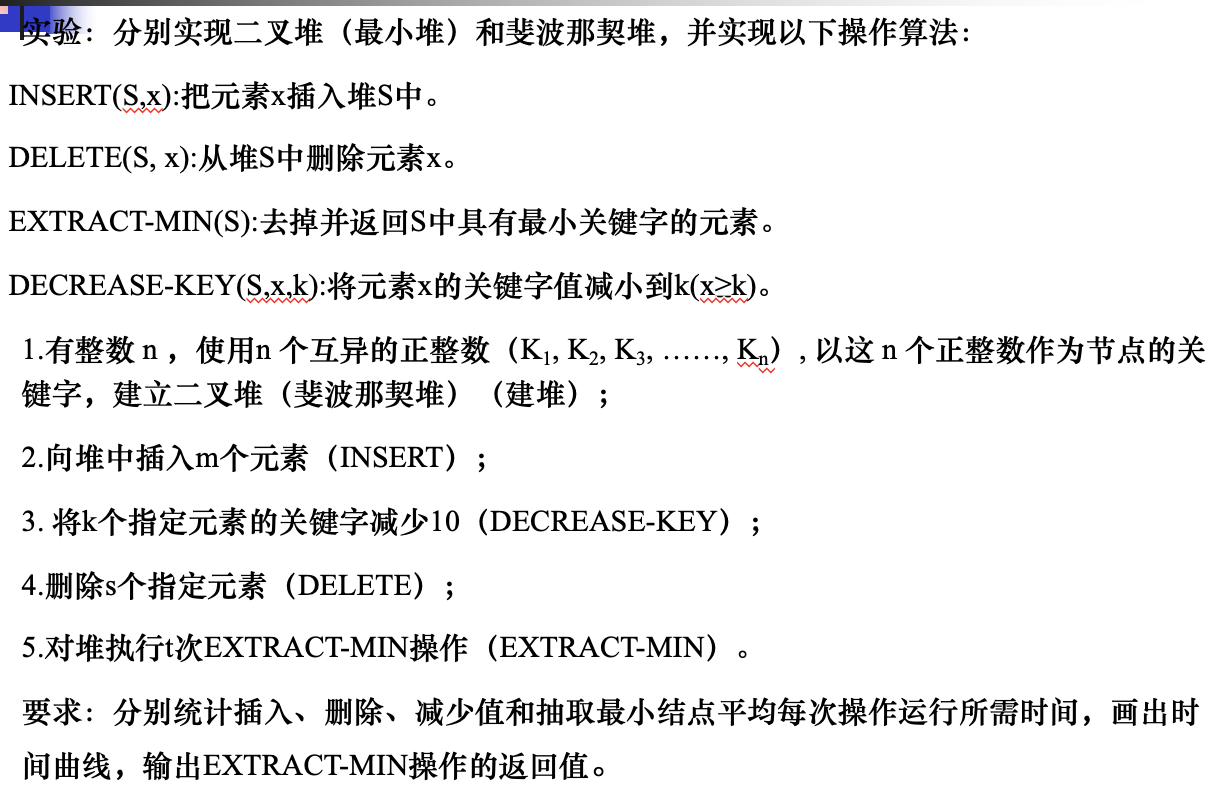
**实验设备：**

个人电脑

**实验环境：**

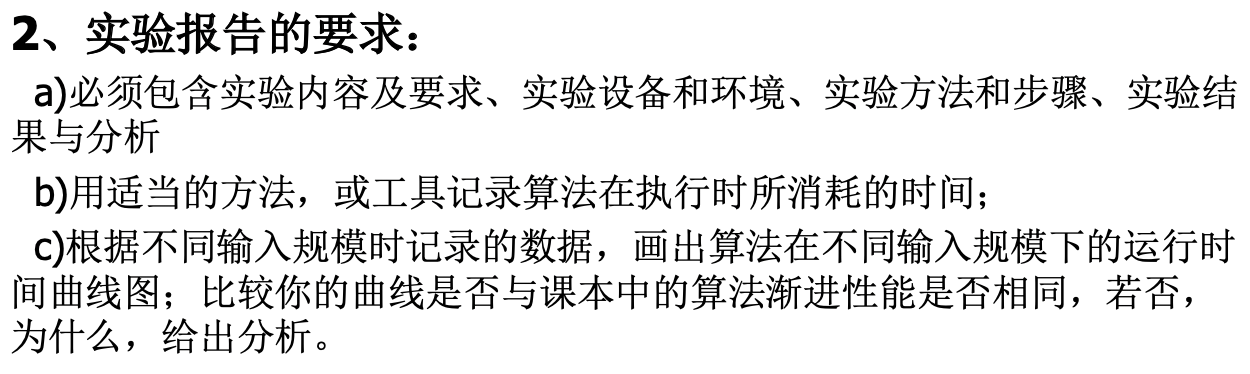
macOS操作系统、Clion IDE

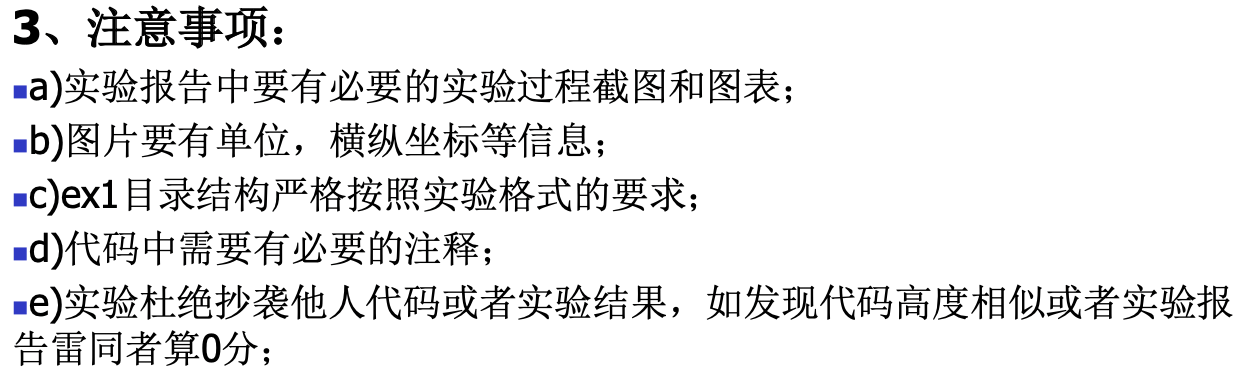
**实验内容：**

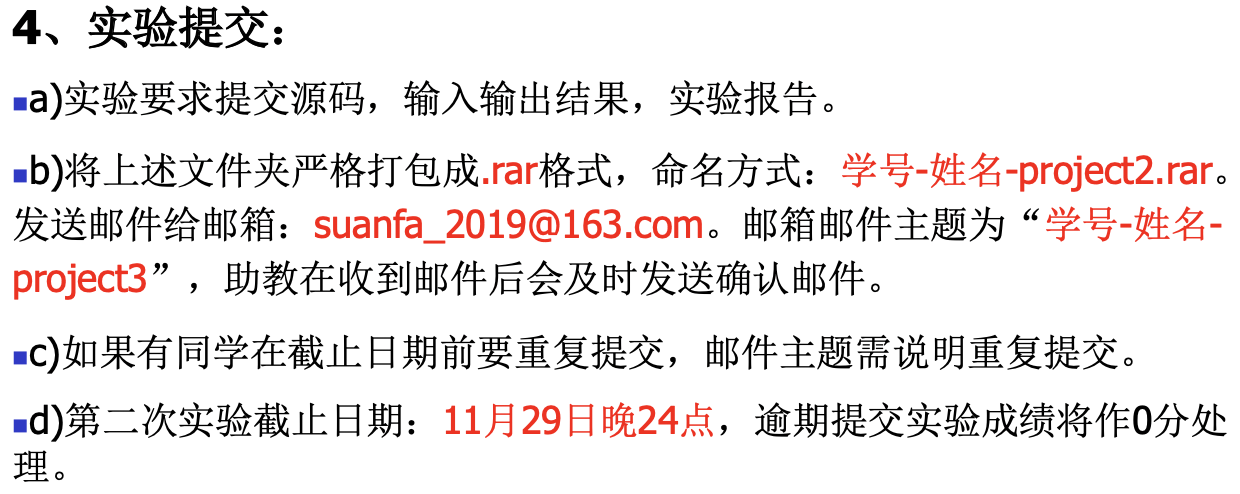
**实验要求：**

图片包含 文字, 屏幕截图

描述已自动生成







**实验思路：**

1. 二叉堆
   1. 按照顺序逐个写出五种操作
      1. 建堆 BUILD\_MIN\_HEAP(Data, n);
      2. 插入Heap\_Insert(698);
      3. 将指定元素的关键字减少10 DECREASE\_KEY(699);
      4. 删除DELETE(698);
      5. 执行EXTRACT-MIN操作，返回最小值min = EXTRACT\_MIN();
   2. 单个文件进行测试
   3. 对IO操作进行改进，一次可以处理所有文件
2. 斐波纳切堆
   1. 首先弄清楚需要定义什么结构，以及结构的类型节点有哪些
   2. 模块化处理函数，按照书上的算法来编写C程序
   3. 分别为单个功能设计调试代码，进行调试

**实验具体步骤：**

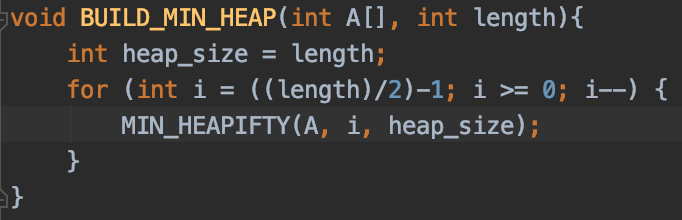
二叉堆：

总体IO框架的设计



五种函数的设计：

建堆：采用lab1的堆排序过程



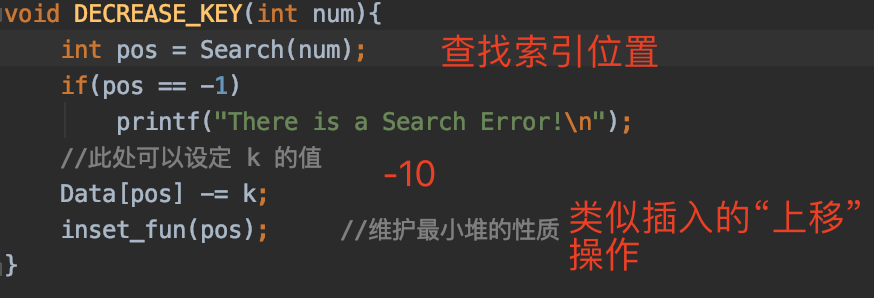
插入：

图片包含 屏幕截图, 文字

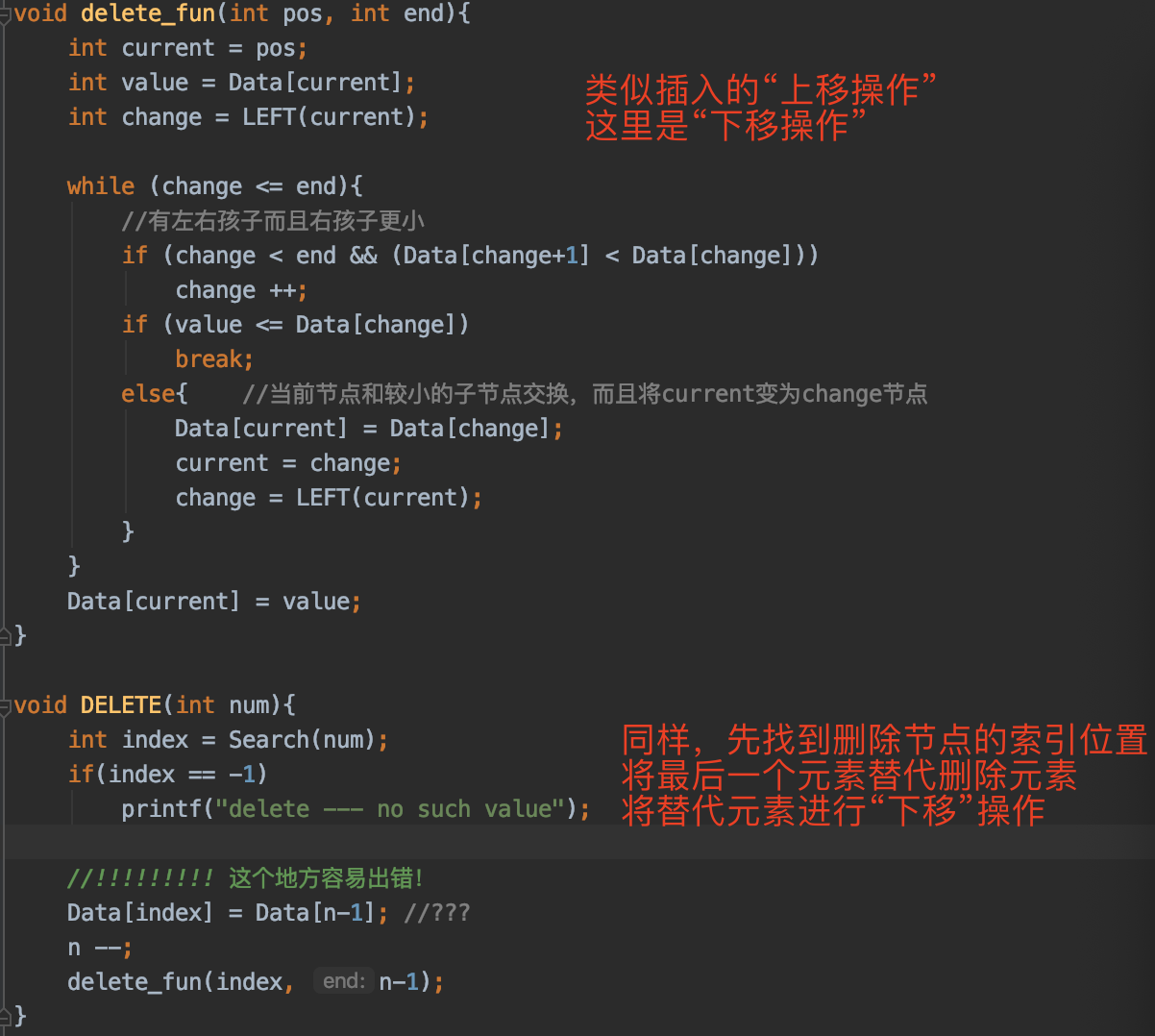
描述已自动生成

DECREASE-KEY过程：

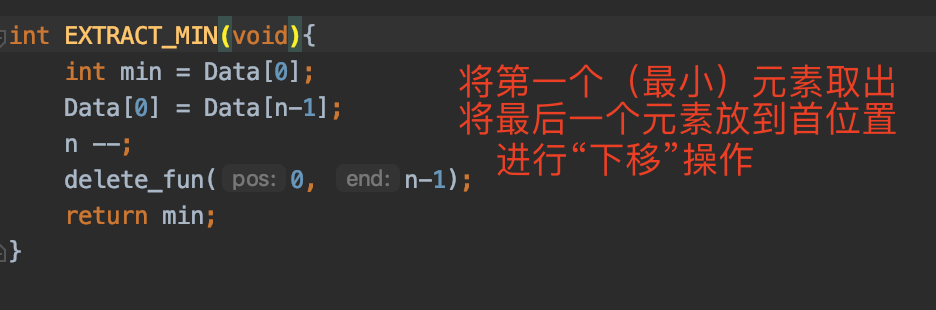
其中k的值已经设定为 – 1 0



delete过程：



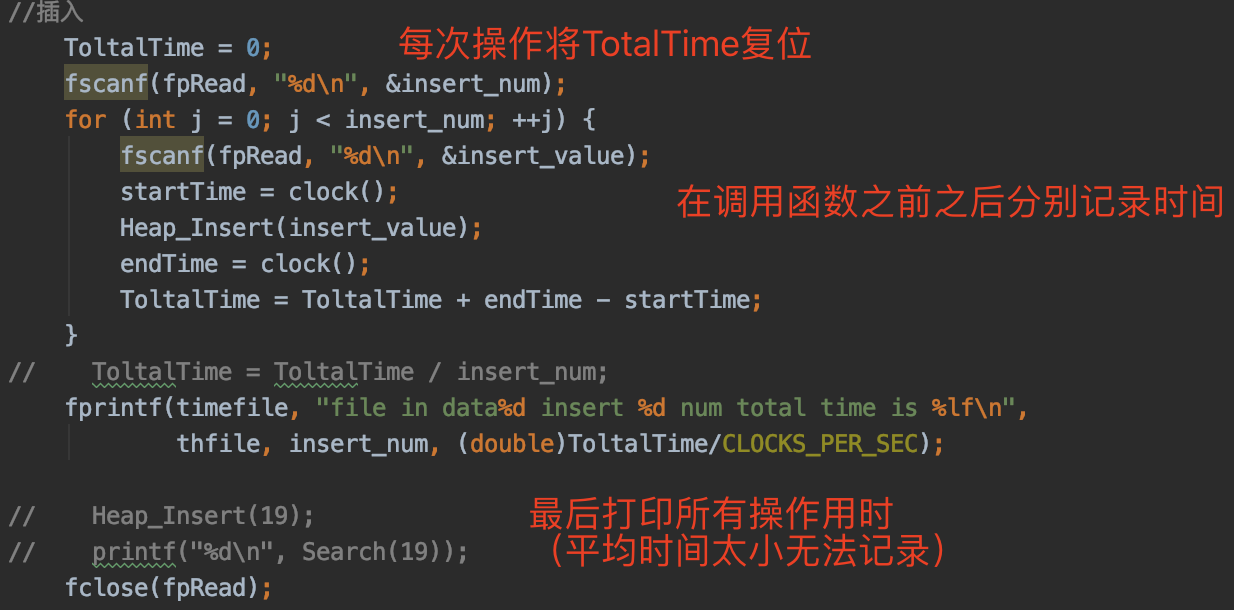
EXTRACT-MIN过程



时间的测试：

总体框架：





其中删除和DECREASE-KEY过程涉及到搜索时间，因此将在内层函数记录总共时间：

图片包含 屏幕截图

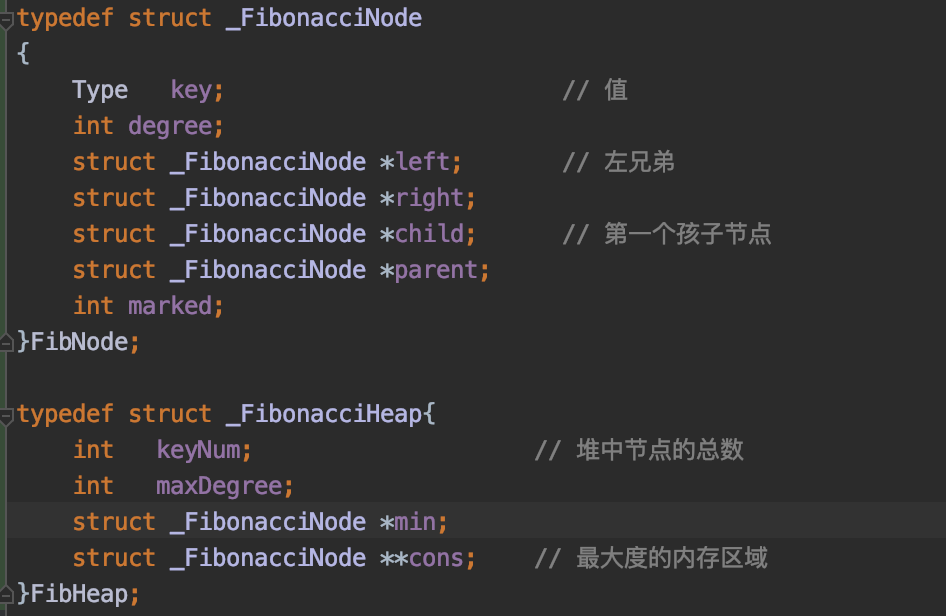
描述已自动生成

图片包含 屏幕截图

描述已自动生成

Fibonacci heap：

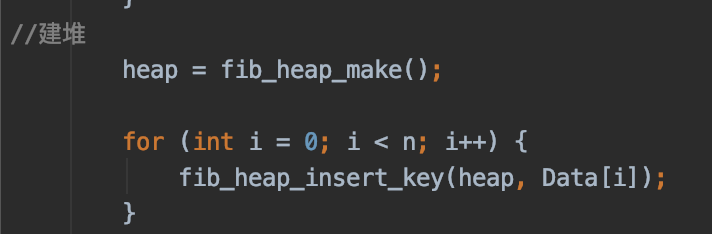
首先整体框架结构的设计：



文件IO接口和二叉堆的接口相同，这里不再赘述

建堆过程：我是按照一个一个插入堆的过程实现的

图片包含 屏幕截图

描述已自动生成

插入函数：

图片包含 屏幕截图

描述已自动生成

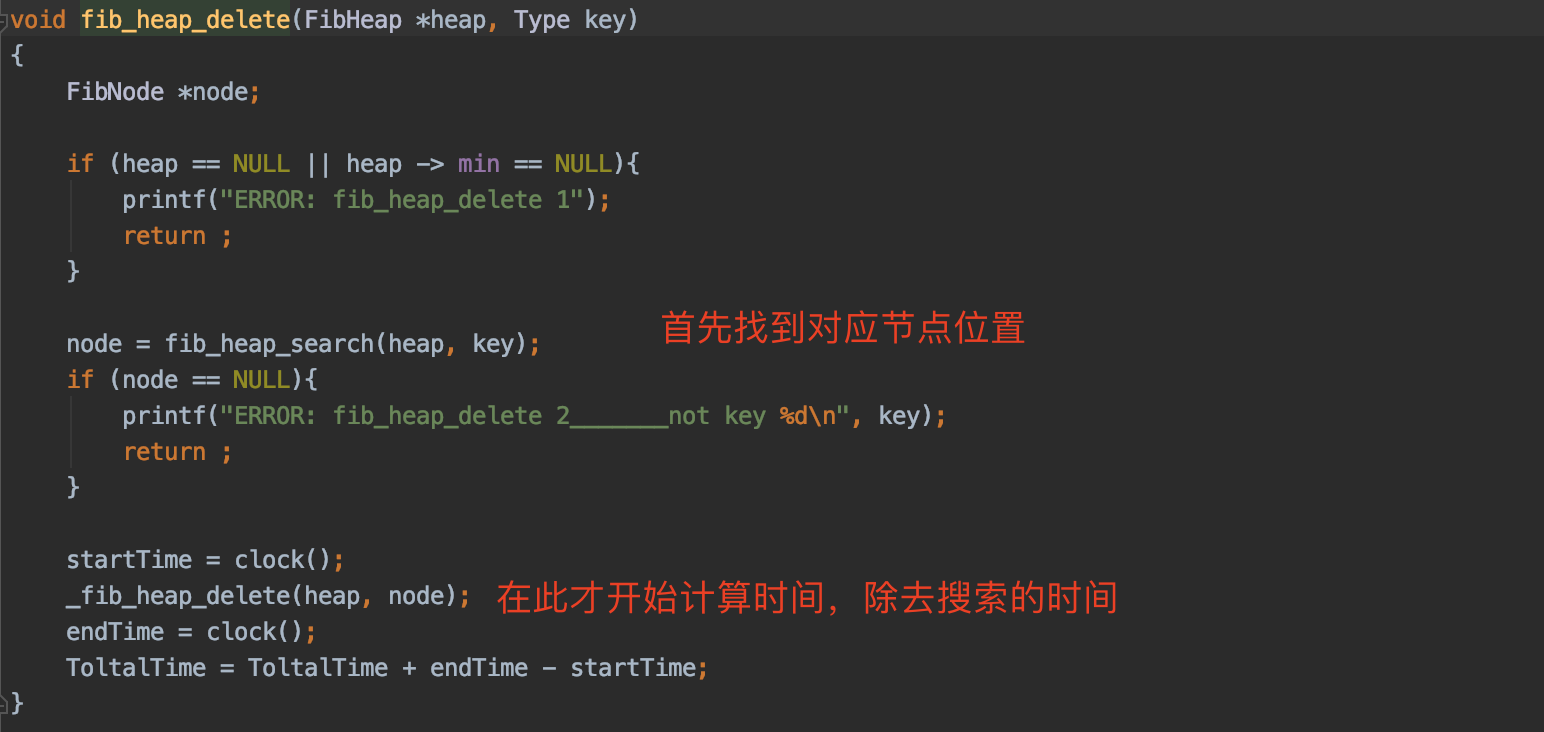
减少值：

图片包含 屏幕截图, 户外

描述已自动生成



删除操作：





抽取最小节点：



**实验结果与分析：**

二叉堆：

输出不同输入文件中EXTRACT-MIN操作的返回值：

图片包含 户外

描述已自动生成

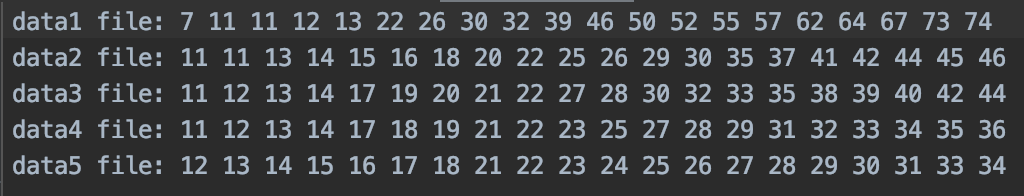
运行时间（每种操作运行的总共实际时间）：

图片包含 屏幕截图

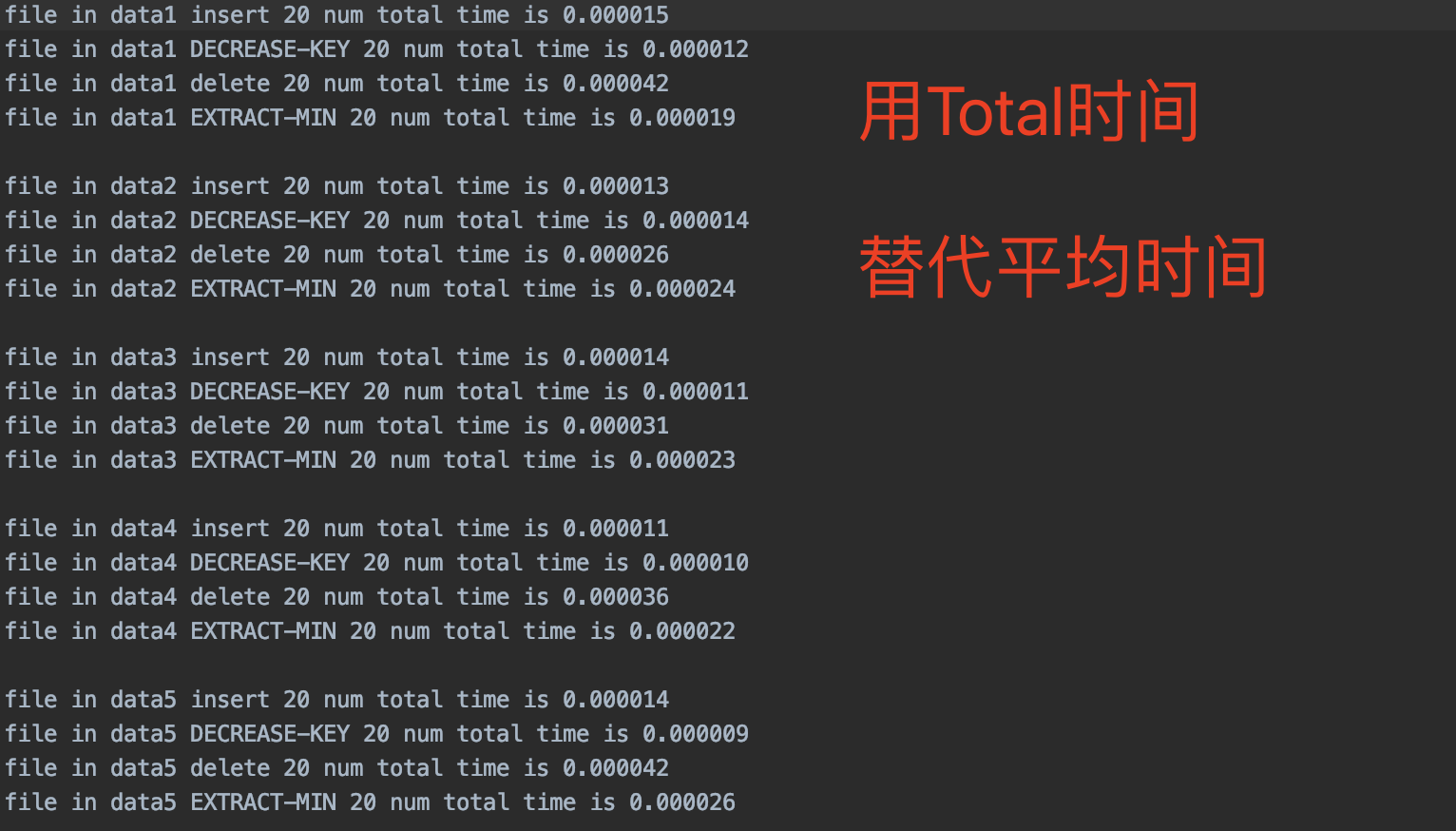
描述已自动生成

Fibonacci heap：

输出不同输入文件中EXTRACT-MIN操作的返回值：



运行时间（每种操作运行的总共实际时间）：



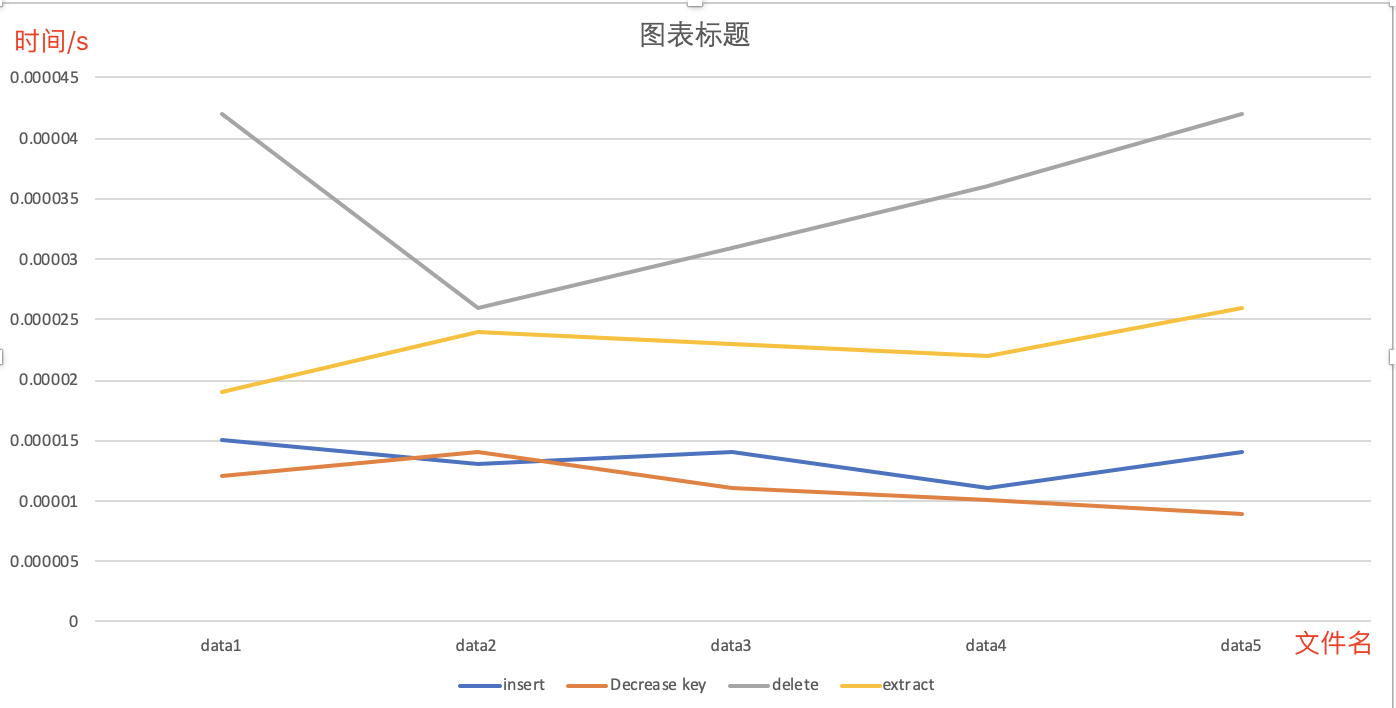
曲线图：

二叉堆：

图片包含 屏幕截图

描述已自动生成

Fibonacci 堆：



我的曲线基本与课本中的算法渐进性能相同