实验二（第2次上机）：Pair Project

**一、实验目标：**

1）体验敏捷开发中的两人合作。

2）进一步提高个人编程技巧与实践。

**二 、实验要求：**

1）根据子数组问题，练习结对编程实践；

2）要求学生两人一组，自由组合。二人共同编码，两人的角色至少切换4次，完成实验要求：

**四则运算问题说明：**

任何编程语言都可以，命令行程序接受一个数字输入，然后输出相应数目的四则运算题目和答案。例如输入数字是 30， 那就输出 30 道题目和答案。 运算式子必须至少有两个运算符，运算数字是在 100 之内的正整数，答案不能是负数。

**问题扩展：**

一次可以出一千道道题目，并且没有重复的，把题目写入一个文件中

支持多个运算符，最多多 10 个运算符，括号的数量不限制

**三、实验内容：**

**1、代码规范：**

实验采用基于Python编程，需要调用库中大量的包，编写Python语言的代码规范有：

命名规范：

　　变量和函数为小驼峰法标识, 即除第一个单词之外，其他单词首字母大写

　　全局变量为大写

　　常量 为大写

对象定义的规则：

　　将左花括号与类名放在同一行。

　　冒号与属性值间有个空格。

　　字符串使用双引号，数字不需要。

　　最后一个属性-值对后面不要添加逗号。

将右花括号独立放在一行，并以分号作为结束符号。

2.程序的总体设计

四则运算加减乘除，采用两个随机数，由于不能出现负数，所以在对两个随机数进行减法运算的时候，需要进行比较大小，而除法在运算中，除数不能取0。

定义函数：用def szys()实现随机生成四则运算，用def test()通过调用syzs()实现题库的制作.

所以代码分为三个部分，第一部分是生成随机四则运算，最后输出算式并返回正确答案；第二部是用户通过输入一个整数来输出所需要的算式，当输出的算式达到目标数量后，便一一输出答案；第三部分为主函数，需要用户自己选择模式，当输入是‘1’时，是为四则运算，通过调用syzs()函数得到算式和返回值，用户输入答案后，便与用户输入值进行比较。当输入是‘2’时，是生成题库，并在最后输出答案。

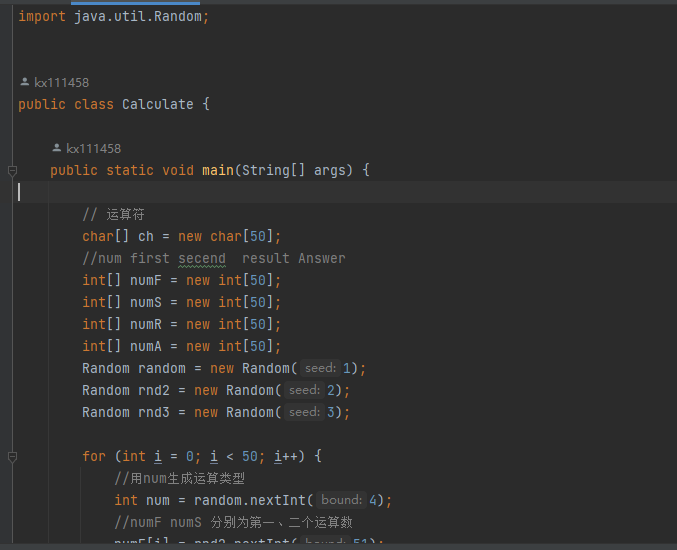
 3．结对编程过程

在编程之前，我们准备使用pycharm进行编程，因为pycharm是当下最流行的编译器，并且编译方便

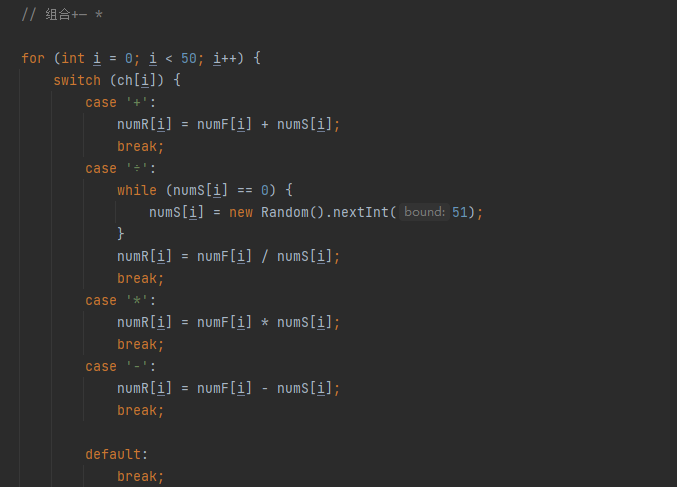
我们在结对编程过程中，坐在教室使用同一台电脑进行讨论，代码编写测试，目前来看是最直接的能够了解彼此编程进程和解决当下问题的方式。

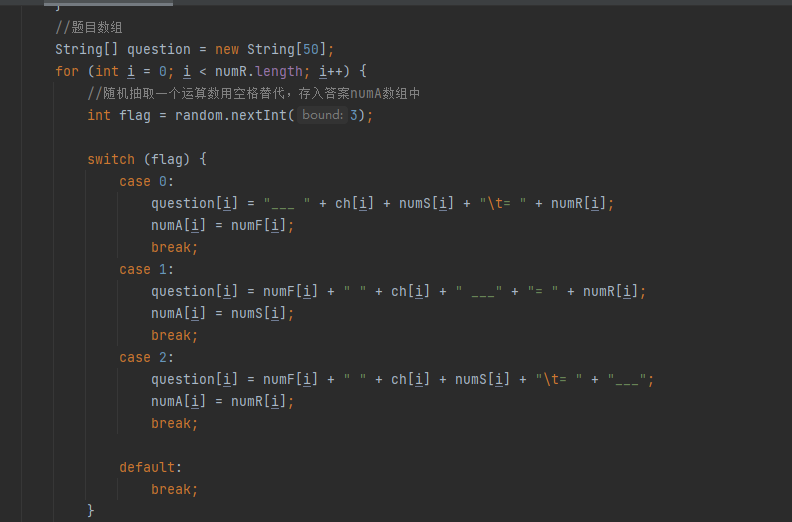
**四、代码**

四则运算函数

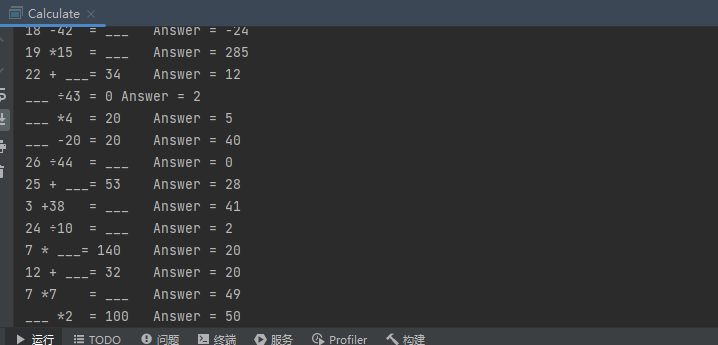


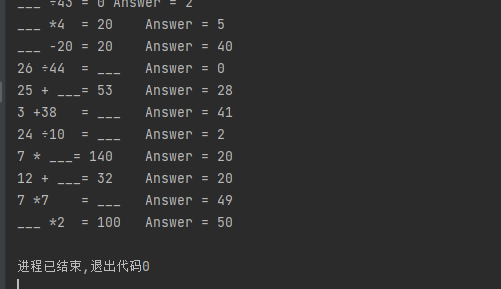






运行结果：





参考资料：

1、 四则运算：软件工程练习：模块化，单元测试，回归测试，TDD - SoftwareTeacher - 博客园 (cnblogs.com)；

现代软件工程讲义 个人项目和结对项目练习 四则运算 - SoftwareTeacher - 博客园 (cnblogs.com)

2、结对编程课程实践：https://www.likecs.com/show-700440.html

PSP评估表：

|  |  |  |  |
| --- | --- | --- | --- |
| Personal Software Process Stages | 个人软件流程阶段 | 估计的时间（分钟） | 实际花费的时间 (分钟) |
| **Planning** | **计划-把工作细化并大致安排次序** | 20 | 15 |
| **Development** | **开发** | 120 | 200 |
| ·         Analysis | ·         需求分析  (包括学习新技术) | 5 | 7 |
| ·         Design Spec | ·         生成设计文档 | / | / |
| ·         Design Review | ·         设计复审  (和同事审核设计文档) | / | / |
| ·         Coding Standard | ·         代码规范   (制定合适的规范) | 5 | 8 |
| ·         Design | ·         具体设计 | 8 | 12 |
| ·         Coding | ·         具体编码 | 35 | 70 |
| ·         Code Review | ·         代码复审 | 20 | 30 |
| ·         Test | ·         测试（自我测试，修改代码，提交修改） | 25 | 20 |
| **Reporting** | **总结报告** | 10 | 15 |
| ·         Test Report | ·         测试报告 | 6 | 5 |
| ·         Size Measurement | ·         计算工作量 | 5 | 5 |
| ·         Postmortem & Improvement Plan | ·         事后总结, 并提出改进 | / | / |
| **Total** | **总计** | 279 | 407 |

代码复审核查表：

1．概要部分

（1）代码能符合需求和规格说明么？能，用户可以输入待抽签的号码集合

（2）代码设计是否有周全的考虑？否 ，还有一些没有考虑到的，会继续完善的

（3）代码容易维护么？比较容易维护

（4）代码的每一行都执行并检查过了吗？是的，都可以执行

2．设计规范部分

（1）设计是否遵从已知的设计模式或项目中常用的模式？是的，因为比较熟悉，更容易操作

（2）代码有没有依赖于某一平台，是否会影响将来的移植（如Win32到Win64）？没有，不会影响移植，任何平台都可以

（3）开发者新写的代码能否用已有的Library/SDK/Framework中的功能实现？在本项目中是否存在类似的功能可以调用而不用全部重新实现？

存在，有些代码是可以调用的

（4）有没有无用的代码可以清除？没有，已经全部清除完毕了

3．代码规范部分

（1）修改的部分符合代码标准和风格么?符合，全都按照代码标准修改的

4．具体代码部分

（1）有没有对错误进行处理？对于调用的外部函数，是否检查了返回值或处理了异常？对错误都进行了处理，没有异常

（2）参数传递有无错误，字符串的长度是字节的长度还是字符（可能是单/双字节）的长度，是以0开始计数还是以1开始计数？是字节的长度，以1开始计数的

（3）循环有没有可能出现死循环？没有可能，可以重复抽取

（4）有没有可能导致资源泄露（内存、文件、各种GUI资源、数据库访问的连接，等等）？有没有可能优化？不会导致资源泄露，可以优化使用断言来保证我们认为不变的条件

5．效能

（1）代码的效能（Performance）如何？达到了具体任务要求的程度

（2）代码中，特别是循环中是否有明显可优化的部分（C++中反复创建类，C#中 string 的操作是否能用StringBuilder 来优化）？可以用StringBuilder 来优化，当我们需要多次的对一个字符串进行多次操作的时候，他的效率要远远高与string

（3）对于系统和网络调用是否会超时？如何处理?目前没有出现超时的现象。假如出现了我们会 杀毒；整理系统，减少运行的进程，释放内存、cpu，释放c盘空间；

6．可读性

代码可读性如何？有没有足够的注释？可以读取，注释都是最新的，只是有部分没有测试

7．可测试性

代码是否需要更新或创建新的单元测试？暂时不需要