Document of PJ 1

第一部分：内容概述

在压缩文件内，共三个文件夹：Corecode + File +Gui

Corecode包括：RBtree.py ; Btree.py ; main.py 分别实现红黑树B树和测试程序

File包括：Unix.txt ; RedBlackTree\_exptomize ; RedBlackTree\_dump ; Btree\_preorder ; Btree\_dump 分别是原始字典文件，红黑树的树结构图，红黑树dump的结果， B树的前序遍历结构图，B树的结构图。

GuI包括：Gui.py,是实现可视化的核心代码

最后我将这些文件进行了整合，只要运行RBtree.py ; Btree.py 就可以实现PJ1要求里的可视化操作。

第二部分：代码概述

* First：RBtree.py：
* 先在Noddy里创建了一个节点类，该节点有key, value, color, parent等属性。
* 再在RB\_tree里创建了一个树的类，具有size, root, traveler(用于显示树结构）等属性。并且在RB\_tree类里定义如下函数：

| * **left\_rotate** | * **RB\_insert\_independen** | * **RB\_delete** | * **preorder\_travel** |
| --- | --- | --- | --- |
| * **right\_rotate** | * **RB\_transplant** | * **RB\_search** | * **\_\_str\_\_** |
| * **RB\_insert\_fixup** | * **Tree\_Minimum** | * **inoder\_tree\_walk** |  |
| * **RB\_insert** | * **RB\_delete\_fixup** | * **inorder\_dump** |  |

* 然后在主程序中定义8个函数：

| Intialize | Create | Put | Delkey |
| --- | --- | --- | --- |
| **Loadfile** | **Insert** | **Getvalue** | **Dump** |

* 实现PJ中要求的insert, put, get, del, load, dump功能
* 最后一部分也在主程序中完成是Gui的整合。
* Second：Btree.py
* Btree和RBtree有几处异同：
* 1.创建类的时候除了创建一个节点Nobby类,还创建了一个pair类，pair类里包括key和satellite(英文，中文翻译）
* 2.在节点Nobby类中定义了如下函数：

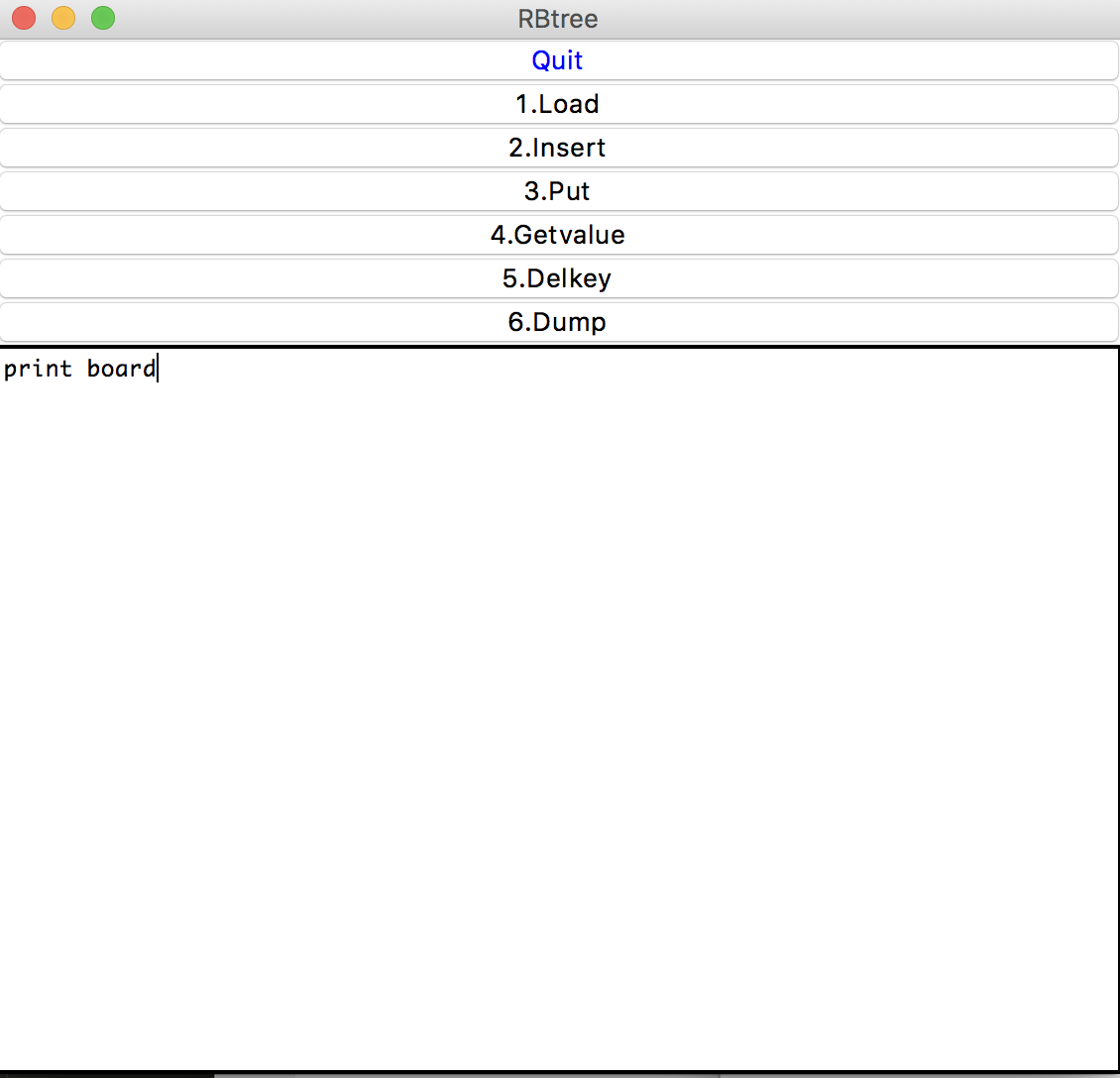
| add\_satellite | del\_satellite | add\_child | del\_child |
| --- | --- | --- | --- |
| **Delete** | **Isleaf** |  |  |

* 定义节点的函数是为后方的delete，insert做准备
* 3.在RBtree类里定义如下函数：

| Btree\_Creat | BTree\_Split\_Child | BTree\_Insert | BTree\_Insert\_Independent |
| --- | --- | --- | --- |
| **BTree\_Put\_Independent** | **BTree\_Search** | **BTree\_Search\_Predecessor** | **BTree\_Search\_Successor** |
| **BTree\_Delete** | **del\_in** | **del\_not\_in** | **BTree\_preorder\_traveller** |
| **BTree\_inorder\_traveller** | **BTree\_printall** |  |  |

* 4.在主程序中同样也定义8个函数：实现PJ要求的功能

| Intialize | Create | Put | Delkey |
| --- | --- | --- | --- |
| **Loadfile** | **Insert** | **Getvalue** | **Dump** |

* 5.最后一部分也在主程序中完成是Gui的整合。

第三部分：GUI功能介绍

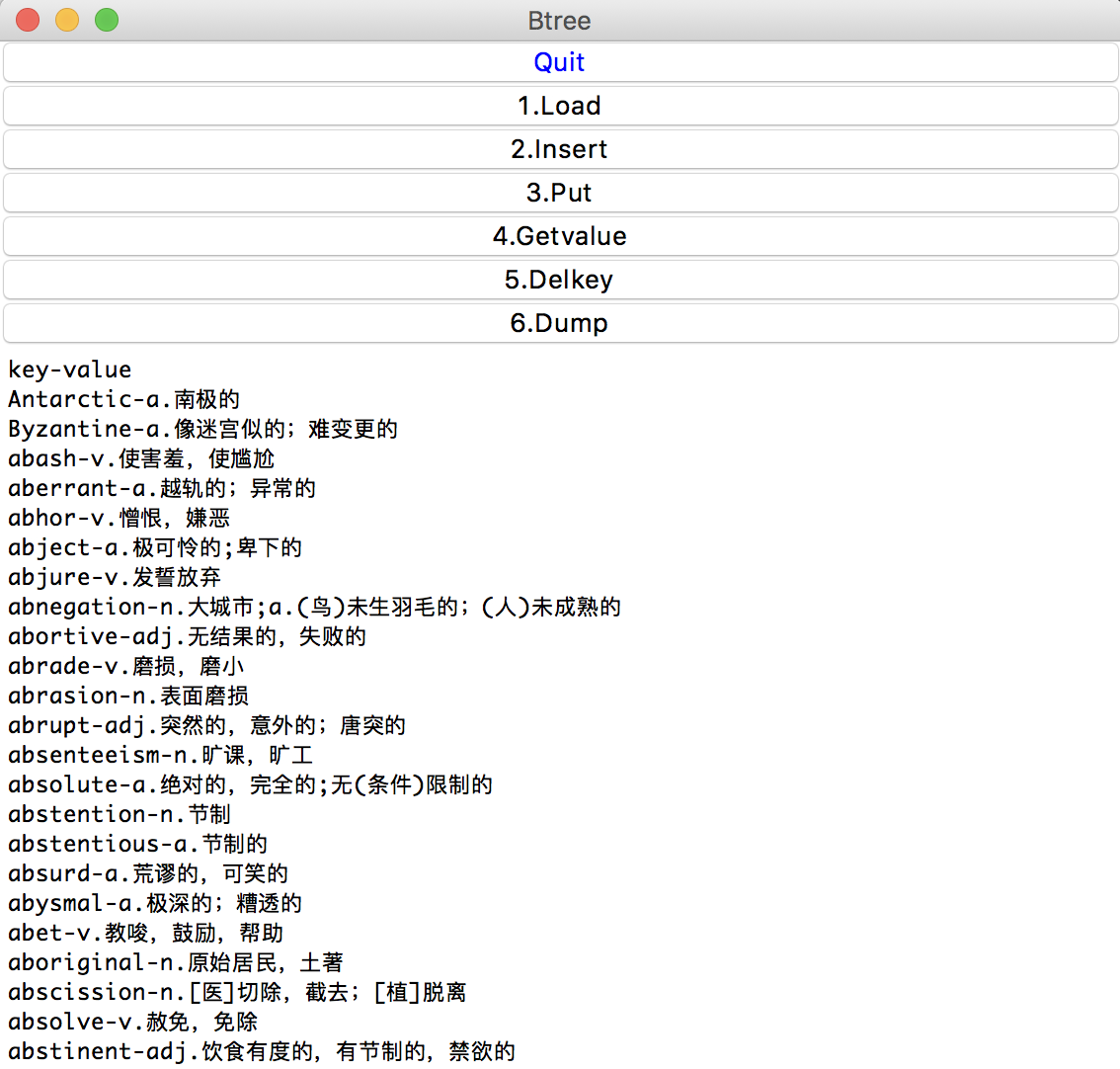
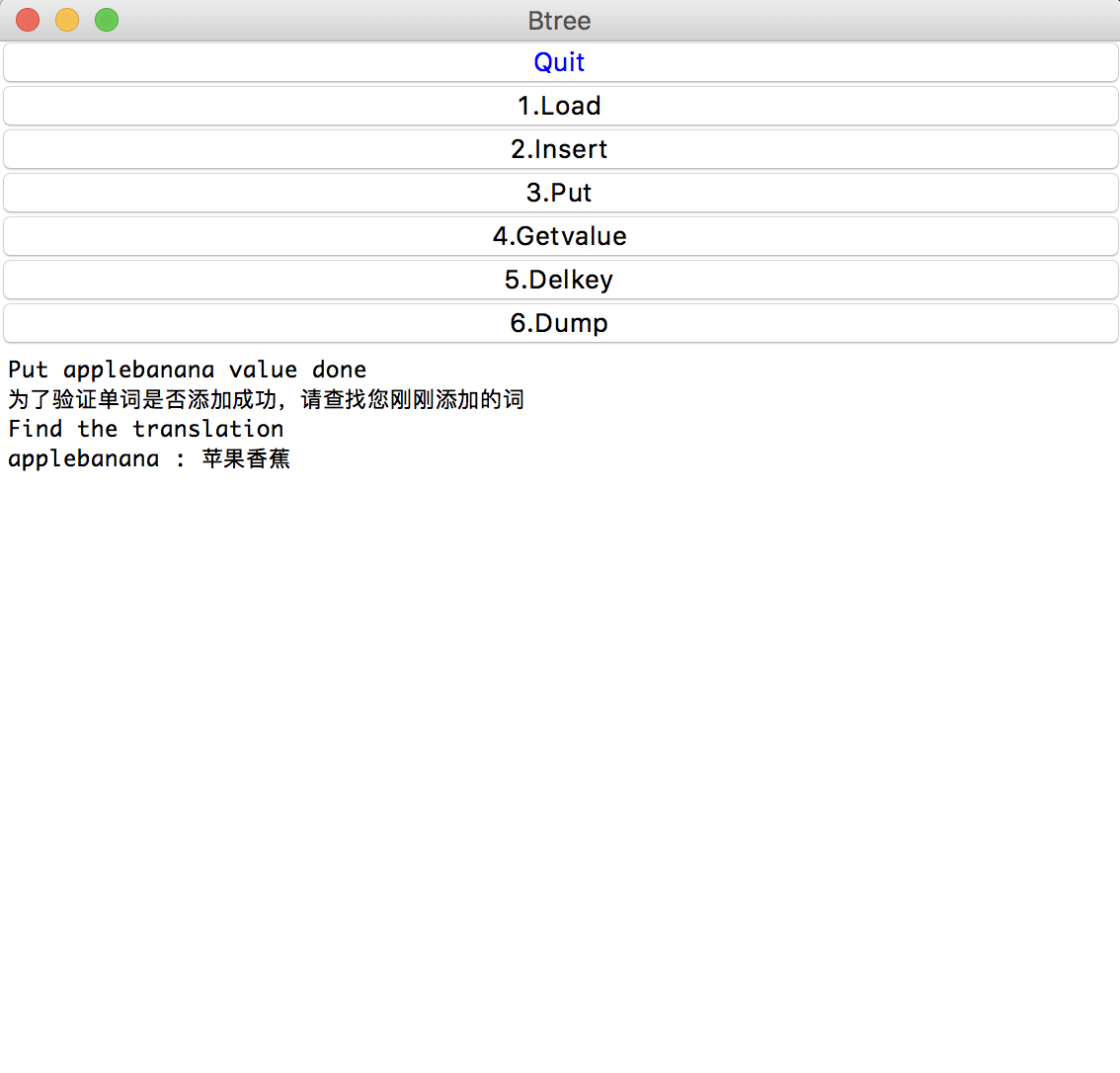
GUI是用python的包tkinter写的。

GUI的界面如右图：

包含QUIT按钮，Load，Insert，Put, GetValue, DelKey, Dump等功能。

和一个text版用来显示结果。

注意的是：每一个操作都是独立的。

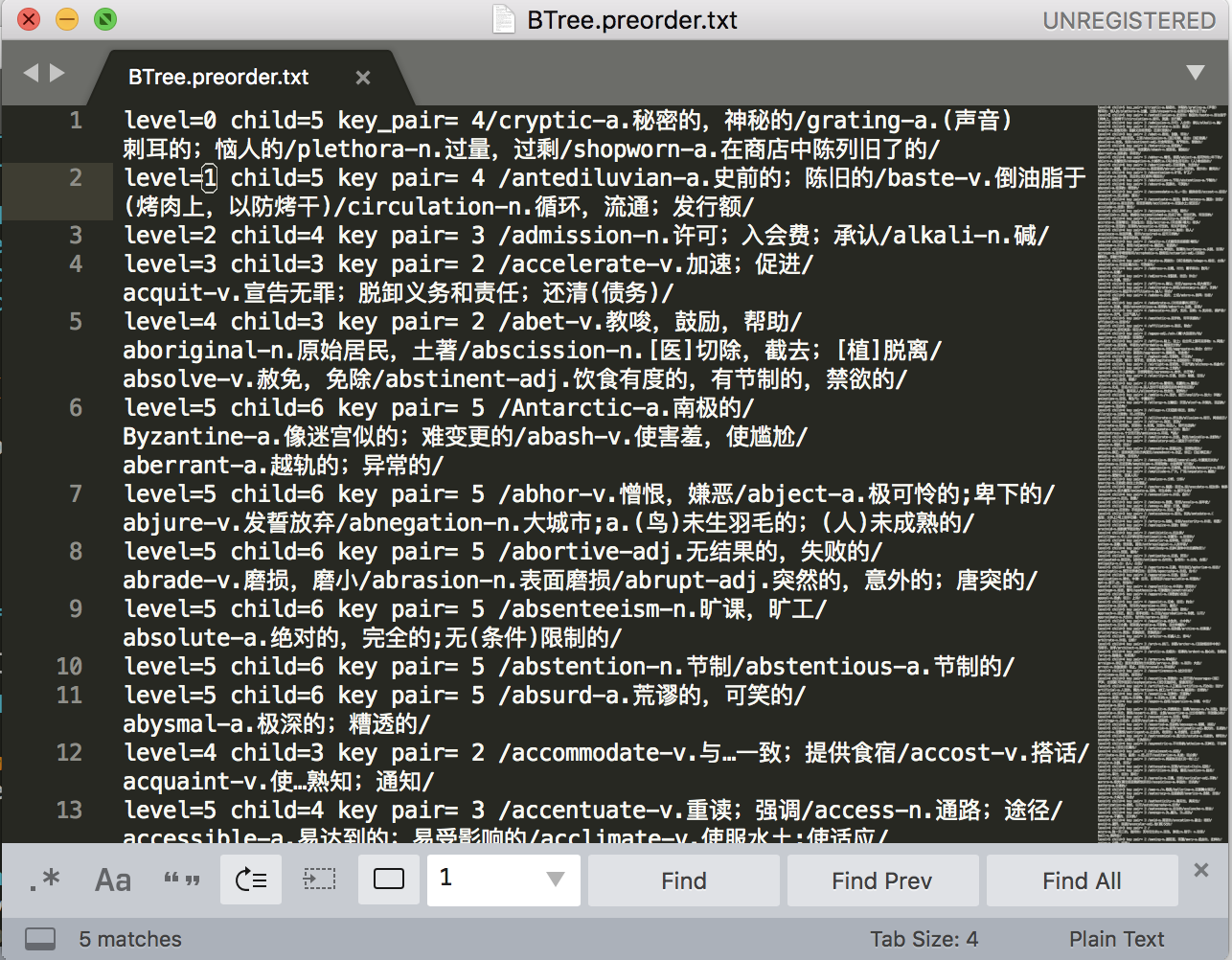
因为在每个操作前，我都create了一棵新的树。然后如果想检测Put后的结果是否正确，我在Put后连接了一个GetValue,可以直接检测是否能查到刚刚Put进去的单词。

如右图是dump的结果：

EXTRA：

我dump的程序额外加了一个功能，可以把输出的结果放入到File中的’RedBlackTree\_e

xptomize.txt’ , ‘Btree\_preorder.txt’ ,并添加level child属性,显示红黑树或B树的结构。

以B树举例：

此情况下：t：degree=3 [ t-1 , 2t-1 ] = [ 2 , 5 ]

不仅bump了所有的pair的key(单词)和satellite(中文翻译)。还包含了每个node的level，child，key\_pair。child的个数就是它指向多少个孩子节点，key\_pair的个数就是关键点的个数。

第一行中：root节点：层数为0，关键词的个数为4，孩子的个数就是5个。level=0, child=5

那么level=1的节点个数就是5个。

从下方的搜索结果中发现level=1的恰好5个。加深了对B树结构的理解。