Java实验报告二

姓名：王维

班级：计算机科学与技术二班

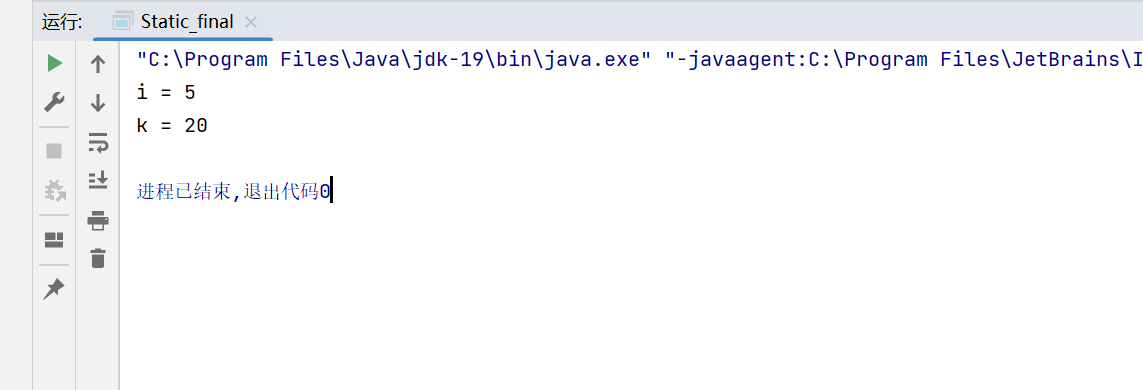
学号：320210942931

一：5.3.1（1）

1. 源代码：

|  |
| --- |
| package ExperimentSourceCode\_2.fill;  class Static\_final {  static int *i* = 10;  static final int *k* = 20;  public static void main(String[] args){  System.*out*.println("i = "+*i*);  System.*out*.println("k = "+*k*);}  static{ *i*=*i*/2; } } |

1. 运行结果：



二：5.3.1（2）

1. 源代码：

|  |
| --- |
| package ExperimentSourceCode\_2.Abstract; abstract class Employee {  String EmpName;  char sex;  double EmpSal;   Employee(String en, char s, double es) {  EmpName = en;  sex = s;  EmpSal = es;  }   public String getName() {  return EmpName;  }   public char getSex() {  return sex;  }   public abstract double getSal();   public void setSal(int basSal) {  EmpSal = basSal;  } }  class Worker extends Employee {  char category;  boolean dressAllowance;   Worker(String en, char s, double es, char c, boolean d) {  super(en, s, es);  category = c;  dressAllowance = d;  }   public char getCategory() {  return category;  }   public boolean getDressAllowance() {  return dressAllowance;  }   public double getSal() {  return EmpSal;  } }  class Superior extends Employee {  int experience;  boolean vehicle;  double medicalAllowance;   Superior(String en, char s, double es, int e, boolean v, double ma) {  super(en, s, es);  experience = e;  vehicle = v;  medicalAllowance = ma;  }   public int getExp() {  return experience;  }   public boolean getVehicle() {  return vehicle;  }   public double getMedicalAll() {  return medicalAllowance;  }   public double getSal() {  return EmpSal \* 4 + 1000 + medicalAllowance;  } }  class Officer extends Superior {  double travelAllowance;   Officer(String en, char s, double es, int e, boolean v, double ma, double ta) {  super(en, s, es, e, v, ma);  travelAllowance = ta;  }   public double getTravelAllowance() {  return travelAllowance;  }   public double getSal() {  return EmpSal \* 2 + 200 + travelAllowance + medicalAllowance;  } }  class Manager extends Superior {  double clubAllowance;   Manager(String en, char s, double es, int e, boolean v, double ma, double ca) {  super(en, s, es, e, v, ma);  clubAllowance = ca;  }   public double getClubAll() {  return clubAllowance;  }   public double getSal() {  return EmpSal \* 5 + 2000 + medicalAllowance + clubAllowance;  } }  class InheDemo {  public static void main(String[] args) {  Worker w = new Worker("M. John", 'M', 1200.50, 'B', true);  System.*out*.println("工人信息：");  System.*out*.println("姓名：" + w.getName());  System.*out*.println("性别：" + w.getSex());  System.*out*.println("薪资：" + w.getSal());  System.*out*.println("类别：" + w.getCategory());  if (w.getDressAllowance()) {  System.*out*.println("提供服装津贴");  } else {  System.*out*.println("未提供服装津贴");  }   Officer o = new Officer("S. David", 'F', 2500.70, 15, true, 345.60, 200);  System.*out*.println("\n主任信息：");  System.*out*.println("姓名：" + o.getName());  System.*out*.println("性别：" + o.getSex());  System.*out*.println("薪资：" + o.getSal());  System.*out*.println("工作经验：" + o.getExp() + "年");  if (o.getVehicle()) {  System.*out*.println("提供交通工具");  } else {  System.*out*.println("未提供交通工具");  }  System.*out*.println("医疗津贴：" + o.getMedicalAll());  System.*out*.println("出差津贴：" + o.getTravelAllowance());   Manager m = new Manager("Arnoldshwaz", 'M', 4500.70, 15, true, 345.60, 300);  System.*out*.println("\n经理信息：");  System.*out*.println("姓名：" + m.getName());  System.*out*.println("性别：" + m.getSex());  System.*out*.println("薪资：" + m.getSal());  System.*out*.println("工作经验：" + m.getExp() + "年");  if (m.getVehicle()) {  System.*out*.println("提供交通工具");  } else {  System.*out*.println("未提供交通工具");  }  System.*out*.println("医疗津贴：" + m.getMedicalAll());  System.*out*.println("会员津贴：" + m.getClubAll());  } } |

1. 运行结果：

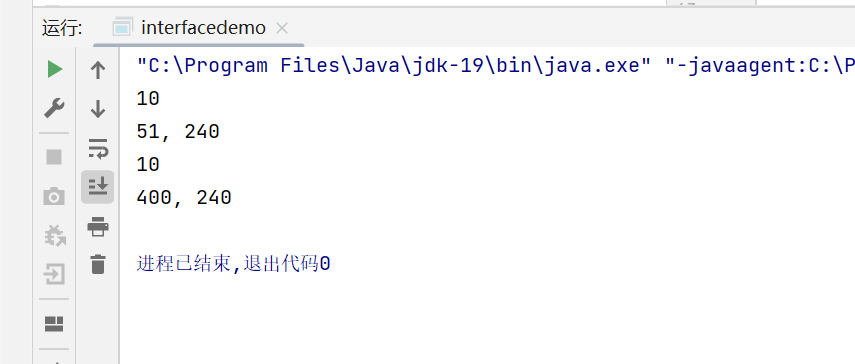


三：5.3.1（3）

1. 源代码：

|  |
| --- |
| package ExperimentSourceCode\_2.fill;   interface Computable {  int *M* = 10;   int f(int x);   public abstract int g(int x, int y); }  class A implements Computable {  public int f(int x) {  return 1 + *M* + 2 \* x;  }   public int g(int x, int y) {  return *M* \* (x + y);  } }  class B implements Computable {  public int f(int x) {  return x \* x;  }   public int g(int x, int y) {  return x \* y \* *M*;  } }  public class interfacedemo {  public static void main(String[] args) {  Computable a = new A();  Computable b = new B();  System.*out*.println(a.*M*);  System.*out*.println("" + a.f(20) + ", " + b.g(12, 2));  System.*out*.println(b.*M*);  System.*out*.println("" + b.f(20) + ", " + b.g(12, 2));  } } |

1. 运行结果：



四：5.3.2（3），5.3.3

因两代码写在了一个包里，故一起演示

1. 源代码：

（1）接口Common

|  |
| --- |
| package ExperimentSourceCode\_2.vehicle.all;  public interface Common {  double runTime(double a,double b,double c); } |

（2）Car实现

|  |
| --- |
| package ExperimentSourceCode\_2.vehicle;  import ExperimentSourceCode\_2.vehicle.all.Common;  public class Car implements Common {  @Override  public double runTime(double a, double b, double c) {  return a\*b/c;  } } |

（3）Plane实现

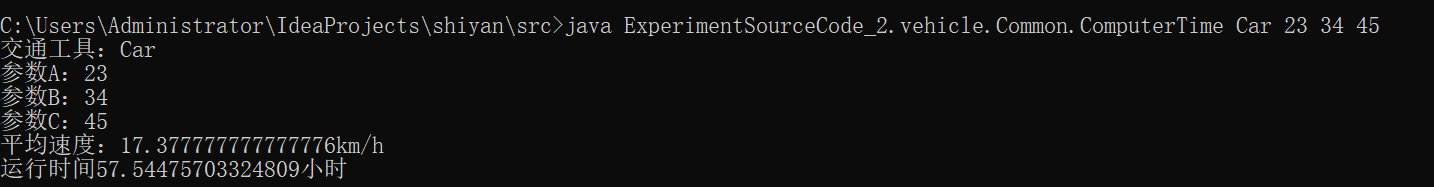
|  |
| --- |
| package ExperimentSourceCode\_2.vehicle;  import ExperimentSourceCode\_2.vehicle.all.Common;  public class Plane implements Common {   @Override  public double runTime(double a, double b, double c) {  return a+b+c;  } } |

（4）Ship实现

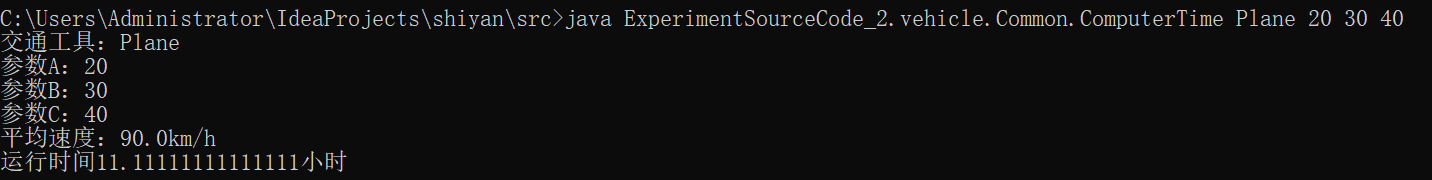
|  |
| --- |
| package ExperimentSourceCode\_2.vehicle; import ExperimentSourceCode\_2.vehicle.all.Common; public class Ship implements Common {  @Override  public double runTime(double a, double b, double c) {  return a+b/c;  } } |

1. 运行结果：
2. 命令窗口：（如书上图所示进行运行）

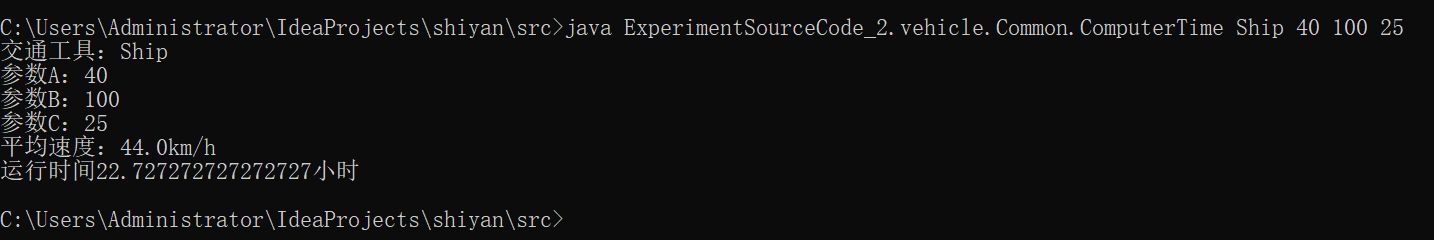
* Car：



* Plane：

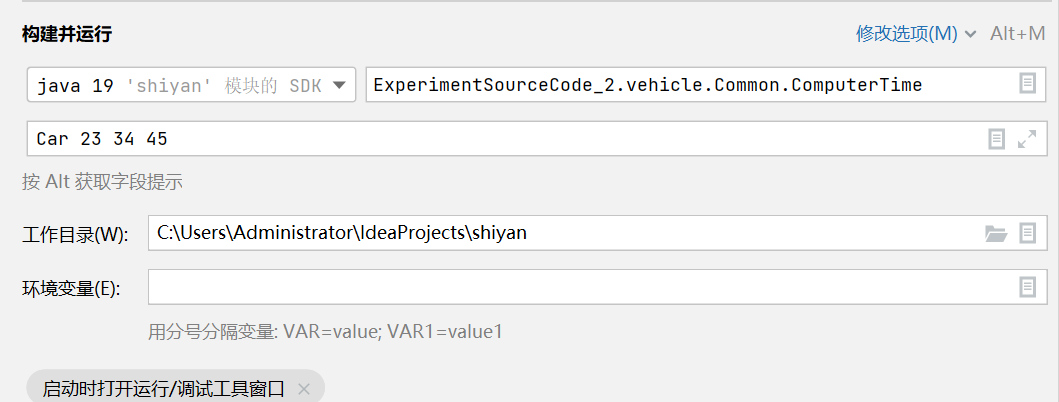


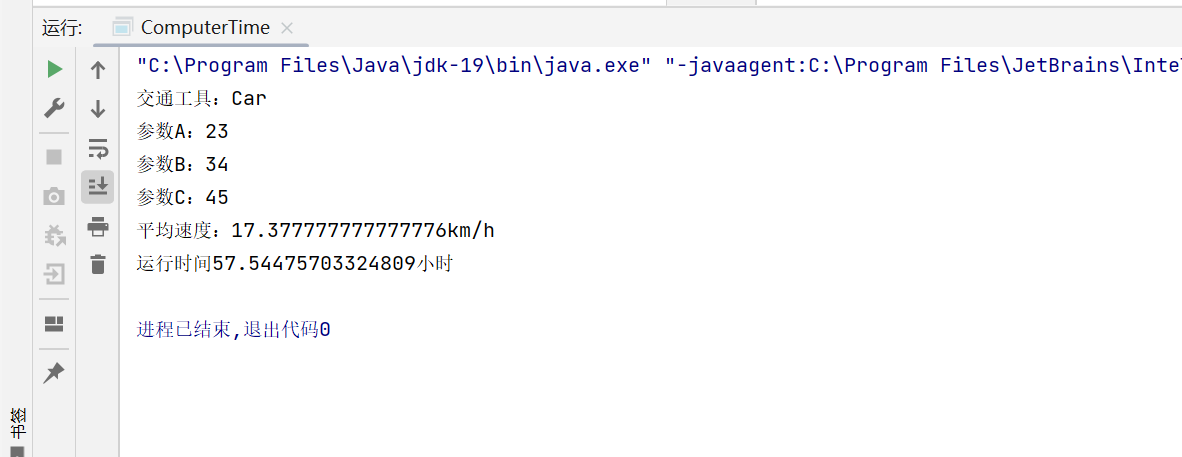
* Ship：



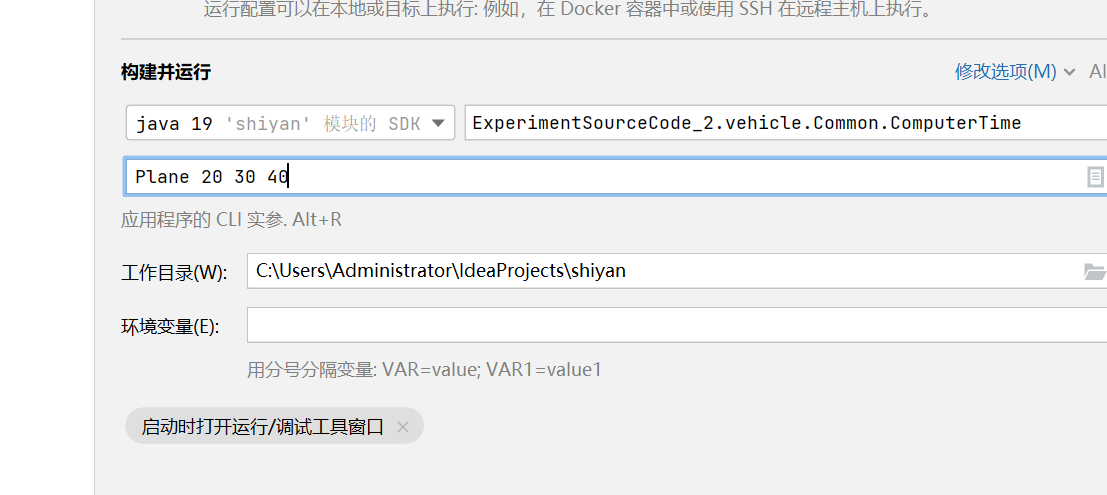
1. Idea：编辑编译程序实参，再进行运行：

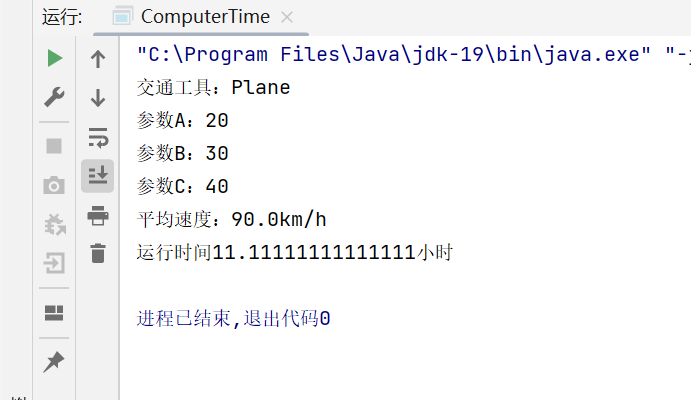
* Car：



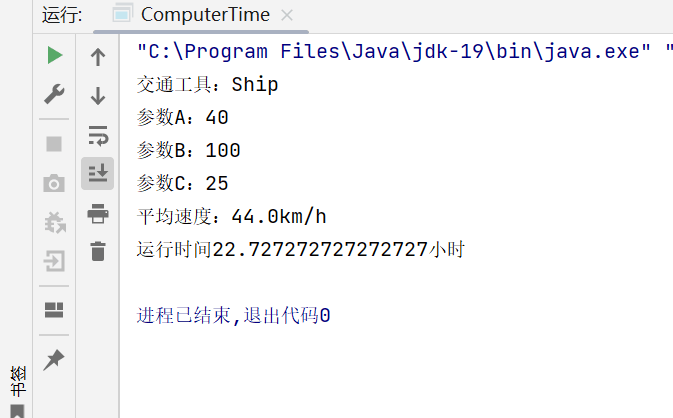
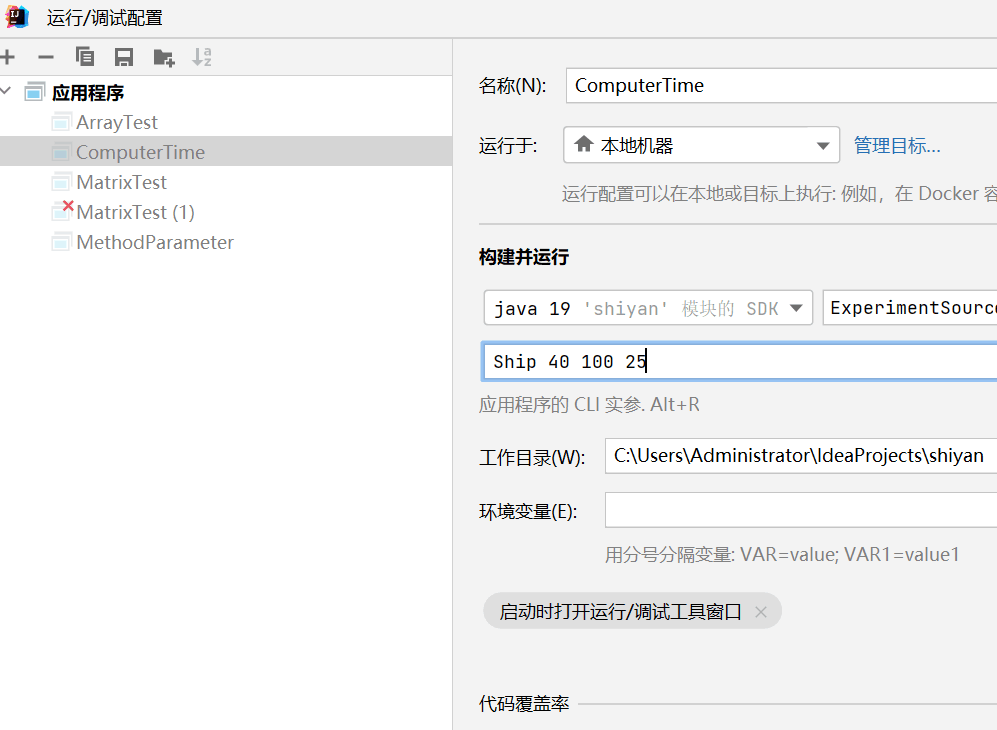


* Plane：





* Ship：



1. 设计程序收获：
2. 接口概念：接口在Java中是一个抽象类型，是抽象方法的集合，接口通常以interface来声明。一个类通过继承接口的方式，从而来继承接口的抽象方法。接口包含类要实现的方法。除非实现接口的类是抽象类，否则该类要定义接口中的所有方法。接口无法被实例化，但是可以被实现。一个实现接口的类，必须实现接口内所描述的所有方法，否则就必须声明为抽象类。
3. 接口的作用：

* 通过接口可以实现不相关类的相同行为，而不需要了解对象所对应的类。
* 通过接口可以指明多个类需要实现的方法。
* 通过接口可以了解对象的交互界面，而不需了解对象所对应的类。