# 课程设计报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学院：电子学院 | 专业：微电子 | 学号：19170629 | 姓名：万力 |
| OOP模式(python 实现)   1. 把停车卡所有信息存储在CardItems类中 2. 由于3种卡所要保存的信息，附加的方法也有差别。故分为3个子类   TmpCaed(临时卡)、AmtCard(次卡)和YearCard(年卡)   1. 每条记录通过带有首位结点的单链表SSList存储起来。单链表的结点类   为Lnode.   1. 最后把存有所有信息的单链表作为CardSystem类的属性封装起来   把要实现的功能，放在该类的方法中。CardSystem有一个构造函数，可以从给定的txt文件中自动实例化.   1. 最后放一个用户接口。 | | | |
| 关键代码 ( 要求有必要的注释 )    对输入的时间进行检测。当输入不合法或者在未来时返回False.  程序里面默认”x”\*12代表此项当前没有值,”now”的时候我用的是另一套逻辑。  通过python的build-in库函数datetime.datetime.now()获取当前时间。    SLList的实现主要就是添加元素:  当在表头添加的时候：一定要判断self.\_head是否为空，不然self.\_tail可能指向不了末尾的元素。在表尾添加元素类似。  构造函数：readfromtxt, 因为python封装不像c++没有私有函数，只能靠修饰符@staticmethod 来完成构造函数，实现封装技术。  这里也是通过自带函数一行一行地读取。每次读的时候判断一下卡的类型，然后把他们放入相应的类里即可。    最后就是归并排序：   1. 通过快慢指针，找到中间结点。然后一分为二进行合并。 2. 合并的时候通过构建一个新的头结点，然后慢慢串联下去，直到有一个为空，就把剩下的接上去即可。 3. 递归的出口就是当不能再分时候，即只剩一个结点的时候。直接返回。 | | | |
| 测试结果 ( 除浏览功能外，每个功能至少2组测试用例 )  计算机生成了可选文字: 1：办卡2：浏览3．进场4．出场5．修改6．排序7．退出 请输入号码0 输入6位车牌号，12位出场时间，中间用空格隔开（退出请按q回车） 示例．E7845F2002071720S8（当前时间可以用now替代） 请依次输入车牌，出场时间09now 此车办理的卡为：年卡 停车代价．到期时间．210213；剩余天数．360 | | | |
| 系统特色：（如果没有可不写）   1. 充分利用python的异常捕捉技术try-except-else作了大量的异常处理。 2. 命令行界面比较美观，每一步都给用户足够的提示。每一步不想执行或者不小心进入下一步时都有返回项可以选择。 3. 通过OOP框架实现，可扩展性好 4. 排序的时候使用了归并排序，时间复杂度减少到O(logn)量级 5. 当剩余次数为0时，或是年卡用户到期：只是给予冻结警告，后续逻辑没有写。当剩余次数不足5次时，或者到期时间不足一个星期，给予友好提醒. 6. 把到期时间更进一步转化为剩余多少天，更加人性化 7. 自己附加了一些逻辑： 8. 车牌号、教工号只能是6位的字符串，否者无效，重新输入 9. 当检测车辆还在车库中时，不准进场 10. 能检测时间的有效性。如果时间在未来某一时候或根本不合法，都不能进入系统。得重新输入 11. 为了方便，加了一个``now``的时间可供用户输入。没必要每次都输入12个字符串。 | | | |