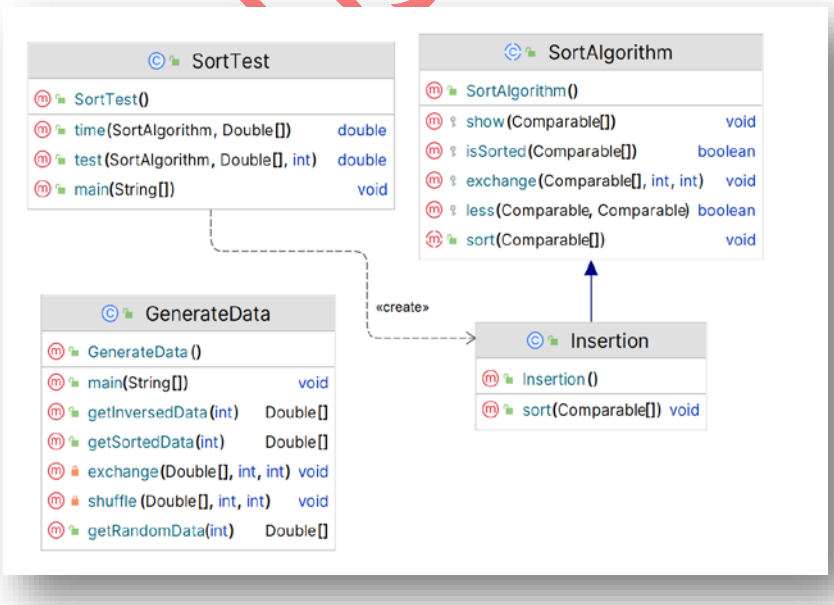
实验 1：渐进分析和排序算法

本次实验内容包括两个方面的内容，一个是对课堂上所讲授的渐进分析加强认知和理解；一个是对若干个排序算法进行实践与比较。

排序是数据处理中经常使用的一种很重要的运算，人们对它进行了深入细致的研究，并已设计出了一些巧妙的算法。但是，仍然有一些与排序相关的问题尚未解决，适应各种不同要求的新算法也不断被开发出来并得到了改进。在面对具体的问题，如何进行分析，并选择较合适的排序算法，这个能力的培养前提就是先要学习已经很成熟的若干个内排序算法并且清楚知道每个排序算法的优缺点。这个实验的目的就是了解排序算法之间的差异性。

排序算法实验内容涉及如下排序算法：Insertion、Selection、Shell、Quicksort 和 Mergesort。

实验中提供如下的类：SortAlgorithm[1]、Insertion、SortTest 和 GenerateData，这些类之间的关系如下图所示：



其中：Insertion 是已经实现了插入排序算法的类类型，参照这种方式，实现其他四种排序算法的类类型，这样就很容易在 SortTest 中使用多态完成对不同排序算法的测试。GenerateData 是一个生成测试数据的类类型（相当于工具类），目前该类型中提供三种形式的测试数据：1）均匀分布的随机数据；2）正序数据序列；3）逆序数据序列。

之所以提供这些类的设计，目的是让同学们能将精力放在对排序算法的实现中，当然也可以借此将之前所学的面向对象编程技术进行实践。请同学们按序完成本次实验的任务。

[1] SortAlgorithm 的子类类型实现的各种排序算法所比较的数据都是 Comparable 接口类型，这样做的目的就是让这些算法能够对所有实现了 Comparable 接口的类类型数据排序，而不受限于某一个基础数据类型。在示例代码中的 Double 类型是实现了 Comparable 接口的类型。

任务 1 证明

1） 使用 O、Ω和Θ的定义，证明下面每一个等式：

a) $2\sqrt{n}+6=O(\sqrt{n})$

b) $n^2=\Omega(n)$

c) $log_2(n)=\Theta(\ln(n))$

d) $4^n \neq O(2^n)$

2） 使用数学归纳法证明$T(n)=O(n^2)$。T(n)的定义式如下：

$$T(n)=\begin{cases}T(n-1)+2n, & n>1\\ 3, & n=1\end{cases}$$

任务 2 设计算法要多思考

已知平面上有 n 个点，设计一个算法求平面内两点距离最小的两个点。朴素的算法是通过计算任意两个点之间的距离，经过统计给出距离最短的两个点，这时的时间复杂度是 $O(n^2)$，现要求完成的算法时间复杂度必须满足 O(nlogn)。（提示：使用分治算法设计思想）

不需要实现代码，但必须要给出算法描述，并给出算法的时间复杂度分析过程。

任务 3 排序算法的实现

参照 Insertion 类的实现方式，为其他四个排序算法实现相对应的类类型。这些类类型中有可能需要相配合的成员方法，请同学们灵活处理。针对 Shell 排序，需要分别实现三种不同间隔策略的递减序列（所以 Shell 排序将有三个，分别命名为 Shell1、Shell2 和 Shell3）：

① Shell 的最开始的间隔递减序列：1、2、4、…

② Hibbard 的间隔递减序列：1、3、7、…、2^k-1

③ Knuth 的间隔递减序列：1、4、13、…、(3^k-1) /2

要求：

- 每个排序算法使用课堂上所讲授的步骤，不要对任何排序算法进行额外的优化；
- 对每个排序算法执行排序之后的数据可以调用 SortAlgorithm 类型中的成员方法 isSorted 进行测试，检查是否排序成功。

任务 4 排序算法性能测试和比较

完成对每一个排序算法在数据规模为：2^8、2^9、2^{10}、……、2^{16} 的均匀分布的随机数据序列、正序序列和逆序序列的排序时间统计。

要求：

- 在同等规模的数据量和数据分布相同下，要做 T 次运行测试，用平均值做为此次测试的结果，用以排除因数据的不同和机器运行当前的状态等因素造成的干扰；（在 SortTest 类型

的 test 方法参数中有对每次数据规模下的测试次数的指定）

- 将所有排序算法的运行时间结果用图表的方式进行展示，X 轴代表数据规模，Y 轴代表运行时间。（如果因为算法之间运行时间差异过大而造成显示上的问题，可以通过将运行时间使用取对数的方式调整比例尺）
  - 注：运行时间会受到机器当时状态的干扰，可能出现不正常的时间趋势表现。如果出现类似现象，可以通过对交换次数与比较次数的统计间接代表算法的运行时间。（比较次数很明确，1 个交换次数可以用 3 个赋值语句进行换算）
- 对实验的结果进行总结：从一个算法的运行时间变化趋势和数据规模的变化角度，从同样的数据规模和相同数据分布下不同算法的时间相对差异上等角度进行阐述。

任务 5 快速排序的再探讨和应用

快速排序算法被誉为 20 世纪科学和工程领域的十大算法之一。前面的任务只是对快速排序的初识，下面从几个方面再更深入了解它：

① 探索快速排序的递归深度：为快速排序算法增加一个整数类型的参数 depth，每递归调用一次让 depth 加 1，每从递归中返回就让 depth 减 1，在整个执行过程中，depth 的最大值即为快速排序的递归深度，验证在平均情形下，depth 的值和规模 N 之间的关系；

② 优化快速排序：当待排序的数据量小于某个阈值时将递归的快速排序调用改为直接插入排序调用，按照这种策略的优化的快速排序算法参照任务 4 的要求进行测试，并与任务 4 中没有优化的快速排序算法的执行时间进行对比；

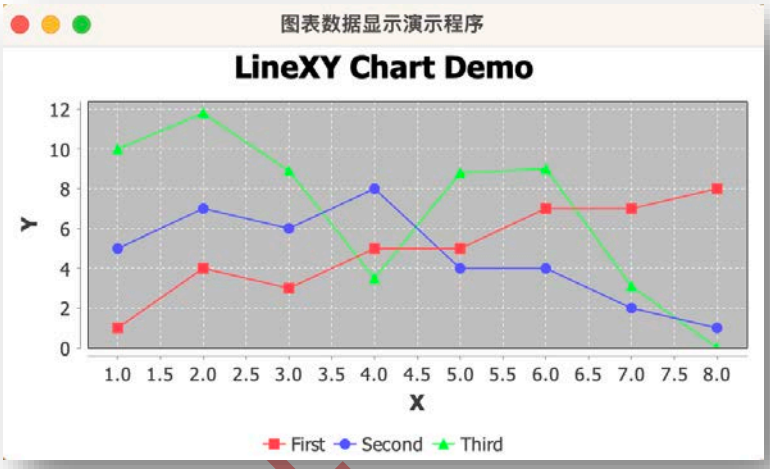
③ 螺母与螺丝的匹配问题：给定一堆螺丝和螺母，它们具有不同的尺寸，且螺丝和螺母之间存在一对一匹配关系，但不能直接比较螺丝或螺母的尺寸大小，只能通过给定的 match(nut bolt)函数进行比较，该函数返回 0 表示匹配，1 表示螺母尺寸大于螺钉尺寸，-1 表示螺母尺寸小于螺钉尺寸。设计一个算法实现对螺母与螺丝的匹配，给出算法的时间复杂度分析。可以通过设置两个数组 nuts 和 bolts， 数组中的内容是它们各自的大小（保证大小各不相同且 nuts 和 bolts 数组中的数据内容完全一致），编写程序测试你所设计的算法是否正确。

补充材料：

实验任务需要将实验结果用图表的方式完成展示，完成这样的功能有很多方式，下面将介绍使用 Java 语言完成此功能的方式。

JFreeChart 是一个开源的 Java 语言编写的图表生成框架。支持生成饼图、条形图、折线图等相关图表。官方的网址：http://jfree.org/jfreechart/，在随实验的附件中还有两个文件：jfreechart.jar 和 LineXYDemo.java。其中 jfreechart.jar 文件是能够创建 JFreeChart 图表的类库，需要在工程项目中将该 jar 包加入到引用路径中，具体方式因开发工具的不同而有所不同，请自行查阅加入方式；LineXYDemo 是一个可以运行的使用 JFreeChart 框架下生成的线型图表，在源代码中有详细的注释供参考。

提供这些补充材料有两个目的：一是希望同学们更熟练地掌握 Java 面向对象编程语言的使用；二是希望同学们将主要精力放在数据结构的实践任务中（毕竟在 LineXYDemo 中只要简单替换其中的数据，就可以显示图表数据了）。有兴趣的同学，可以在课余时间继续了解 JFreeChart。下图是 LineXYDemo 运行的结果：


图表数据显示演示程序
LineXY Chart Demo
12
10
8
6
4
2
0
Y
1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0 5.5 6.0 6.5 7.0 7.5 8.0
X
First
Second
Third