**实验报告**

**实验名称**  图与散列表

**班 级**

**学 号**

**姓 名**

**成 绩**

|  |
| --- |
| **实验概述：** |
| **【实验目的及要求】**  通过编程掌握图的度分布、平均路径长度、聚类系数、度度相关性和簇度相关性的概念和计算方法；掌握散列表的常见构造方法和两种冲突处理方法。 |
| **实验内容：** |
| **【实验代码1】附件：图实验.c + 画图.py**  **【实验过程1】**  先运行**图实验.c**，可以实现需要的功能，得到窗口输出和文件。  再运行**画图.py**，绘制出所需的图像。    运行结果    纵坐标为概率，横坐标为度的散点图    节点聚类系数和度的散点图  **【遇到的问题1】**  因为python画图更方便，所以一开始用python写的，但是循环过多导致程序执行太慢，所以最后还是用C语言重写了一遍，再用python画的图。  **------------------------------------------------------------------**  **【实验代码2】附件：散列表实验.py**  **【实验过程2】**  1．输入关键字序列：{24,30,23,41,51,68,46,11,14,61,35}  1)输入关键字个数n = 11，哈希表长度m = 15，取余的质数13，采用线性探测法解决冲突（输入1）。输出哈希表。计算查找成功和失败的 ASL。  2)输入关键字个数n =11，哈希表长度m = 13，取余的质数13，采用线性探测法解决冲突（输入1）。输出哈希表。计算查找成功和失败的 ASL。  观察比较 1）和 2）中查找成功 ASL 的关系以及查找失败 ASL 的关系并简要分析填充系数对查找性能的影响。  2. 输入关键字序列：{46,25,40,15,67,34,6,21}  1)输入关键字个数n = 8，哈希表长度m = 11，取余的质数11，采用线性探测法解决冲突（输入1）。输出哈希表。计算查找成功和失败的 ASL。  2)输入关键字个数n = 8，哈希表长度m = 11，取余的质数11，采用二次探测法解决冲突（输入1）。输出哈希表。计算查找成功和失败的 ASL。  观察比较1）， 2）中查找成功 ASL 的关系以及查找失败 ASL 的关系并简要分析两种探测法的优劣。  **【实验结论2】**  1.  1)    2)    分析：  第一次的ASL比第二次的都要小。  装填系数越大，填入表中的元素较多，产生冲突的可能性就越大，查找起来冗余的比较次数增多，查找性能就越低。  2.  1)    2)    分析：  第一次查找成功的ASL比第二次小  第一次查找失败的ASL比第二次大  两个算法并没有绝对的优劣，线性探测法会造成数据的聚集，需要多次解决冲突，但是查找性能相对更好；二次探测法会能够很好的避免这种聚集现象，但是查找性能相对较差。 |