

**面向对象程序设计（双语）大作业**

**2020年第一学期**

**学院： 计算机科学学院**

**班级：计算机科学与技术1802班**

**姓名： 段星雨**

目录

[整体目录结构 1](#_Toc44606763)

[Task 1（25 points） 1](#_Toc44606764)

[1.1题目思路 1](#_Toc44606765)

[1.2源代码，UML图，程序框图 2](#_Toc44606766)

[Sequence.java 2](#_Toc44606767)

[测试程序：Text.java 3](#_Toc44606768)

[1.3测试结果 5](#_Toc44606769)

[Task 2（25 points） 6](#_Toc44606770)

[2.1题目思路 6](#_Toc44606771)

[2.2源代码 7](#_Toc44606772)

[ArithmeticExpression.java 7](#_Toc44606773)

[测试程序：Text.java 8](#_Toc44606774)

[2.3测试结果 9](#_Toc44606775)

[out.txt 9](#_Toc44606776)

[文件已存在 10](#_Toc44606777)

[Task 3（25 points） 11](#_Toc44606778)

[3.1题目思路 12](#_Toc44606779)

[3.2源代码，UML图 12](#_Toc44606780)

[ArmedPolice.java 12](#_Toc44606781)

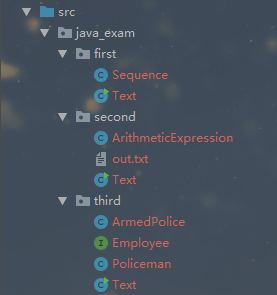
[Employee.java 13](#_Toc44606782)

[Policeman.java 13](#_Toc44606783)

[测试程序：Text.java 15](#_Toc44606784)

[3.3测试结果 16](#_Toc44606785)

# 整体目录结构



# Task 1（25 points）

Design the program according to the following requirements, draw the corresponding UML diagram, program flow chart, and list the complete program code.

(1) Read in a sequence of Numbers separated by commas.

(2) Design a method to complete sequence inversion.

(3) Design a method to complete the sequencing of the sequence.

(4) Design a method to complete the calculation of the average value of the sequence.

(5) Design an algorithm to find the number closest to the average value, and display its position in the sequence.

根据以下要求设计程序，绘制相应的UML图，程序流程图，并列出完整的程序代码。

（1）读取以逗号分隔的数字序列。

（2）设计一种完成翻转序列的方法。

（3）设计一种完成排序序列的方法。

（4）设计一种方法来完成求平均值的计算。

（5）设计一种算法，以找到最接近平均值的数字，并显示其在序列中的位置。

## 1.1题目思路

读取数列方法是依次扫描，直到碰见逗号是将一扫的字符串转化为数字储存，然后继续扫描，知道字符串末尾。

翻转的方法是每次交换对称的两个元素。

排序使用冒泡排序。

平均值的算法是先读取所有数字，然后求和取平均值。

最接近平均值的算法是，数列里的数字依次与平均值求差值，最小的差值对应的数就是结果。

## 1.2源代码，UML图，程序框图

### Sequence.java

package java\_exam.first;

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class Sequence {

List<Integer> list = new ArrayList<>();

int len;

public void read() {

Scanner sc = new Scanner(System.in);

String str = sc.next();

//String会占用每次修改都会占用过多的缓冲区空间，StringBuffer则不会

StringBuffer sb = new StringBuffer();

//使用charAt会占用跟多时间，将字符串转化为数组速度跟快

for (char c : str.toCharArray())

if (c == ',') {

list.add(Integer.parseInt(sb.toString()));

sb = new StringBuffer();

} else sb.append(c);

//最后一次未录入

list.add(Integer.parseInt(sb.toString()));

len = list.size();

}

public void reversal() {

//每次交换i和中心对称的元素

for (int i = 0; i < len / 2; i++) {

int tem = list.get(i);

list.set(i, list.get(len - 1 - i));

list.set(len - 1 - i, tem);

}

}

public void sort() {

//冒泡排序

for (int i = 0; i < len - 1; i++)

for (int j = 0; j < len - 1 - i; j++)

if (list.get(j) > list.get(j + 1)) {

int tem = list.get(j);

list.set(j, list.get(j + 1));

list.set(j + 1, tem);

}

}

public double average() {

double avg = 0;

for (int i = 0; i < len; i++)

avg += list.get(i);

return avg / len;

}

public int nearAvg() {

//index记录索引，dif记录数字与平均值的差值

int index = 0;

double avg = average(), dif = Math.abs(avg - list.get(0));

for (int i = 1; i < len; i++)

if (Math.abs(avg - list.get(i)) < dif) {

//更新新的值

dif = Math.abs(avg - list.get(i));

index = i;

}

return index;

}

}

### 测试程序：Text.java

package java\_exam.first;

public class Text {

public static void main(String[] args) {

Sequence s = new Sequence();

//读取数列

s.read();

System.out.println(s.list);

//求平均值

System.out.println("avg = " + s.average());

//翻转

s.reversal();

System.out.println("翻转：" + s.list);

//排序

s.sort();

System.out.println("排序：" + s.list);

//找到最接近平均值的数

int nearAvg = s.nearAvg();

System.out.println("找到最接近平均值的索引为：" + nearAvg + "，值为" + s.list.get(nearAvg));

}

}

Sequence

+ len : int

+ list : List

+ read () : void

+ reversal () : void

+ sort () : void

+ average () : double

+ nearAvg () : int

**是**

**开始**

**i=0**

**交换len - 1 – i和i的值**

**i++**

**i > len/2**

**reversal ()**

**否**

**结束**

**是**

**将数字录入数组**

否

**开始**

**扫描数组**

**是否出现逗号**

**read()**

**否**

**是否到结尾**

**是**

**结束**

**是**

**开始**

**结束**

**avg=0，i=0**

**i > len**

**avg+=list[i]**

**i++**

**average ()**

**否**

**输出avg/len**

**开始**

**结束**

**abs(avg–list[i]<dif)**

**i<len**

**dif = abs(avg – list[0])**

**index=0**

**i=1**

**dif=abs(avg-list[i])**

**index = i;**

**nearAvg()**

**否**

**是**

**是**

**否**

**i++**

**开始**

**结束**

**i=0**

**List[i]<list[j]**

**j < len - 1 - i**

**i<len**

**sort()**

**否**

**是**

**j=0**

**否**

**是**

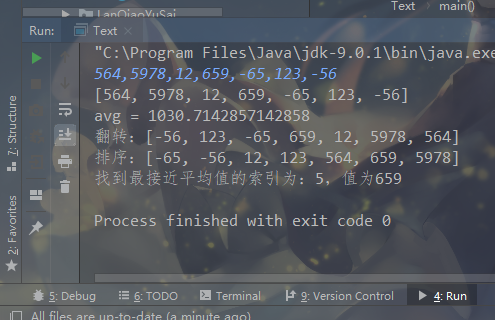
**否**

**交换i和j的值**

**i++**

**j++**

## 1.3测试结果



# Task 2（25 points）

**(A)** Design a binary *ArithmeticExpression* class, which includes:

(1) Two double private data members named x and y, representing left and right operands respectively.

(2) A double private data member named *operator*, representing the operator.

(3) A constructor for creating objects based on concrete binary expressions. Where the operator is qualified as "+, -, \*, \" and the constructor throws an exception if the argument does not qualify.

(4) Accessor method of three members.

(5) *getResult()* method to evaluate the expression’s value. This method throws an exception when it encounters a division operation with a zero divisor.

**(B)** Write the main program, if the disk file named *out.txt* does not exist, then create the file, and create 100 binary arithmetic expression objects with random Numbers, and write the expression and operation results into the file; If the file exists, all expressions are read from the file and displayed in line.

**（A）**设计一个二元ArithmeticExpression类，其中包括：

（1）两个分别名为x和y的私有double成员，分别表示左边的数字和右边的数字。

（2）创建operator的私有double成员，代表运算符。

（3）创建具体二元表达式对象的构造函数。运算符被限定为“ +，-，\*，\” ，如果运算符不符合条件，构造函数将引发异常。

（4）三个成员的访问器。

（5）使用getResult（）方法表示表达式的值。 当该方法遇到除数为零的除法运算时，将引发异常。

**（B）**编写主程序，如果不存在名为out.txt的文件，则创建该文件，并写入100个随机的二元算术表达式，并将表达式和运算结果写入该文件； 如果文件存在，则从文件中读取所有表达式并在行中显示。

## 2.1题目思路

首先识别字符串，从头开始扫描，碰到符号后停止，将符号前的子串记录为x，右边的记为y，如果碰到其他符号抛出异常。

getResult()方法也要抛出异常被除数不能为0的异常。

主程序要判断文件是否存在，若不存在则要创建随机数，符号也要随机。

## 2.2源代码

### ArithmeticExpression.java

package java\_exam.second;

public class ArithmeticExpression {

private double x, y;

//1,2,3,4代表 + - \* /

private double operator;

public ArithmeticExpression(String s) throws Exception {

StringBuffer sb = new StringBuffer();

int i = 0;

char[] str = s.toCharArray();

for (; i < str.length; i++) {

if (Character.isDigit(str[i]) || str[i] == '.' || str[i] == 'E'

|| (i == 0 && str[i] == '-'))

sb.append(s.charAt(i));

else if (str[i] == '+') {

operator = 1;

break;

} else if (str[i] == '-') {

operator = 2;

break;

} else if (str[i] == '\*') {

operator = 3;

break;

} else if (str[i] == '/') {

operator = 4;

break;

} else throw new Exception("符号不为 + - \* /");

}

if (operator == 0) throw new Exception("未检测到符号");

else {

//录入x和y

x = Double.parseDouble(sb.toString());

y = Double.parseDouble(s.substring(i + 1));

}

}

public double getX() {

return x;

}

public double getY() {

return y;

}

public double getOperator() {

return operator;

}

public double getResult() throws Exception {

if (operator == 1) return x + y;

else if (operator == 2) return x - y;

else if (operator == 3) return x \* y;

if (y == 0) throw new Exception("被除数为0");

return x / y;

}

}

### 测试程序：Text.java

package java\_exam.second;

import java.io.File;

import java.io.PrintWriter;

import java.util.Random;

import java.util.Scanner;

public class Text {

public static void main(String[] args) throws Exception {

File f = new File("src\\java\_exam\\second\\out.txt");

//文件不存在

if (!f.exists()) {

//PrintWriter 若文件不存在自动创建新文件

PrintWriter output = new PrintWriter(f);

Random r = new Random();

for (int i = 0; i < 100; i++) {

//sb储存表达式

StringBuffer sb = new StringBuffer();

//r.nextDouble()是小数位，r.nextInt()是整数位

//生成x

sb.append(r.nextDouble()+r.nextInt());

int operator = r.nextInt(4) + 1;

//随机符号

if (operator == 1) sb.append('+');

if (operator == 2) sb.append('-');

if (operator == 3) sb.append('\*');

if (operator == 4) sb.append('/');

//生成y

sb.append(r.nextDouble()+r.nextInt());

//写入表达式

output.print(sb.toString());

ArithmeticExpression ae = new ArithmeticExpression(sb.toString());

//写入结果

output.println(" = "+ae.getResult());

}

output.close();

}else{

Scanner sc = new Scanner(f);

while (sc.hasNextLine()) {

System.out.println(sc.nextLine());

}

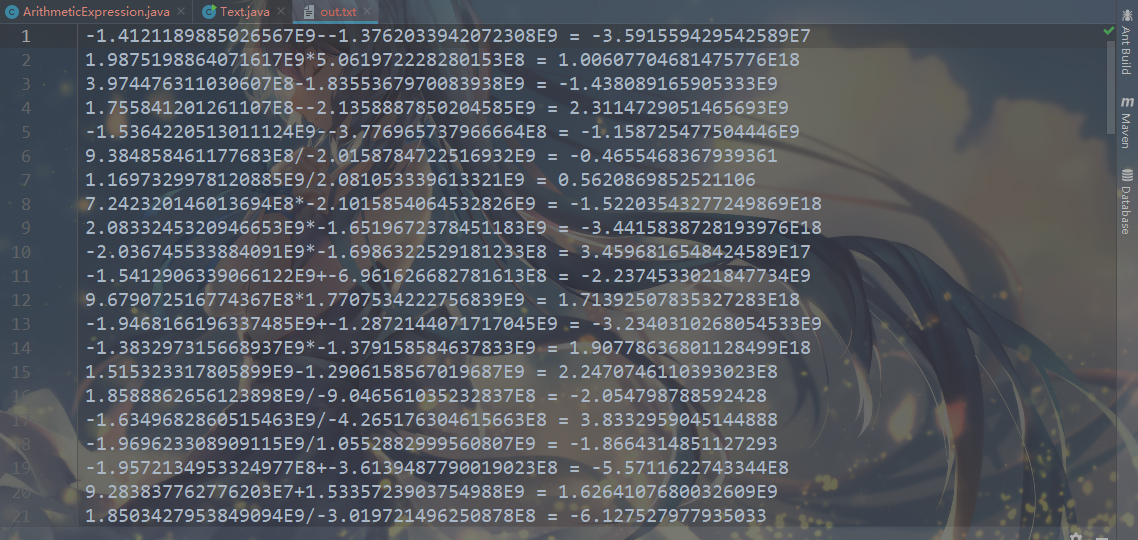
}

}

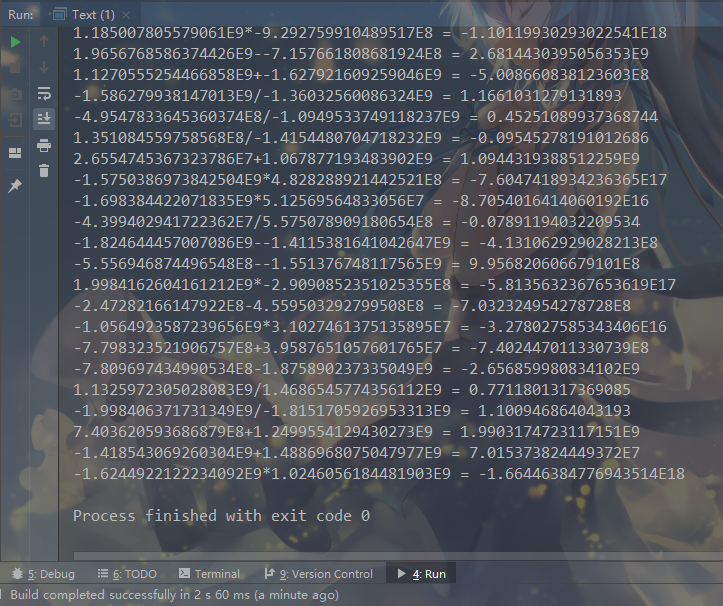
}

## 2.3测试结果

### out.txt



### 文件已存在



# Task 3（25 points）

**(A)** Design a class called *Policema*n, Which includes the following information:

(1) Three private property: police number, name and salary.

(2) A private property which is date type, and represents the date of work.

(3) The constructor to create the object according to the specified number, name and salary (the time when the object is created is the date of starting work).

(4) Accessors for all properties.

(5) Modifier of salary property.

(6) Modifier for date of work property.

(7) Override the inherited *toString()* method and output "Number + name + years of service + salary".

**(B)** Design an interface named *Employee* that contains an abstract method called *Raise ()*

**(C)** Design a class named *ArmedPolice*, inherit the police class and implement the employee interface, and include the following information:

(1) A String type property, representing the name of the army to which it belongs.

(2) A String type property , representing the rank.

(3) The constructor to create the object according to the specified number, name, salary, army and rank (the time to create the object is the date to start work).

(4) Realize the abstract method *raise* to increase the salary attribute by 30%.

(5) Override the inherited *toString()* method and output "Number + name + years of service + salary + army + rank".

**(D)** Draw the UML diagram of the above class structure.

**(E)** Compile the test program and create a police officer with the police number 10034, the name is "Jackie chan", the salary is 3000, the army is the 7th Brigade of the Armed Police, the rank is the third Police supervisor. Set the working time as April 1, 2004.Assume a raise every two years, modify his current salary according to the current system time, and output his personal information.

**（A）**设计一个 “Policeman”的类，其中包括以下信息：

（1）三个私有属性：警察编号，姓名和工资。

（2）日期类型的私有属性，代表工作日期。

（3）根据指定的编号，名称和工资（创建对象的时间为开始工作的日期）创建对象的构造函数。

（4）所有属性的访问器。

（5）薪金属性的修饰符。

（6）工作日期属性的修饰符。

（7）重写toString（）方法并输出“Number + name + years of service + salary”。

**（B）**设计一个Employee的接口，其中包含一个名为Raise（）的抽象方法

**（C）**设计一个名为ArmedPolice的类，继承Police类并实现Employee接口，并包括以下信息：

（1）一个String类型的属性，代表它所属的军队的名称。

（2）一个String类型的属性，表示等级。

（3）构造函数根据指定的编号，姓名，薪水，军队和等级来创建对象（创建对象的时间为开始工作的日期）。

（4）实现抽象方法提高工资30％。

（5）重写toString（）方法并输出“Number + name + years of service + salary + army + rank”。

**（D）**绘制上述类结构的UML图。

**（E）**编写测试程序，并创建一个Policeman对象，其警察编号为10034，名称为“ Jackie chan”，薪水为3000，the 7th Brigade of the Armed Police，军衔是the third Police supervisor。 将工作时间设置为2004年4月1日。假设每两年加薪一次，请根据当前系统时间修改其当前薪水，并输出其个人信息。

## 3.1题目思路

创建Policeman，ArmedPolice对象，Employee接口，然后实现接口，实现继承关系。

主程序修改日期时要获取毫秒数，然后更新新的毫秒数。计算加薪次数时要换算成毫秒然后做除法求出次数，然后调用对应次数的Raise()方法。

## 3.2源代码，UML图

### ArmedPolice.java

package java\_exam.third;

public class ArmedPolice extends Policeman implements Employee {

private String armyName;

private String rank;

public ArmedPolice(int number, String name, double salary, String armyName, String rank) {

super(number, name, salary);

this.armyName = armyName;

this.rank = rank;

}

public String getArmyName() {

return armyName;

}

public void setArmyName(String armyName) {

this.armyName = armyName;

}

public String getRank() {

return rank;

}

public void setRank(String rank) {

this.rank = rank;

}

@Override

public void Raise() {

setSalary(getSalary() \* 1.3);

}

@Override

public String toString() {

return super.toString() +

"\narmyName: " + getArmyName() +

"\nrank: " + getRank();

}

}

### Employee.java

package java\_exam.third;

public interface Employee {

void Raise();

}

### Policeman.java

package java\_exam.third;

import java.text.SimpleDateFormat;

import java.util.Date;

public class Policeman {

private int policeNumber;

private String name;

private double salary;

private Date date;

public Policeman(int policeNumber, String name, double salary) {

this.policeNumber = policeNumber;

this.name = name;

this.salary = salary;

date = new Date();

}

public int getPoliceNumber() {

return policeNumber;

}

public String getName() {

return name;

}

public double getSalary() {

return salary;

}

public Date getDate() {

return date;

}

public void setPoliceNumber(int number) {

this.policeNumber = number;

}

public void setName(String name) {

this.name = name;

}

public void setSalary(double salary) {

this.salary = salary;

}

public void setDate(Date date) {

this.date = date;

}

@Override

public String toString() {

//SimpleDateFormat格式化Date格式输出，空参默认当前所在时区

SimpleDateFormat sdf = new SimpleDateFormat();

return "policeNumber: " + policeNumber +

"\nname: " + name +

"\nsalary: " + salary +

"\ndate: " + sdf.format(date);

}

}

### 测试程序：Text.java

package java\_exam.third;

import java.util.GregorianCalendar;

public class Text {

public static void main(String[] args) {

//创建对象

ArmedPolice ap = new ArmedPolice(10034, "Jackie chan", 3000, "the 7th Brigade of the Armed Police", "the third Police supervisor");

//格式化日期,获取日期的毫秒值,并修改日期

GregorianCalendar gc = new GregorianCalendar(2004, 1, 1);

ap.setDate(gc.getTime());

//首先算现在的秒数减去工作日期的秒数

//然后除以2年的秒数，计算修改几次工资

long count = (System.currentTimeMillis() - ap.getDate().getTime()) / (63072000000L);

for (long i = 0; i < count; i++)

ap.Raise();

System.out.println(ap);

}

}

<interface>

Employee

+ Raise () : void

ArmedPolice

- armyName : string

- rank : string

+ ArmedPolice ()

+ ArmedPolice (number: int,

name: String , salary : double, armyName : string , rank : string)

+ Raise () : void

+ toString () : String

Policeman

- policeNumber : int

- name: String

- date: Date

- salary: Date

+ Policeman ()

+ Policeman (policeNumber : int,

name : String , salary : double)

+ getName () : String

+ setName (name: String) : void

+ getSalary () : double

+ setSalary (salary: double) : void

+ getNumber () : int

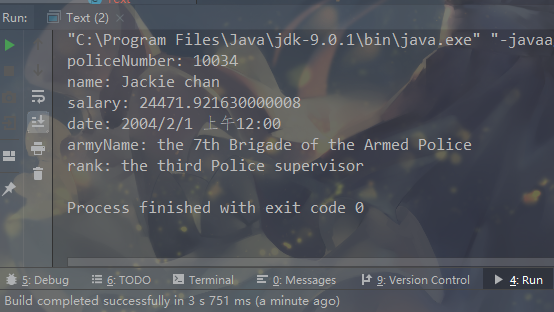
+ setNumber (number: int) : void

+ getDate () : Date

+ setDate (Date: date) : void

+ toString () : String

## 3.3测试结果



**Task 4**（**25 points**）

Combining the C language and Java language we have learned, Please summarize the differences between object-oriented programming idea and process-oriented programming idea. (You can use examples to illustrate your idea, and the total number of words between 300 and 500 words. Warning: If you direct copy of the content on Baidu then you will get 0 points)

结合我们所学的C语言和Java语言，请总结一下面向对象的编程思想和面向过程的编程思想之间的区别。 （您可以使用示例来说明您的想法，以及单词总数在300到500个单词之间。警告：如果您直接在百度上复制内容，则将获得0分）

以下内容如有纰漏，请老师指正。

经过了两年的学习，我了解了面向对象的编程思想和面向过程的编程思想。C语言使用的是面向过程的编程思想，将一个问题拆分成一个个小步骤，然后一次实现。而Java语言使用的是面向对象的编程思想，引入了权限修饰符等，将事物包装成一个对象，无论是具体事务，还是抽象事物都可以封装为对象，就想异常一样，它也是一个对象。解决问题的人不需要关心步骤是什么，只需要使用对象的方法就可以。用继承来表示相同类型的事物的共有特征，并且子类可以在父类基础上出现新的特征，用接口来实现不同事物间的相同特征，丰富了编程的可创造性。

当然面向对象的编程思想也有面向过程的编程思想，不如说后者是基础，前者拓宽了后者。两种思想各有优点。而我觉得Java的最大好处是，简化了许多繁琐的过程，有许多的api可以使用，并且简化了好多的写法。就比如使用动态数组时，在C语言只能在程序开始时创建好长度，或是在之后使用指针动态分配内存实现动态数组，而在Java中就避免了这种情况，既可以在任何地方创建定长数组，还可以使用list的实现类来创建动态数组，还可以使用不重复元素的set，和引入键值对的map。