2 : {  System.out.println("你要将作者修改成什么？");  String size = sc.next();  books[ser].setAuthor(size);  System.out.println("修改完成");  break;  }  case 3 : {  System.out.println("你要将价格修改成什么？");  int price = sc.nextInt();  books[ser].setPrice(price);  System.out.println("修改完成");  break;  }  case 4 :{  System.out.println("你要将数量修改成什么？");  int num = sc.nextInt();  books[ser].setNum(num);  System.out.println("修改完成");  break;  }  case 5:  return;  }  }  }  }  **public** **class** TextShopping {  **public** **static** **void** main(String[] args) {  IBookDao b = **new** BookDaoImpl();  Book[] books = b.quertBooks();  Scanner sc = **new** Scanner(System.***in***);  **while**(**true**) {  System.***out***.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*\*请输入你想进行的操作\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*1.陈列当前书籍的数据\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*2.根据名字查找书籍\*\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*3.根据作者查找书籍\*\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*4.根据价格查找书籍\*\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*5.根据编号查找书籍\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*6.增加当前的库中数据\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*7.更改当前的库中数据\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*8.退出主程序\*\*\*\*\*\*\*\*\*\*\*\*\*");  System.***out***.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");  **int** choose = sc.nextInt();  **switch** (choose) {  **case** 1 : {  **for** (**int** i = 0; i < books.length - 1; i++) {  Book clo = books[i];  **if**(books[i] != **null**) {  System.***out***.println(clo);  }  }  **break**;  }  **case** 2 : {  System.***out***.println("请输入你想要查找的书籍:");  String boName = sc.next();  Book c = b.queryBookByName(boName);  **if**(c != **null**) {  System.***out***.println(b.queryBookByName(boName));  }**else** System.***out***.println("很抱歉，没有这件书籍");  **break**;  }  **case** 3 : {  System.***out***.println("请输入你想要查找的作者的书籍:");  String Author = sc.next();  Book c = b.queryBookByAuthor(Author);  **if**(c != **null**) {  System.***out***.println(b.queryBookByAuthor(Author));  }**else** System.***out***.println("很抱歉，没有这个作者所著作的书籍");  **break**;  }  **case** 4 : {  System.***out***.println("请输入你想要查找的价格:");  **int** boPrice = sc.nextInt();  Book c = b.queryBookByPrice(boPrice);  **if**(c != **null**) {  System.***out***.println(b.queryBookByPrice(boPrice));  }**else** System.***out***.println("很抱歉，没有属于这个价格的书籍");  **break**;  }  **case** 5 : {  System.***out***.println("请输入你想要查找的编号:");  **int** boBid = sc.nextInt();  Book c = b.queryBookById(boBid);  **if**(c != **null**) {  System.***out***.println(b.queryBookById(boBid));  }**else** System.***out***.println("很抱歉，没有书籍属于这个编码");  **break**;  }  **case** 6 : {  b.insertBook();  **break**;  }  **case** 7 : {  b.modifyBook();  **break**;  }  **case** 8 : {  **return**;  }  }  7.实验运行结果截图及心得体会  (1)  屏幕截图 2023-10-25 123934  (2)  屏幕截图 2023-10-25 122841  屏幕截图 2023-10-25 122847  屏幕截图 2023-10-25 122852  屏幕截图 2023-10-25 122857  屏幕截图 2023-10-25 122904  屏幕截图 2023-10-25 122909  (3)  屏幕截图 2023-10-25 161226  屏幕截图 2023-10-25 161231  屏幕截图 2023-10-25 161236  屏幕截图 2023-10-25 161242  屏幕截图 2023-10-25 161250 | | | |
|  | | | |