设计概述

1. 使用链表存储干支数字，这样可以支持非常大的长度，因为这个数据结构和列表不一样，不需要开辟连续的一段内存空间。
2. 设计使用了两个类，分别是节点类LNode 和干支数字链表类Sexagenary，使用的是单向链表，每个节点存放的是干支数字数据。
3. Sexagenary类实现的函数有初始化函数\_\_init\_\_，节点初始函数initList，判断链表是否为空函数isEmpty，获取链表长度函数get\_length，插入节点函数insert，移除节点函数remove,干支数字转十进制函数toDecimal，十进制转干支数字函数Decimalint\_to\_sexagenary，遍历链表函数get\_sexagenary\_string，已经加法，减法，乘法运算的函数。

实现策略

1. 初始化函数，对于传入的干支字符串首先判断是否在60个干支数字里，利用初始化的self.sexagenary2int字段的keys（）方法，循环遍历判断是否在这个list里面即可
2. 获取链表的长度，用p变量指向链表的头结点，len变量用来计数，每次遍历一个节点时，p变量指向该节点的下一个节点，并且len变量加1，最后得出链表的长度
3. insert插入操作，新建一个节点，该值为要插入的干支数字的值，获取index插入的节点，将该节点指向这个新建的节点，并且新建节点的下一个节点指向原先index节点的下一个节点。
4. 将干支数字转换为10进制数字，本质即是 60进制转10进制，即是先将干支数字转换为int类型，再每一位乘以对应的60的对应次平方数，即可获得10进制的数值
5. 讲10进制转干支数字，本质是10进制转60进制，采用的是除以60取余数的方法，再用字典的映射关系转换为干支数字。
6. 加法，减法，乘法操作的本质都一样，先将干支数字转换为10进制，10进制进行加减乘除操作，再转60进制，即转换为干支数字

测试报告

test2 = Sexagenary("戊戌甲子乙丑丙寅")  
if test2.get\_sexagenary\_string() != "戊戌甲子乙丑丙寅":  
 print("Error")  
test2.insert(1, "甲寅")

得到结果是：戊戌甲寅甲子乙丑丙寅