9.3

**package** demo2;

**import** java.util.Date;

**public** **class** helloworld {

Date date = **new** Date();

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

Date date = **new** Date();

**for**(**long** i = 10000; i <= 1e10;i=i\*10) {

date.setTime(i);

System.***out***.println(date.toString());

}

}

}

9.4

**package** demo2;

**import** java.util.Random;

**public** **class** helloworld {

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

Random ran = **new** Random(1000);

**for** (**int** i = 0 ; i < 50 ; i++){

System.***out***.print(ran.nextInt(100) + " ");

}

}

}

9.5

**package** demo2;

**import** java.util.GregorianCalendar;

**import** java.util.Scanner;

**public** **class** helloworld {

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

GregorianCalendar calendar = **new** GregorianCalendar();

*showtime*(calendar);

**long** a =1234567898765L;

calendar.setTimeInMillis(a);

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

*showtime*(calendar);

}

**public** **static** **void** showtime(GregorianCalendar calendar){

System.***out***.print(calendar.get(GregorianCalendar.***YEAR***)+":");

System.***out***.print(calendar.get(GregorianCalendar.***MONTH***)+1+ ":");

System.***out***.print(calendar.get(GregorianCalendar.***DAY\_OF\_MONTH***));

}

}

9.6

**package** demo2;

**import** java.util.GregorianCalendar;

**import** java.util.Scanner;

**class** StopWatch{

**private** **long** startTime;

**private** **long** endTime;

**public** **void** setStartTime(**long** newStartTime){

startTime = newStartTime;

}

**public** **void** setEndTime(**long** newEndTime){

endTime = newEndTime;

}

**public** StopWatch(){

startTime = System.*currentTimeMillis*();

}

**public** **void** start() {

startTime = System.*currentTimeMillis*();

}

**public** **void** end() {

endTime = System.*currentTimeMillis*();

}

**public** **long** getElapsedTime() {

**return** endTime - startTime;

}

}

**public** **class** helloworld {

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

StopWatch time = **new** StopWatch();

**int**[] nums = **new** **int**[100000];

**for**(**int** i = 1; i <= 100000; ++i)

nums[i - 1] = (**int**)((System.*currentTimeMillis*() / i) % 1000);

time.start();

**for**(**int** i = 0; i < 99999; ++i){

**for**(**int** j = i+1; j < 100000; ++j){

**if**(nums[i] > nums[j]){

**int** temp = nums[i];

nums[i] = nums[j];

nums[j] = temp;

}

}

}

time.end();

System.***out***.println(time.getElapsedTime());

}

}

9.10

**package** demo2;

**import** java.util.GregorianCalendar;

**import** java.util.Scanner;

**class** QuadraticEquation{

**private** **double** a,b,c;

**public** **void** getA(**double** x) {

a=x;

}

**public** **void** getB(**double** x) {

b=x;

}

**public** **void** getC(**double** x) {

c=x;

}

**public** **int** getDiscriminant(){

**if**( b\*b-4\*a\*c>0)**return** 1;

**else** **if**(b\*b-4\*a\*c==0)**return** 0;

**return** -1;

}

**public** **double** getRoot1(){

**double** r1=(Math.*sqrt*(b\*b-4\*a\*c)-b)/(2\*a);

**return** r1;

}

**public** **double** getRoot2(){

**double** r2=(-Math.*sqrt*(b\*b-4\*a\*c)-b)/(2\*a);

**return** r2;

}

}

**public** **class** helloworld {

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

Scanner cin = **new** Scanner(System.***in***);

**double** a,b,c;

a=cin.nextDouble();

b=cin.nextDouble();

c=cin.nextDouble();

QuadraticEquation qua=**new** QuadraticEquation();

qua.getA(a);

qua.getB(b);

qua.getC(c);

**if**(qua.getDiscriminant()==1) {

System.***out***.println(qua.getRoot1());

System.***out***.println(qua.getRoot2());

}

**else** **if**(qua.getDiscriminant()==0) {

System.***out***.println(qua.getRoot1());

}

**else**

System.***out***.println("The equation has no roots.");

}

}

9.11

**package** demo2;

**import** java.util.GregorianCalendar;

**import** java.util.Scanner;

**class** LinearEquation {

**private** **double** a;

**private** **double** b;

**private** **double** c;

**private** **double** d;

**private** **double** e;

**private** **double** f;

LinearEquation(**double** a,**double** b,**double** c,**double** d,**double** e,**double** f){

**this**.a=a;

**this**.b=b;

**this**.c=c;

**this**.d=d;

**this**.e=e;

**this**.f=f;

}

**public** **double** getA() {

**return** a;

}

**public** **double** getB() {

**return** b;

}

**public** **double** getC() {

**return** c;

}

**public** **double** getD() {

**return** d;

}

**public** **double** getE() {

**return** e;

}

**public** **double** getF() {

**return** f;

}

**boolean** isSolvable() {

**if**((a\*d-b\*c)==0)

**return** **false**;

**else**

**return** **true**;

}

**double** getX() {

**double** x;

x=(e\*d-b\*f)/(a\*d-b\*c);

**return** x;

}

**double** getY() {

**double** y;

y=(a\*f-e\*c)/(a\*d-b\*c);

**return** y;

}

}

**public** **class** helloworld {

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

Scanner input = **new** Scanner(System.***in***);

System.***out***.println("请输入a、b、c、d、e、f： ");

**double** a=input.nextDouble();

**double** b=input.nextDouble();

**double** c=input.nextDouble();

**double** d=input.nextDouble();

**double** e=input.nextDouble();

**double** f=input.nextDouble();

LinearEquation l = **new** LinearEquation(a,b,c,d,e,f);

System.***out***.println("a: "+l.getA());

System.***out***.println("b: "+l.getB());

System.***out***.println("c: "+l.getC());

System.***out***.println("d: "+l.getD());

System.***out***.println("e: "+l.getE());

System.***out***.println("f: "+l.getF());

**if**(!(l.isSolvable()))

System.***out***.println("The equation has no solution.");

**else**

{

System.***out***.println("X is "+l.getX());

System.***out***.println("Y is "+l.getY());

}

}

}