实验3.4 类和对象综合实验

|  |
| --- |
| 实验目的：   1. 掌握类的初始化一般顺序。 2. 熟练运用面向对象的方式编写程序。 |
| 实验要求和过程  1、仔细阅读下列程序（程序实现一双目运算计算器），并回答题后的问题。  abstract class Operation {  **private** double opNumberA;  **private** double opNumberB;  public void setOpNumberA(double opNumberA) {  **this**.opNumberA = opNumberA;  }  public void setOpNumberB(double opNumberB) {  **this**.opNumberB = opNumberB;  }  public double getOpNumberA() {  return opNumberA;  }  public double getOpNumberB() {  return opNumberB;  }  public **abstract** double getResult();  }  //加法运算  class OperationAdd **extends** Operation{  public double getResult(){  double result = getOpNumberA() + getOpNumberB();  return result;  }  }  //除法运算  class OperationDivide **extends** Operation {  public double getResult() {  double result = getOpNumberA() / getOpNumberB();  return result;  }  }  class OperationFactory{  public static Operation createOperate(char operate){  Operation oper = null;  switch(operate){  case '+':  oper = **new** OperationAdd() ;  break;  case '/':  oper = **new** OperationDivide() ;  break;  }  return oper;  }  }  public class OperationTest{  public static void main(String[] args) {  Operation oper;  A: oper = OperationFactory.createOperate(‘/’);  B: oper.setOpNumberA(90); oper.setOpNumberB(0);  double result = oper.getResult( );  System.out.println(result);  }  }   1. 按最优方式完成程序中空缺部分。 2. 程序运行结果是 Infinity 。   A处语句和B处语句的位置能交换吗？ 不能 。  2、预测下列程序的运行结果，总结Java类初始化的一般顺序。  class Sample {  Sample(String s) {  System.out.println(s);  }  Sample() {  System.out.println("Sample默认构造函数被调用");  }  }  class Test {  static {  System.out.println("父类static 块 执行11111");  }  static Sample staticSam1 = new Sample("父类 静态成员staticSam1初始化222222");  Sample sam1 = new Sample("父类 sam1成员初始化555555");  Test() {  System.out.println("父类 Test默认构造函数被调用77777");  }  Sample sam2 = new Sample("父类 sam2成员初始化666666");  }  class TestSub extends Test {  static Sample staticSamSub = new Sample("子类 静态成员staticSamSub初始化3333");  TestSub() {  System.out.println("子类 TestSub 默认构造函数被调用xxxx");  }  Sample sam1 = new Sample("子类 sam1成员初始化88888");  static Sample staticSamSub1 = new Sample("子类 静态成员staticSamSub1初始化4444");  Sample sam2 = new Sample("子类 sam2成员初始化999999");  }  public class TT {  public static void main(String str[]) {  TestSub a = new TestSub();  }  }  父类static 块 执行11111  父类 静态成员staticSam1初始化222222  子类 静态成员staticSamSub初始化3333  子类 静态成员staticSamSub1初始化4444  父类 sam1成员初始化555555  父类 sam2成员初始化666666  父类 Test默认构造函数被调用77777  子类 sam1成员初始化88888  子类 sam2成员初始化999999  子类 TestSub 默认构造函数被调用xxxx  静态优先非静态，再执行构造方法  3\*、（可结合作业上的一些要求）参考以下类图，利用所学面向对象的方式编写一个学生信息管理系统。    MainClass类可参照如下，完成该系统的基本功能框架（具体实现可渐次完成）。  **import** java.util.Scanner;  **public** **class** **MainClass** {  **static** **int** **menu**(**Scanner** **scanner**) {  **int** choice = -1;  **System**.**out**.**printf**("\n");  **System**.**out**.**printf**("\t\t\t 学生信息管理系统功能菜单 \n");  **System**.**out**.**printf**("\t\t\t ======================= \n");  **System**.**out**.**printf**("\t\t\t 1.插入学生信息记录 \n");  **System**.**out**.**printf**("\t\t\t 2.删除学生信息记录 \n");  **System**.**out**.**printf**("\t\t\t 3.更新学生信息记录 \n");  **System**.**out**.**printf**("\t\t\t 4.显示所有学生信息记录 \n");  **System**.**out**.**printf**("\t\t\t 5.按学号查询指定学生信息 \n");  **System**.**out**.**printf**("\t\t\t 0.结束程序 \n");  **System**.**out**.**printf**("\t\t\t ======================= \n");  **System**.**out**.**printf**("\t\t\t 请输入您的选择： \n");  choice = **scanner**.**nextInt**();  **return** choice;  }  **public** **static** **void** **main**(**String**[] **args**) {  **int** menuChoice = -1;  **Scanner** scanner = **new** **Scanner**(**System**.**in**);  **do** {  menuChoice = ***menu***(scanner);  **switch** (menuChoice) {  // 插入学生信息记录  **case** 1:  // 删除学生信息记录  **case** 2:  // 修改更新学生信息记录  **case** 3:  // 显示所有学生信息记录  **case** 4:  // 按学号查询指定学生信息  **case** 5:  }  } **while** (menuChoice != 0);  scanner.**close**();  }  } |