1. 设计一个工资管理系统，针对两种雇员的工资进行管理：长期合同工、短期小时工；

合同工的工资按月发放，小时工的工资为每小时工资乘以工作小时数；都是月底结算；

如果当月为雇员生日所在月，则当月加发200元的贺金；

建立一个雇员父类Employee，两种雇员类继承Employee类；

利用循环结构，输出各个雇员对象的工资；

（扩展功能，不必须实现：如果是合同工，则给该员工每月加发当月所有人基本工资和的千分之五做为红利；）

构造方法：

Employee(String name, String employeeId, int year, int month, int day)

SalariedEmployee(String name, String employeeId, int year, int month, int day, double salary)

HourlyEmployee(String name, String employeeId, int year, int month, int day, double wage, double hours)

最后创建一个测试类SalaryManagement，使用前面的三个类，输出下面四个雇员的相关信息；再单独输出每个雇员的类型；

雇员如下：

合同工张三，员工号：111-11-1111，生日：1975年2月5日，工资：2000.00

合同工李四，员工号：222-22-2222，生日：1978年11月25日，工资：2222.00

小时工王五，员工号：333-33-3333，生日：1974年6月24日，每小时工资：20.00,

工作了120小时

小时工赵六，员工号：444-44-4444，生日：1975年8月9日，每小时工资：30.00,

工作了56小时

假设当月为2月份；

**public** **abstract** **class** Employee {

**private** String name; //雇员姓名

**private** String employeeId; //雇员工号

**public** **int** year; //雇员生日所在 年

**public** **int** month; //雇员生日所在 月

**public** **int** day; //雇员生日所在 日

**public** Employee(String name, String employeeId, **int** year, **int** month, **int** day){

**this**.name = name;

**this**.employeeId = employeeId;

**this**.year = year;

**this**.month = month;

**this**.day = day;

}

//返回雇员信息

**public** String toString(){

**return** name+"\n 雇员工号："+employeeId+"\n 雇员生日："+getBirthDate();

}

//抽象方法：被子类覆盖

**public** **abstract** **double** earnings(); //返回工资数

**public** **abstract** String getEmployeeType(); //返回雇员类型

//获取生日的字符串

**public** String getBirthDate() {

**return** year + "-"+month+"-"+day;

}

//获取月份

**public** **int** getMonth() {

**return** month;

}

}

**public** **class** HourlyEmployee **extends** Employee{

**private** **double** wage; //每小时工资

**private** **double** hours; //工作小时数

**public** HourlyEmployee(String name, String employeeId,

**int** year, **int** month, **int** day, **double** wage, **double** hours){

**super**(name, employeeId, year, month, day);

setWage(wage);

setHours(hours);

}

**public** **void** setWage(**double** wage) {

**this**.wage = wage<0.0 ? 0.0 : wage;

}

**public** **void** setHours(**double** hours) {

//按照每天8小时，每月工作21天计算

**this**.hours = (hours>=0.0&&hours<=8.0\*21) ? hours : 0.0;

}

**public** **double** earnings(){

**return** hours \* wage;

}

**public** String toString(){

**return** "\n 小时工："+**super**.toString();

}

**public** String getEmployeeType(){

**return** "小时工";

}

}

**public** **class** SalariedEmployee **extends** Employee{

**private** **double** salary;

**public** SalariedEmployee(String name, String employeeId,

**int** year, **int** month, **int** day, **double** salary){

**super**(name, employeeId, year, month, day);

setSalary(salary);

// this.salary = salary<0.0 ? 0.0 : salary;

}

**public** **void** setSalary(**double** salary) {

**this**.salary = salary<0.0 ? 0.0 : salary;

}

**public** **double** earnings(){

**return** salary;

}

**public** String toString(){

**return** "\n 合同工："+**super**.toString();

}

**public** String getEmployeeType(){

**return** "合同工";

}

}

**public** **class** SalaryManagement {

**public** **static** **void** main(String[] args){

//创建雇员数组

Employee[] employees = **new** Employee[4];

//初始化雇员信息

employees[0] = **new** SalariedEmployee("张三","111-11-1111",1975,2,5,2000.00);

employees[1] = **new** SalariedEmployee("李四","222-22-2222",1978,11,25,2222.00);

employees[2] = **new** HourlyEmployee("王五","333-33-3333",1974,6,24,20.00,120);

employees[3] = **new** HourlyEmployee("赵六","444-44-4444",1975,8,9,30.00,56);

String output = "";

**int** currentMonth = 2; //设定当前月份

**for**(**int** i=0; i< employees.length; i++){

output += employees[i].toString();

output += "\n 基本工资为："+employees[i].earnings();

//如果当前月份为某雇员的生日所在月份，则加发200元工资

**if**(currentMonth == employees[i].getMonth()){

output += "\n 实发工资为："+(employees[i].earnings()+200)+" 额外增发200元生日贺金\n";

}**else**{

output += "\n 实发工资为："+employees[i].earnings()+" \n";

}

}

**for**(**int** i=0; i<employees.length; i++){

output += "\n 雇员"+i+"是"+employees[i].getEmployeeType();

}

System.***out***.println(output);

}

}

2. 编写Addition类，该类中应包含一组实现两数相加运算的重载方法。 实现加法运算的方法，应接受两个参数（即加数和被加数），方法将两个参数进行加法运算后，返回相加结果。考虑可能针对不同的数据类型进行计算，重载一组方法，包括整型、长整型、浮点型、双精度浮点型、还有字符串。 在main方法中创建Addition类的实例，分别调用重载方法测试其效果。 应将Addition类打入到包中，包名为com.epoint.practice。

**package** com.epoint.practice;

**class** Addition

{

**public** Addition(){

}

**void** add(**int** a,**int** b){

**int** c = a+b;

System.***out***.println("a+b="+c);

}

**void** add(**long** a,**long** b){

**long** c = a+b;

System.***out***.println("a+b="+c);

}

**void** add(**float** a,**float** b){

**float** c = a+b;

System.***out***.println("a+b="+c);

}

**void** add(**double** a,**double** b){

**double** c = a+b;

System.***out***.println("a+b="+c);

}

**void** add(String a,String b){

String c = a+b;

System.***out***.println("a+b="+c);

}

}

**public** **class** AdditionTest {

**public** **static** **void** main(String args[]){

Addition m = **new** Addition();

Addition n = **new** Addition();

m.add(5,6);

n.add("hello","world");

}

}

3. 创建一个Vehicle类并将它声明为抽象类。在Vehicle类中声明一个NoOfWheels方法，使它返回一个字符串值。创建两个类Car和Motorbike从Vehicle类继承，并在这两个类中实现NoOfWheels方法。在Car类中，应当显示“四轮车”信息；而在Motorbike类中，应当显示“双轮车”信息。创建另一个带main方法的类，在该类中创建Car和Motorbike的实例，并在控制台中显示消息。

**abstract** **class** Vehicle

{

**public** **abstract** String Noofwheels();

}

**class** Car **extends** Vehicle

{

**int** a;

**int** b;

**public** Car(**int** a, **int** b) {

**this**.a = a;

**this**.b = b;

}

**public** String Noofwheels() {

**return** "四轮车————\n载重：" + a + "\n体积：" + b;

}

}

**class** Motorbike **extends** Vehicle

{

**int** c;

**int** d;

**public** Motorbike(**int** c, **int** d) {

**this**.c = c;

**this**.d = d;

}

**public** String Noofwheels() {

**return** "双轮车————\n长度：" + c + "\n高度：" + d;

}

}

**public** **class** VehicleTest

{

**public** **static** **void** main(String args[]) {

Vehicle m = **new** Car(88, 44);

System.***out***.println(m.Noofwheels());

Vehicle n = **new** Motorbike(3, 4);

System.***out***.println(n.Noofwheels());

}

}

4. 设计一个系统： xxx纯净水生产线

目前流程是：从某个地方把水取出来，然后经过缓冲，过滤，加热和放糖的步骤

abstract 水{ public void 水(); }

interface 过滤{} interface 缓冲{} interface 加热{} interface 放糖{}

class 纯净水1 extends 水 imps 过滤,缓冲,加热{}

class 纯净水2 extends 水 imps 缓冲,放糖{}

class 纯净水3 extends 水 imps 过滤{}

**abstract** **class** Water

{

**public** **abstract** **void** Water(); //取水

}

**interface** Filter

{

**public** **abstract** **void** Filter(); //过滤

}

**interface** Buffer

{

**public** **abstract** **void** Buffer(); //缓冲

}

**interface** Heat

{

**public** **abstract** **void** Heat(); //加热

}

**interface** Sugar

{

**public** **abstract** **void** Sugar(); //放糖

}

**class** Water1 **extends** Water **implements** Filter, Buffer,Heat

{

**public** **void** Water() {

System.***out***.println("水1的制造过程：\n");

}

**public** **void** Filter() {

System.***out***.println("水已经过滤\n");

}

**public** **void** Buffer() {

System.***out***.println("水已经缓冲\n");

}

@Override

**public** **void** Heat() {

System.***out***.println("水已经加热\n");

}

}

**class** Water2 **extends** Water **implements** Buffer,Sugar

{

**public** **void** Water() {

System.***out***.println("水2的制造过程：\n");

}

**public** **void** Buffer() {

System.***out***.println("水已经缓冲\n");

}

@Override

**public** **void** Sugar() {

System.***out***.println("水已经放糖\n");

}

}

**class** Water3 **extends** Water **implements** Filter

{

**public** **void** Water() {

System.***out***.println("水3的制造过程：\n");

}

**public** **void** Filter() {

System.***out***.println("水已经过滤\n");

}

}

**public** **class** TestWater

{

**public** **static** **void** main(String args[]) {

Water1 water1 = **new** Water1();

Water2 water2 = **new** Water2();

Water3 water3 = **new** Water3();

water1.Water();

water1.Buffer();

water1.Filter();

water1.Heat();

water2.Water();

water2.Buffer();

water2.Sugar();

water3.Water();

water3.Filter();

}

}