* **厦门大学**

**Java上机实验**

**（第三次四个实验）**

**姓名：刘馨欣**

**学号：22920132203820**

**班级：计算机1班**

**系别：计算机**

**学院：信息科学与技术学院**

**实验时间：2015年04月17日**

**报告完成时间：2015年04月25日**

**实验一 三角形类**

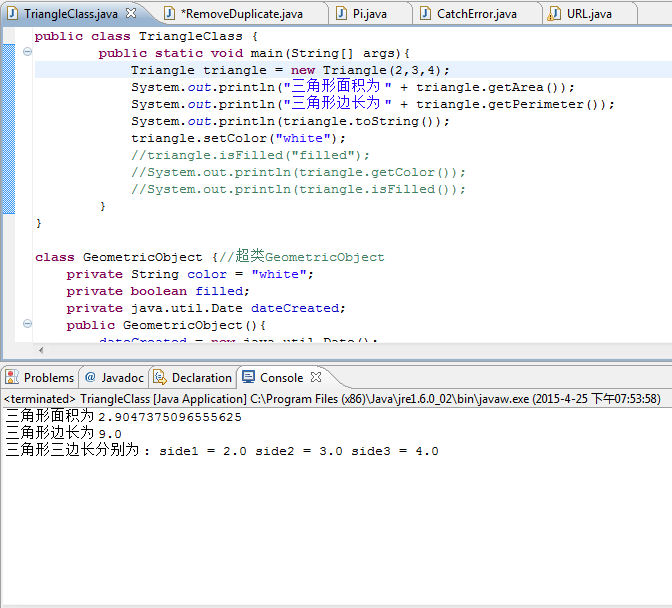
**（Triangle Class）**

**一、实验题目**

从课本的**GeometricObject**类继承一个**Triangle** 类，并满足下列条件：

* + 有3个**私有double**成员表示三条边，默认长度为1.0；
  + 有1个无参构造方法，用于创建默认三角形；
  + 有1个3参数构造方法，用于创建指定三角形，不需要考虑3边非法的情况；
  + 有1个getArea()，1个 getPerimeter() 分别返回三角形的面积和周长；
  + 有1个toString()方法返回三条边的信息，形如 "Triangle: side1 = " + side1 + " side2 = " + side2 + " side3 = " + side3;
  + 写一个测试类，测试Triangle类，并且需要设置该类继承来的**color**和**filled** 属性。

**二、实验结果**



**三、实验设计思路**

本实验重点在于类的继承！！！（其实三边已给side1等命名。。。不用再命名成abc等。。。。。。）

**四、实验收获**

1.学会编写类的继承；

2.复习方法的构造。

**五、源代码**

**public** **class** TriangleClass {

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

Triangle triangle = **new** Triangle(2,3,4);

System.*out*.println("三角形面积为 " + triangle.getArea());

System.*out*.println("三角形边长为 " + triangle.getPerimeter());

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

triangle.setColor("white");

//triangle.isFilled("filled");

//System.out.println(triangle.getColor());

//System.out.println(triangle.isFilled());

}

}

**class** GeometricObject {//超类GeometricObject

**private** String color = "white";

**private** **boolean** filled;

**private** java.util.Date dateCreated;

**public** GeometricObject(){

dateCreated = **new** java.util.Date();

}

**public** String getcolor(){

**return** color;

}

**public** **void** setColor(String color){

**this**.color = color;

}

**public** **boolean** isfilled(){

**return** filled;

}

**public** **void** setFilled(**boolean** filled){

**this**.filled = filled;

}

**public** java.util.Date getDateCreated(){

**return** dateCreated;

}

**public** String toString(){

**return** "created on " + dateCreated + "\ncolor:" + color + " and filled " + filled;

}

}

**class** Triangle **extends** GeometricObject {//默认长度1.0

**private** **double** a = 1.0;

**private** **double** b = 1.0;

**private** **double** c = 1.0;

**public** Triangle(){

}

**public** Triangle(**double** a,**double** b,**double** c){

**this**.a = a; **this**.b = b; **this**.c = c;

}

**public** **double** getArea(){//海伦公式

**double** p = **this**.getPerimeter() / 2;

**double** q = p \* (p - **this**.a) \* (p - **this**.b) \* (p - **this**.c);

**double** s = Math.*sqrt*(q);

**return** s;

}

**public** **double** getPerimeter(){

**return** **this**.a + **this**.b + **this**.c;

}

**public** String toString(){

**return** ("三角形三边长分别为 : side1 = " + a + " side2 = " + b + " side3 = " + c);

}

}

**实验二 删除重复数字**

**（Remove Duplicate）**

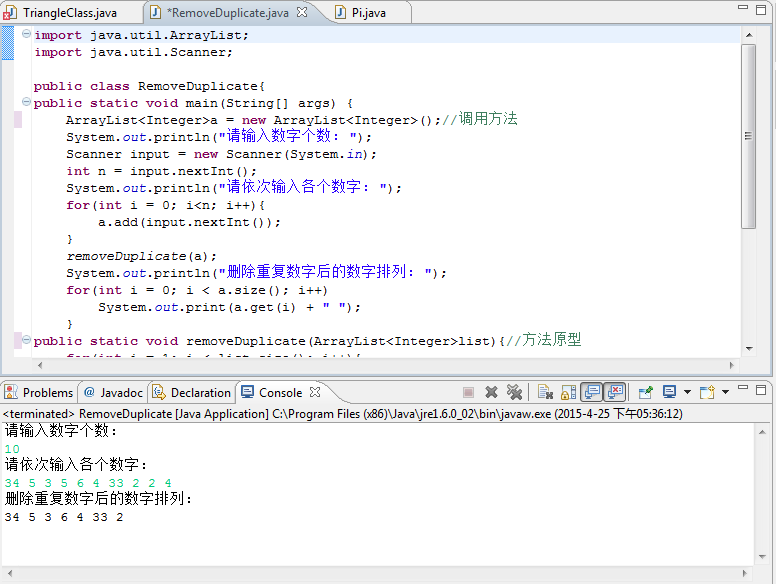
**一、实验题目**

* 写一个方法，能够删除ArrayList中的重复数字，其中数字的类型是Integer。要求**在原ArrayList中直接删除**重复数字。方法原型如下：

**public static void** removeDuplicate(ArrayList<Integer> list)

* 在main方法中调用上述方法，输出删除后的结果。例如某次运行结果如下：
  + 输入：34 5 3 5 6 4 33 2 2 4
  + 输出：34 5 3 6 4 33 2

**二、实验结果**



**三、实验设计思路**

本实验难点在于删除操作的算法设计。

**四、实验收获**

1.复习Java中方法调用；

2.成功设计删除重复数字的算法。

**五、源代码**

**import** java.util.ArrayList;

**import** java.util.Scanner;

**public** **class** RemoveDuplicate{

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

ArrayList<Integer>a = **new** ArrayList<Integer>();//调用方法

System.*out*.println("请输入数字个数： ");

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

**int** n = input.nextInt();

System.*out*.println("请依次输入各个数字： ");

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

a.add(input.nextInt());

}

*removeDuplicate*(a);

System.*out*.println("删除重复数字后的数字排列： ");

**for**(**int** i = 0; i < a.size(); i++)

System.*out*.print(a.get(i) + " ");

}

**public** **static** **void** removeDuplicate(ArrayList<Integer>list){//方法原型

**for**(**int** i = 1; i < list.size(); i++){

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

**if**(list.get(i) == list.get(j)){

list.remove(i);

i--;

}

}

}

}

}

**实验三 异常捕捉**

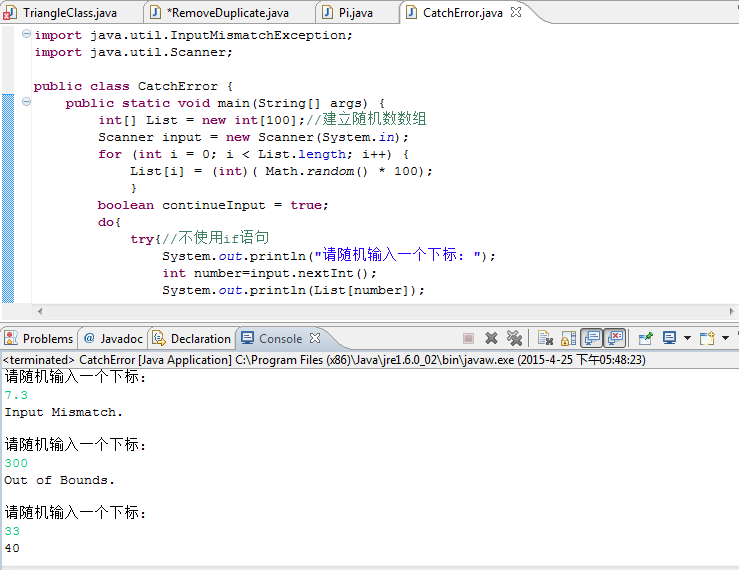
**（Catch Error）**

**一、实验题目**

写一个程序，练习捕捉异常。

* 1. 首先创建一个100个元素的数组，并用随机数填充该数组，类型随意。
  2. 提示用户输入一个下标，输出该下标对应的元素值。如果输入不是整数，输出“**Input Mismatch.**”并提示用户重新输入；如果下标越界，输出“**Out of Bounds.**”并提示用户重新输入；如果不是以上情况，程序输出元素值后正常结束。
  3. 代码不允许使用if语句，只能用异常捕捉实现上述功能。

**二、实验结果**

****

**三、实验设计思路**

参考课件PDF13中的例子进行编程，主要难点在于运用异常处理的方式。

**四、实验收获**

1.学习Java语言中异常处理的运用；

2.学习Java中随机数的生成。

**五、源代码**

**import** java.util.InputMismatchException;

**import** java.util.Scanner;

**public** **class** CatchError {

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

**int**[] List = **new** **int**[100];//建立随机数数组

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

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

List[i] = (**int**)( Math.*random*() \* 100);

}

**boolean** continueInput = **true**;

**do**{

**try**{//不使用if语句

System.*out*.println("请随机输入一个下标: ");

**int** number=input.nextInt();

System.*out*.println(List[number]);

continueInput = **false**;

}

**catch**(InputMismatchException ex){//输入不是整数

System.*out*.println("Input Mismatch.\n");

input.nextLine();

}

**catch**(IndexOutOfBoundsException ex){//数组越界

System.*out*.println("Out of Bounds.\n");

input.nextLine();

}

}**while**(continueInput);

}

}

**实验四 网页分析**

**（URL）**

**一、实验题目**

* 抓取厦大主页中的“快速链接”下的所有内容。
* 提示，“快速链接”部分的HTML形如：

<li><a href="http://e.weibo.com/xmunews" target="\_blank"><i class="icon icon-list-alt"></i>官方微博</a></li>

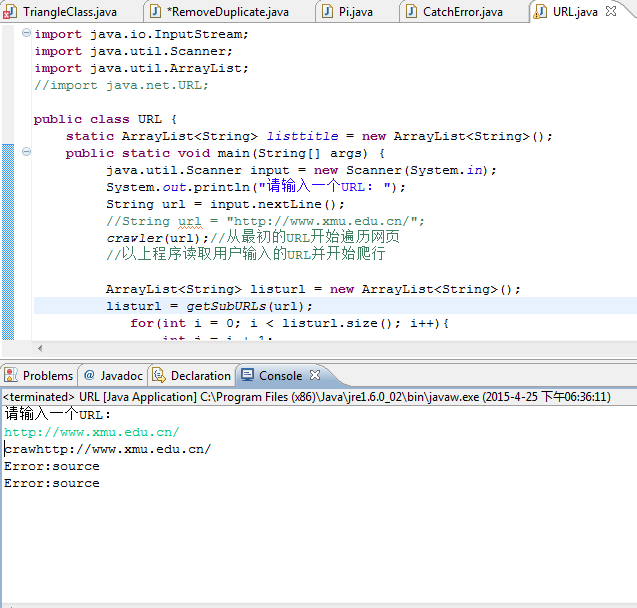
* 程序输出为：[序号] 标题 URL ，每行一个链接，如：  
  [1]官方微博 <http://e.weibo.com/xmunews/>

[2]易班厦大 <http://yiban.xmu.edu.cn/>

[3]厦大网址导航 <http://123.xmu.edu.cn/>

……

**二、实验结果**

****

**三、实验设计思路**

参考PDF12中网页爬虫的标准程序进行改编。

**四、实验收获**

复习网络爬虫程序。。。分析此程序。。。有难度！！！

**五、源代码**

**import** java.io.InputStream;

**import** java.util.Scanner;

**import** java.util.ArrayList;

//import java.net.URL;

**public** **class** URL {

**static** ArrayList<String> *listtitle* = **new** ArrayList<String>();

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

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

System.*out*.println("请输入一个URL： ");

String url = input.nextLine();

//String url = "http://www.xmu.edu.cn/";

*crawler*(url);//从最初的URL开始遍历网页

//以上程序读取用户输入的URL并开始爬行

ArrayList<String> listurl = **new** ArrayList<String>();

listurl = *getSubURLs*(url);

**for**(**int** i = 0; i < listurl.size(); i++){

**int** j = i + 1;

System.*out*.println("["+j+"]"+*listtitle*.get(i)+" "+listurl.get(i));

}

}

**public** **static** **void** crawler(String staringURL){

ArrayList<String> listOfPendingURLs = **new** ArrayList<String>();

ArrayList<String> listOfTraversedURLs = **new** ArrayList<String>();

listOfPendingURLs.add(staringURL);

**int** count = 1;

**while**(!listOfPendingURLs.isEmpty()&&listOfTraversedURLs.size() <= 100){

String urlString = listOfPendingURLs.remove(0);

**if**(!listOfTraversedURLs.contains(urlString)){

listOfTraversedURLs.add(urlString);

System.*out*.println("craw" +urlString);

**for**(String s:*getSubURLs*(urlString)){

**if**(!listOfTraversedURLs.contains(s)){

listOfPendingURLs.add(s);

}

}

}

}

}

//以上程序为网络爬虫算法的具体实现

**public** **static** ArrayList<String> getSubURLs(String urlstring) {

ArrayList<String> list = **new** ArrayList<String>();

**try**{

java.net.URL url = **new** java.net.URL(urlstring);

URL url1 = **new** URL();

Scanner input = **new** Scanner(url1.openStream(),"UTF-8");

**int** current = 0;

**int** current1 = 0;

**while**(input.hasNext()){

String line = input.nextLine();

current = line.indexOf("http:",current);

**while**(current > 0){

**int** endIndex = line.indexOf("target=\"\_blank\"><i class=\"icon icon-list-alt\">",current)-2;

**if**(endIndex > 0){//确保能找到一个正确的URL

current1 = line.indexOf("</i>")+4;

**int** endindex1 = line.indexOf("</a></li>",current1);

list.add(line.substring(current,endIndex));

*listtitle*.add(line.substring(current1,endIndex));

current = line.indexOf("blank\">",endIndex);

}

**else**

current = -1;

}

}

}

**catch**(Exception ex){

System.*out*.println("Error:" +ex.getMessage());

}

**return** list;

}

**private** InputStream openStream() {

// **TODO** Auto-generated method stub

**return** **null**;

}

}

//以上程序为URL页面提取