作业

内容

homework 0914

作业一

Rotate an array of n elements to the right by k steps.

For example, with n = 7 and k = 3, the array [1,2,3,4,5,6,7] is rotated to [5,6,7,1,2,3,4]

进一步如要要求空间复杂度O(1)，除了原来分配的数组，怎么解？

作业二

给定一个二维数组，输出数据按螺旋形顺序

比如

[1, 2, 3]

[4, 5, 6]

[7, 8, 9]

输出[1, 2, 3, 6, 9, 8, 7, 4, 5]

Homework 0921

1. 在IntArray基础上实现insert/delete算法

void insert(int index, int value);

void delete(int index);

注意原来的\_size变成逻辑数组大小，真正的数组内存大小应该是\_capacity

2. 把单链表翻转过来 （参考双链表实现）

void reverse();

Homework 0925

根据伪代码在SortedIntArray里实现成员函数MergeSort

public:

void MergeSort(int start, int end);

private:

void Merge(int start, int middle, int end);

数组值需要随机生成来初始化。

同时修改QuickSort，使得取的轴pivot不是最后一个，而是随机的一个，从而避免最差情况出现。

测试修改后的SelectionSort，MergeSort，QuickSort，至少要十组不同大小的数据，数组最大可以到上百万元素。

最后输出测试数据，画图显示时间，横坐标为数组大小

附录：

计算时间需要include<ctime>

然后在之前记录一下时间，之后记录一下时间，两者之差就是运行时间

{

clock\_t tstart = clock();

.....

clock\_t tend = clock();

cout << "Elaspe " << tend - tstart << " milliseconds" << endl;

cout << "Elaspe " << (float)(tend – tstart)/ CLOCKS\_PER\_SEC << " seconds" << endl;

}

Homework0928

实现自定义字符串：MyString类, 参考书上例子和之前的IntArray。

尤其是加亮的函数

Find()，现在用暴力匹配，之后我们再添加优化算法。

Resize()，把字符串长度改变，原有的内容复制过来，如果新的长度变短了，注意要剪切。

Reserve()，改变数组内存分配大小,capacity。

Append()，和“+”区别，+是指两个字符串合并，然后返回一个新的字符串；Append是指把别的字符串附加在自己后面，返回的也是对自己的引用。注意这里要考虑性能问题，也就是如果对几十万个小字符串进行append操作

inline int Length() const {return \_size - 1;}

mystring

Homework 1016

1. 把几个匹配算法加入MyString，然后修改课件中的算法，使得返回值是text中

匹配的次数， 而不是第一个匹配位置

2. 任给一个字符串，比如“abacabadd”，输出它的一个最长的是回文的前缀。

这里是“abacaba”。

提示：对”abacabadd” + “#” + “ddabacaba”，求next[M]

Homework 1026

按BinaryTree的类声明，实现public函数。

遍历的几个函数，除了课件上讲的，还需要一个LevelOrder\_Backwards（），就是按层输出，不过不是“Z”字形，是“己”字形输出

两个构造函数，一个是随机生成给定大小的二叉树，一个是根据前序遍历和中序遍历，这两个数组，把二叉树还原。比如 preorder = {7,10,4,3,1,2,8,11}， inorder = {4,10,3,1,7,11,8,2}

注意析构函数需要把树的所有节点内存释放

复制构造函数和赋值重载函数，需要把树的结构完全复制过来，不是指针的copy，需要对每个节点都新建一个节点，复制后的新树结构和原来一样。

Homework1109

1. 把二叉树，改造成AVL树。注意AVL树的节点是有高度信息的（到叶节点的最长路径的长度，不是到根节点）。

AVL树需要维护二叉查找树的性质，以及根据高度的保持平衡的方法，即旋转

需要添加AVL插入函数，AVL删除函数，和AVL查找函数

实现Euler Tour的遍历（提示，使用递归方式比较简单，迭代的方式会复杂）

2. 列出自己觉得课上没讲清楚的知识点（之前所有讲过的内容），每人一到三个，不过要具体点，不要列章标题，节标题。最好具体到某个问题。之后我们在习题课上有针对性的展开。

homework1116

第七章习题5， 7，11

Homework 1123

实现类MyGraph，包括

DFS递归遍历算法

DFS+始末时间算法

BFS算法

实现找SCC算法

以上遍历算法参考课件。

graph类的实现可以参考下面

Homework1130

最后一次作业（12/13交）

1. 在上一次作业中graph类的基础上，实现dijkstra算法（用heap作为优先队列，heap是单独一个类）找从s出发到某个顶点v的最短路径（一找到v后，可以提前结束循环），并且依次输出s到v的最短路径节点

2. 实现bellman-ford算法中动态规划过程，topdown的方式：递归+记忆所需中间结果。