**实验设备：**

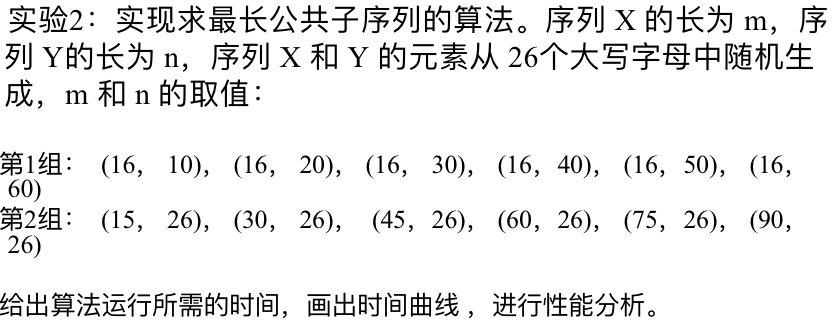
Laptop

**实验环境：**

macOS操作系统、Clion IDE

**实验内容：**

实验1：实现求最优二叉搜索树算法。关键字 n 数目为5, 9, 13,17, 21。输入在相应的txt文件中，输出要求打印二叉树，并给出期望搜索代价。统计算法运行所需时间 ，画出时间曲线，并进行性能分析。



**实验要求：**

**A close up of text on a white background

Description automatically generated**

A close up of text on a white background

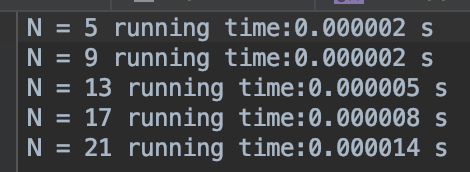
Description automatically generated

**实验方法：**

用clock函数记录程序运行排序算法时所花费的时间（以ex2举例）：

A screenshot of a cell phone

Description automatically generatedA close up of text on a black background

Description automatically generated

（程序代码） （结果） （ex1的时间）

算法的实现主要来源于课本

（坑：书上ex1算法的p是从1开始的，而我一开始写的是从0 开始，一步之差错成千里。）

**实验步骤：**

首先将课本知识复习了一遍。

其次复习了一下C语言文件读写的知识以及如何产生足够多的随机数

再先将ex的核心排序算法写好，之后文件读写部分代码基本无修改，主要是算法的部分进行改动

文件指针部分的知识比较坑爹，有的时候忘记关闭文件指针会出错

最后为每个算法测试运行时间并写入文件

EX1:感觉难点不在于算法求期望值的部分，而在于如何前序输出树的结构

输出树结构的思路如下：首先明确root数组的含义。其次是前序遍历的含义。最后是如何构造一个递归算法。root[i, j]表示从i到j的节点组成的树，其中信息含有根节点。所以从1到n的树，可以表示为根节点加上两棵子树：从1到root[i, j] - 1的树和从root[i, j] + 1到n的树。前序遍历，就是把先打印根节点，再打印左子树，再打印右子树。那么从前两条出发，就可以制造出一个递归的算法。

EX2:主要是文件读写操作的繁琐使得调Bug的时间花费较多……主要的LCS算法思路比较清晰，就是繁琐的格式比较花费时间。主要难点在于如何将书本上的结构顺序输出公共最长子序列

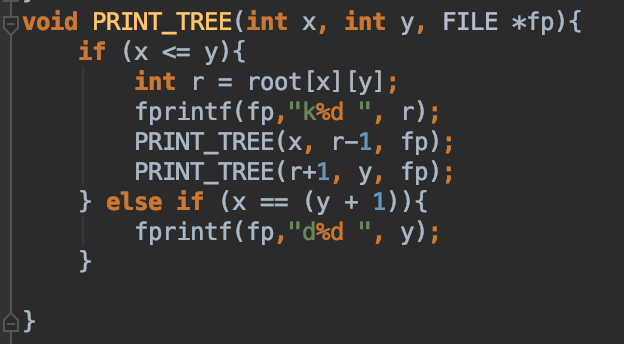
EX1:

1. 构造二叉树（书上算法）

A circuit board

Description automatically generated

1. 打印二叉树

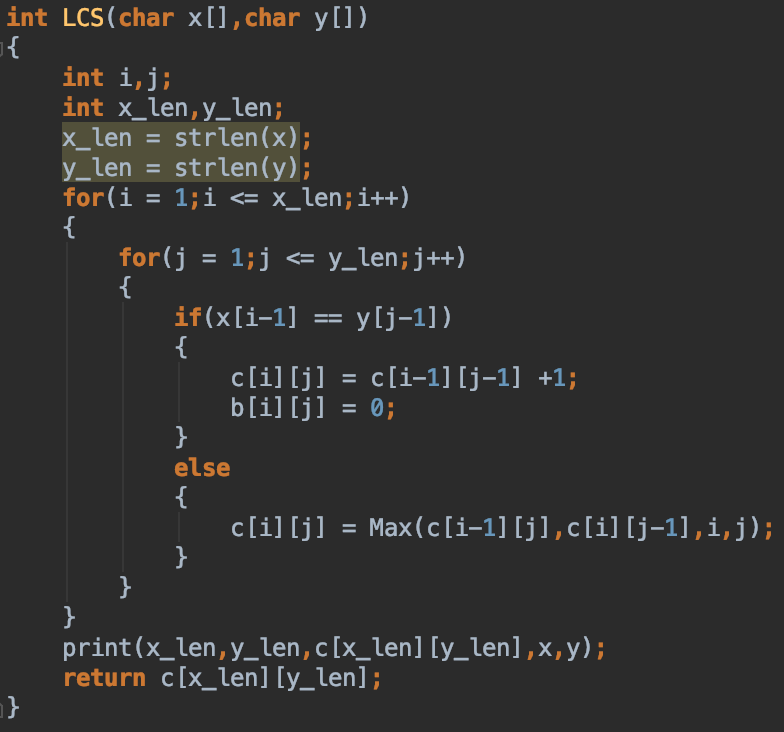


EX2:

1. 产生随机数的过程



1. LCS核心算法

s

**实验结果与分析：**

1. 经检查，实验均已完成且无错误
2. 算法在不同规模下的运行时间

EX1:

A close up of a keyboard

Description automatically generated

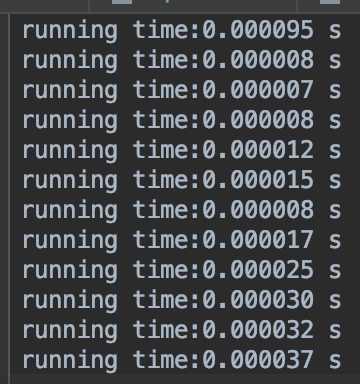
时间：

A black sign with white text

Description automatically generatedA screenshot of a cell phone

Description automatically generated

EX2:

A screenshot of a cell phone

Description automatically generated

解释曲线的原因：可能开开始调用函数的时候需要开辟一些数组空间，计算机分配内存所以需要的时间比较多，而后面调用函数这些部分工作已经省去，时间增长和课本出入（除了第一个）变化不大