**实验二 类的设计（封装）**

**实验内容（目的）：**

1. 练习以对象方式来设计程序。
2. 练习使用类属性语法进行封装性设计。
3. 练习使用构造函数进行初始化。

**实验前思考：**

1. 对于对象的数据成员和函数成员的可访问性设置，原则是什么 ？

原则是对象的数据成员一般为私有类型，不可在对象外部直接访问，而对象的函数

成员一般为公有，可以在外部直接访问。

1. 如果对象在运作时必须与外界交换一些信息（即对对象的数据读取或写入），你会怎样设计？

对于要修改对象的私有数据，可以给对象设置一些公有的函数成员来实现。通过向公有的方法传入参数，然后在公有方法中修改私有数据。

1. 预习教程24页1.9.2节 属性。请回答： 属性的本质是什么？有何作用？

**任务：**

1. 编写一个DealSentence 程序，主要功能： 对用户输入的英文句子，根据命令选项执行相应操作，命令格式是：

DealSentence [-w ] [-r ] [-u ] <a sentence>

其中[ ] 表示可选参数，句子以双引号引住(如：DealSentence “Hello world!”)

-w: 分解为单词分行输出

-r: 反向输出（若同时有-w选项，分词分行反向输出）

-u：处理结果以大写字母输出

（提示： 反向功能 也可利用数组Array的成员函数Reverse()来实现）

输出要求示例： 原句子为 ：Hello world！

1. 无选项则直接输出：Hello world!
2. 含-w不含-r选项， 输出： Hello

World

1. 含-r 不含-w选项，输出：!dlrow olleH
2. 同时有-w -r ,输出： dlrow

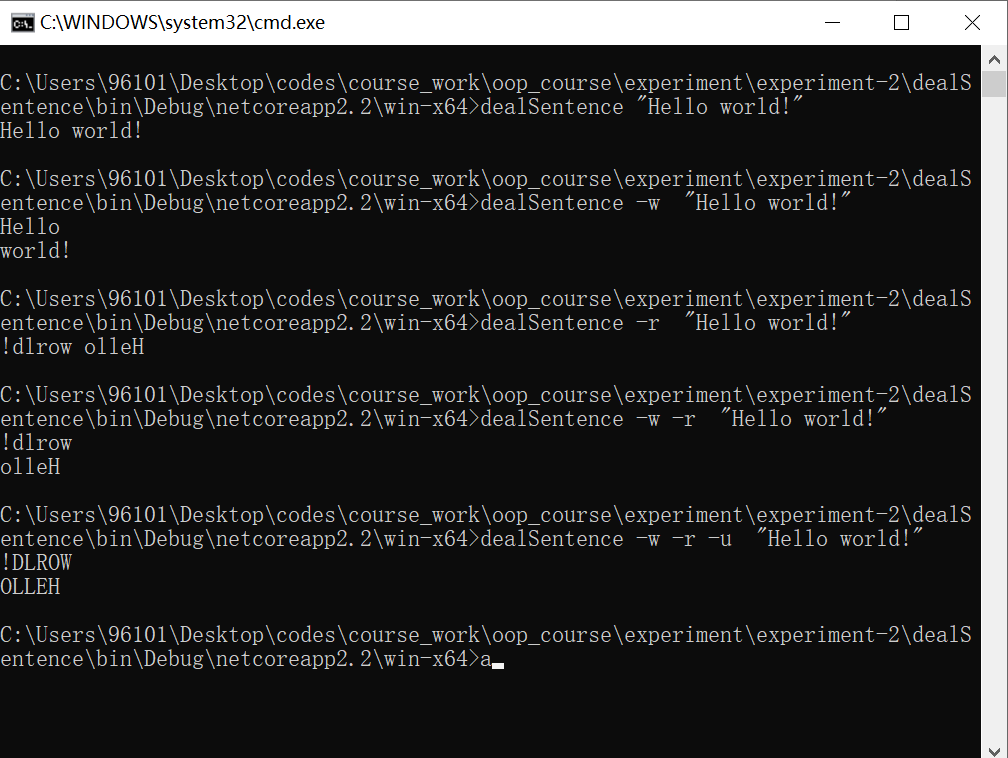
olleH

1. 只要含-u ，则大写所有字符，再按上面4条规则输出。

（1）最终你用了过程式还是对象方式来完成此任务？你认为哪种方式更合适，为什么？

运用对象方式完成。个人感觉在这种情况下用对象方式更加适合，因为本次任务输入变化具有多样性，采用面向过程的话需要很多条件判断进行分类，代码冗长不易理解。而面向对象编程面对这种输入复杂多变的任务更能

1. 请**从控制台**执行你的DealSentence程序，把执行过程截图记录下来：



1. 设计一个自定义日期类（MyDate）：包含月(int类型)、日(int类型)和年(int类型) 三个字段；增加合适的构造函数既能创建默认日期(01/01/1900)也能创建用户指定日期；另增设接口SetDate()用于日期设置操作；对每个字段提供属性；提供接口Display()能按”月/日/年”格式显示日期（提示： 要防范创建或设置了无效日期，考虑大月、小月、闰年的判断）。

另写一个测试类DateTest，

（1）在Main()中创建MyDate实例 oneDate，并执行其接口的功能；

（2）在Main() 最后加入Console.WriteLine(oneDate); 记录此句输出的内容：

myDate.Program+MyDate

你了解这样输出的原因是什么吗？

此时往Console.WriteLine函数里面传的参数类型为object

两个任务的代码请附录于此：

源代码：

（1）dealSentence

using System;

namespace dealSentence

{

    class Program

    {

        class Sentence {

            private string ret;

            public void setArgs(string[] args) {

                ret = args[args.Length-1];

                for (int i = 0; i < args.Length-1; i++) deal(args[i]);

            }

            public void deal(string para) {

                    if (para == "-w") {

                        split();

                    } else if (para == "-r") {

                        reverse();

                    } else if (para == "-u") {

                        upper();

                    }

            }

            public void split() {

                ret = ret.Replace(" ", "\n");

            }

            public void reverse() {

                char[] str = ret.ToCharArray();

                Array.Reverse(str);

                ret = new String(str);

            }

            public void upper() {

                ret = ret.ToUpper();

            }

            public void output() {

                Console.WriteLine(ret);

            }

        }

        static void Main(string[] args)

        {

            if (args.Length == 0 || args.Length > 4) {

                Console.WriteLine("输入参数数量错误");

            } else if (args.Length == 1) {

                Console.WriteLine(args[0]);

            } else {

                Sentence newSen = new Sentence();

                newSen.setArgs(args);

                newSen.output();

            }

        }

    }

}

（2）myDate