Strategy 说明

作者——张鸣昊（Debug 过程：三人共同完成）

设计思路：面对爬虫爬取的冗杂的信息，想要处理清晰并且存入需要的数据基类中便需要很复杂的算法过程（主要是字符串处理）。因此，作者考虑采用策略模式的思路，实际上只是把各个目标字符串所需的算法过程整理到对应的 strategy 里，这样会使 BaseObject 调用时，只需要一个 BaseStrategy 指针，大大减少了代码冗杂程度。

结构：与 Catcher 相似，一个 Base 基类对应三个网站的三个不同目标（排行榜特殊处理，见 Catcher 说明），共九个派生类。
工作过程：每个派生类使用时，都会在 exec（执行）函数里，先调用 BaseCatcher 对应的 MakeCatcher 函数爬取信息，再用 SaveinObject 函数将信息读入 ifstream (readfile) 中，然后根据情况不同，定义了多个局部变量，临时用来储存分割得到的信息，准备存进 Basedata 中。当处理完后，自动清空 txt 内容，便于下次使用。这是，Strategy 会 new 出一系列 BaseData*来读取临时变量信息，然后 push_back 进需要进入的 BaseObject 的 vector 中。其中 complexData 用来存“可以进一步爬取（在用户界面上可以点击）”的信息，simpleData 用来储存仅仅可以显示的信息。以上就是 Strategy 工作流程。

说明：
1. Strategy 可以说是建立在 Catcher 和 Object 之间的重要桥梁，采用的设计模式作者认为较为恰当。虽然原理简单，但是有着不可忽视的作用。
2. Strategy 内的代码虽然不易看懂，但是其工作极为简单，即分割字符串成为一个个需要的数据，作者认为与 py 文件一样，不必过于在意具体如何实现，只需要理解其功能，便可以清晰理解整个程序框架。
3. 由于网页信息格式多变，在测试程序中出现了许多由于网页源代码在同一处却表现不同的情况，作者尽可能的设计处理过程，增加判断条件，以防出现爬取不到的情况。但是，由于测试样例数量有限，可能有我们没有发现的特殊之处，这也是我们的不足之一。