《Java SE程序设计》

实 验 指 导 手 册

授课教师： 崔毅东

实验指导教师： 崔毅东

教学对象： 二年级本科生

开课时间： 秋季学期

北京邮电大学软件学院

2018年10月

# 实验二：Java程序编制（基础练习）

## 实验目的

学生通过使用Java语言进行基本程序的开发，练习继承、多态、接口、异常处理、多线程同步、Socket通信等。

## 实验内容（详见【二、实验内容说明】）

## 实验环境

1. Windows
2. Eclipse 或者 NetBean

## 实验要求

1. 独立完成实验内容要求
2. 熟练使用Java常用 IDE 进行编程
3. 上交源程序文件（纸质版或者电子版，以课程指导教师要求为准）
4. 上交实验报告（纸质版或者电子版，以课程指导教室要求为准。标准格式见附件二）

## 实验步骤

1. 启动Java常用 IDE。
2. 建立project。
3. 编辑源程序。
4. 编译、链接并执行源程序，看结果是否正确。
5. 如果报错或告警，做必要修改，重复3－5步骤直到没有错误和告警。

# 实验内容说明

本次实验一共4个Project。

## Project #1 Design Classes with Inheritance

CSCI 1302 Introduction to Programming

Armstrong Atlantic State University

### Design a class named Triangle that extends GeometricObject.

The class contains:

1. Three double data fields named side1, side2, and side3 with default values 1.0 to denote three sides of the triangle.
2. A no-arg constructor that creates a default triangle.
3. A constructor that creates a rectangle with the specified side1, side2, and side3.
4. The accessor methods for all three data fields.
5. A method named getArea() that returns the area of this triangle.
6. A method named getPerimeter() that returns the perimeter of this triangle.
7. A method named toString() that returns a string description for the triangle.

For the formula to compute the area of a triangle, see Exercise 5.19. The toString() method is implemented as follows:

return "Triangle: side1 = " + side1 + " side2 = " + side2 + " side3 = " + side3;

### Draw UML diagram

Draw the UML diagram that involving the classes Triangle and GeometricObject.

### Implement and Test

Implement the class. Write a test program that creates a Triangle object with sides 1, 1.5, 1, setting color **yellow** and filled **true**, and displaying the **area, perimeter, color, and whether filled or not**.

### Submissions

What to submit?

The source code and the screen shot of a sample run.

## Project #2 Design Classes with Abstract Classes and Interfaces

CSCI 1302 Introduction to Programming

Armstrong Atlantic State University

### Description

This project consists of two separate problems. For each, print the source code and the screen shot of a sample run.

### Problem 1

Design an interface named Colorable with a void method named **howToColor()(输出一个字符串)**. Every class of a colorable object must implement the Colorable interface. Design a class named **Square** that extends GeometricObject and implements Colorable. Implement howToColor to display a message on how to color the square.

Draw a UML diagram that involves Colorable, Square, and GeometricObject.

Write a test program that creates an array of five **GeometricObject**. For each object in the array, invoke its howToColor method if it is colorable.

### Problem 2

Develop a class named Octagon that extends GeometricObject and implements the **Comparable** and **Cloneable** interfaces. Assume that all eight sides of the octagon are of equal size. The area can be computed using the following formula:



Draw the UML diagram that involves Octagon, GeometricObject, Comparable, and Cloneable.

Write a test program that creates an Octagon object with side **value 5** and displays its area and perimeter. Create a new object **using the clone method** and compare the two objects **using the compareTo method**.

## Project #3 Multi-Threading programming

Develop a multi-threaded program.

join()

wait() notifyAll()

1. There are **three threads** in the program.
2. Thread A and Thread B **sleep a random time** (no more than 1 seconds) respectively and then each produces a random character (limited in 'a'-'z');
3. Thread C then compares the two characters produced by Thread A, and B. The thread producing bigger character (according to its ASCII code) will get 2 points and the thread producing smaller character will get 0 point. If two characters are equal, then each thread get 1 point.
4. Thread C prints the result in the following format. You can use either console or GUI as the program’s UI.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Round | **Thread A** | | | **Thread B** | | |
| Sleep time | Random character | Points obtained | Sleep time | Random character | Points obtained |
| 1 |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |
| … |  |  |  |  |  |  |
| N |  |  |  |  |  |  |

Final result: A (or B or None) is the winner

(Hint: You do NOT need to print the table frame)

1. Repeat step 2 - step 4 **at least 3 times** (which means N is not smaller than 3)
2. Finally, Thread C compares the total points obtained by thread A and thread B respectively to declare who is the winner: A (or B or None) is the winner.
3. At least one of the three threads must inherits from **Thread class**; And at least one of the three threads must implement **Runnable interface**.
4. You are NOT allowed to use the default main thread as one of your 3 threads. That is to say, you MUST explicitly create 3 threads.
5. Maybe you need to use one or more methods such as: sleep(); wait(); notify(); notifyall(); join(); etc.

改一下, 将ThreadA和ThreadB放到其他文件中

## Project #4 Socket programming

This project converts [Project #3 Multi-Threading programming] into a client-server project. You can make a copy of your Project #3 and then modify the codes to fit the requirements of project #4.

### Description

The game "rock paper scissors" is a popular children's game using three hand signs representing "rock", "paper" and "scissors", frequently used as a tiebreaker.

The details of the game can be found at <https://en.wikipedia.org/wiki/Rock-paper-scissors>.

The rules is shown in the following table:

Table 1. "rock paper scissors" score table

|  |  |  |  |
| --- | --- | --- | --- |
| Kid A | Kid B | Kid A Score | Kid B Score |
| rock | paper | 0 | 2 |
| rock | scissors | 2 | 0 |
| paper | scissors | 2 | 0 |
| paper | rock | 2 | 0 |
| scissors | rock | 0 | 2 |
| scissors | paper | 2 | 0 |
| rock | rock | 1 | 1 |
| paper | paper | 1 | 1 |
| scissors | scissors | 1 | 1 |

### Design and Develop

Develop **two socket programs**: one server, one client.

1. The **client has two threads** (You can deem each thread as a kid):
   1. Thread A (i.e. Kid A) and Thread B (i.e. Kid B) both sleep a random time (no more than 1 seconds) and then each produces a random selection among [rock, paper, scissors];
   2. Thread **A use TCP** to send data to server
   3. Thread **B use UDP** to send data to server
2. The server receives data from the client (the server acts as both TCP server and UDP server)
3. Once the server receives two selection made by Kid A and Kid B (one from TCP socket and another from UDP socket), the server compares the two selections (denoted by "Kid A" and "Kid B") and assign scores to the two kids according 【 Table 1. "rock paper scissors" score table】.
4. The server prints the result in the following format. You can use either console or GUI as the program’s UI.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Round | **Thread A** | | | **Thread B** | | |
| Sleep time | Random selection | Points obtained | Sleep time | Random selection | Points obtained |
| 1 | 250ms | rock | 1 | 413ms | rock | 1 |
| 2 | 721ms | scissors | 2 | 54ms | paper | 0 |
| … |  |  |  |  |  |  |
| N |  |  |  |  |  |  |

Final result: A (or B or None) is the winner

(Hint: You do NOT need to print the table frame)

1. Repeat step 1 - step 4 at least 3 times.
2. Finally, the server compares the total points obtained by thread A and thread B respectively to declare who is the winner: A (or B or None) is the winner.

# 附件二：

实验报告模版，参见Word文档《实验报告模版》