

数据结构及其算法实验

2025 年秋季学期

实验说明

实验要求

1. 每个实验都需要每人独立完成。
2. 实验 1 不得使用 C++ 提供的标准模板库 (STL, 包括 `vector`、`list` 等), 其他实验**鼓励**使用 C++ STL。
3. 实验 1~5 (均为必做) 不得使用任何 AI 辅助工具, 且要在规定时间内完成, 由助教检查程序运行结果。实验 6、7 (均为选做) 允许使用 AI 辅助工具, 但必须在提交的文档中明确写出 AI 辅助工具的使用情况 (包括工具名称和使用的全过程等)。

提示

1. 实验中需要用到的几个文件, 请下载 `dsa_exp_supp.zip` 并解压缩得到。
2. 如果有条件可以用自己的电脑写程序; 如果没有, 在上机前需要做好准备, 将算法写好。
3. 课件 PPT 提供的代码绝大部分都是正确的, 课本上提供的代码可能有小的瑕疵。
4. 要学会利用编程工具提供的调试 (`debug`) 功能, 例如通过在程序中加入 `printf` 函数等方式来查找程序中的问题。
5. 上机时可以相互讨论或请同学帮助解决问题, 但注意不要影响其他同学。
6. 每次上机结束时, 将源代码保存到自己的 U 盘或网络存储中, 谨防丢失。

实验 1 一元多项式的运算

实验时间

9 课时。

实验目的

1. 熟悉编程环境，学习程序调试的方法。
2. 熟练掌握 C/C++ 语言中指针的操作。
3. 掌握链表的常用算法。

问题描述

一元多项式可以表示为线性表，相应地一元多项式的运算可以用线性表的基本运算来实现。本实验要求以链表为存储结构，设计一个一元多项式运算器，实现一元多项式的创建、销毁、复制、打印显示、四则运算等功能。

实验内容

1. 熟悉编程环境，对文件 `listdebug.cpp` 进行调试，观察指针越界、指针非法等情况下，程序运行和调试时的表现。
2. 以链表作为一元多项式的存储结构（可以自选单链表或者双向链表，自己决定是否加入头结点、是否使用循环链表、是否设置尾指针等等），实现一元多项式运算器。

基本功能：(1) 创建多项式；(2) 打印显示多项式；(3) 销毁多项式；(4) 求两个多项式的和；(5) 求两个多项式的差；(6) 求两个多项式的积。

可选做的高级功能：(1) 求两个多项式的商和余式；(2) 求两个多项式的最大公约式和最小公倍式。

实现提示

1. 由于程序有多项功能，可采用菜单选项的方式来分别调用各项功能。
2. 销毁多项式时，要注意释放所有结点的存储空间。
3. 求两个多项式的积可分解为一系列多项式求和运算。
4. 求商和余式可分解为一系列单项式除法、乘法和减法运算。
5. 求最大公约式可采用欧几里得辗转相除法，最小公倍式可由两个多项式及其最大公约式求出。

实验 2 表达式求值 / 银行排队系统仿真（二选一）

实验时间

6 课时。

实验目的

1. 掌握栈和队列的基本算法。
2. 学习利用数据结构解决实际问题的方法。

问题描述

以下两个问题，任选其一。

1. 表达式求值：计算一个算术表达式的值。
2. 银行排队系统仿真：模拟一个银行排队系统一天的运行情况。

实验内容

1. 表达式求值

基本功能：计算一个语法正确的、仅有四则运算（带括号）且操作数全部为正数的算术表达式的值。

可选做的高级功能：(1) 对表达式的语法进行分析，若语法错误则提示；(2) 允许输入的操作数为负数；(3) 增加更多的运算，如乘方、开方、三角函数、对数函数等。

2. 银行排队系统仿真

基本功能：模拟一个单队列、多窗口的银行排队系统一天的运行情况，随机生成顾客的到达时间和办理业务所需时间等数据。一天结束后，统计顾客的平均等待时间和窗口的平均占用率。

可选做的高级功能：(1) 改变顾客办理业务所需时间或银行的窗口数，观察顾客平均等待时间和窗口平均占用率如何变化；(2) 假设每位顾客有一个容忍时间，等待超过容忍时间之后顾客将离开，统计顾客的离开率；(3) 假设银行有两个队列，其中一个是 VIP 队列，另一个是普通队列，窗口服务的规则改为：若 VIP 队列不空，则优先服务 VIP，否则服务普通顾客，统计 VIP 和普通顾客的平均等待时间有多大的差别。

实验 3 Huffman 编码和解码

实验时间

6 课时。

实验目的

1. 掌握二叉树的存储结构和常用算法。
2. 熟练掌握递归程序设计方法。

问题描述

Huffman 编码是二叉树的典型应用之一。给定一个文本文件 `stdio.h`，对其进行编码和解码，计算压缩比，从而了解数据压缩的基本原理。

实验内容

1. 对文本文件统计各个字符的出现频率，构造 Huffman 树。
2. 以 Huffman 树对文本文件进行编码，统计编码后的比特数，除以 8 得到字节数。用原文件的大小（字节数）除以编码后的字节数，即求得压缩比。
3. 将编码后的比特流再进行解码，写入一个新的文本文件，与原文件比较，是否完全一致？比较文件可使用 Windows 命令行工具 `fc`。

实验 4 最短路径

实验时间

6 课时。

实验目的

1. 掌握图的存储结构。
2. 掌握 Dijkstra 算法或 Floyd 算法。

问题描述

给定全国铁路网，对于任意一对城市，找出它们之间的最短路径经过哪些城市，并输出最短路径的长度。

铁路网的信息可查看 `dist.txt`，其格式为：

城市 A 编号 城市 B 编号 距离

城市编号和城市名称信息可查看 `city.txt`，其格式为：

编号 城市名称

实验内容

1. 基本功能：图的存储结构使用邻接矩阵；可选做的高级功能：图的存储结构使用邻接表。
2. 求出下列城市之间的最短路径：沈阳至西安、呼和浩特至成都、上海至乌鲁木齐。
3. 从铁路网中删除一些城市（例如郑州），再重新计算上述城市之间的最短路径。

实验 5 排序算法的计算复杂度

实验时间

3 课时。

实验目的

1. 掌握各种排序算法。
2. 学习测量程序运行时间的方法。

问题描述

在课程学习中，我们已经知道不同的排序算法具有不同的时间复杂度，那么在具体应用中，各种排序算法的运行时间究竟相差多少？通过这个实验，对程序运行时间进行实际的测量，可以直观感受到时间复杂度与问题规模的关系。

实验内容

本实验要求编程实现至少 5 种排序算法（快速、堆、归并必做，其他选做），并在不同 N 值（如 10000、100000、1000000）的条件下多次运行程序计算平均运行时间。

实现提示

为了公平起见，我们应该使用同一个无序序列作为输入，来测量不同排序算法的运行时间。那么无序序列如何得到？一种方法是，先生成一个长度为 N 的有序序列，再将该序列随机重排(random shuffle)，从而得到一个长度为 N 的无序序列。

测量程序的运行时间，我们可以使用 C/C++ 语言提供的计时器。需要注意的是，该计时器的灵敏度比较低，在 Windows 系统中，一般只有当两组运行时间相差 0.1 秒以上时，才能认为这两组时间是有差别的。

部分程序代码示例如下：

```
typedef int ElemType;

typedef struct {
    ElemType *r;
    int len;
} SqTable;

void InitList(SqTable &L, int len) {
    // 0 号单元不用
    L.r = (ElemType*)malloc((len+1)*sizeof(ElemType));
    L.len = len;
}

void CopyList(SqTable L, SqTable &newL) {
    newL.r = (ElemType*)malloc((L.len+1)*sizeof(ElemType));
    newL.len = L.len;
    memcpy(newL.r, L.r, (L.len+1)*sizeof(ElemType));
}
```

```

// 求一个整数的 p 次方
int intpow(int n, unsigned int p) {
    int res = 1;
    for (unsigned int i=0; i<p; ++i)
        res *= n;
    return res;
}

// 生成一个随机整数，其取值范围是[0, bound]
int randb(int bound) {
    int r = 0;
    unsigned int power = 0;
    do {
        r *= RAND_MAX;
        r += rand(); ++power;
    } while (intpow(RAND_MAX, power) < bound);
    return r % (bound+1);
}

// 随机打乱一个数组
void RandomShuffleList(SqTable L) {
    ElemType* array = L.r + 1; int n = L.len;
    for (int i=n-1; i>0; --i) {
        int j = randb(i); // 0<=j<=i
        ElemType tmp = array[i]; array[i] = array[j]; array[j] = tmp;
    }
}

int main() {
    int N = 100000;
    SqTable L;
    InitList(L, N);
    for (int i=1; i<=N; ++i) L.r[i] = i;
    RandomShuffle(L);
    clock_t begin, end;
    // 排序算法 1
    SqTable L1;
    CopyList(L, L1);
    begin = clock(); // 计时器开始
    BubbleSort(L1);
    end = clock(); // 计时器结束
    printf("BubbleSort time: %g seconds\n", (float)(end-begin) /
CLOCKS_PER_SEC);
    // 排序算法 2

```

```
SqTable L2;  
CopyList(L, L2);  
begin = clock(); // 计时器开始  
QuickSort(L2);  
end = clock(); // 计时器结束  
printf("QuickSort time: %g seconds\n", (float)(end-begin) /  
CLOCKS_PER_SEC);  
// ...  
}
```


实验 6 旅行商问题

实验时间

选做。

实验目的

综合运用所学知识，尝试以不同的算法设计策略解决同一个问题，加深对算法设计的理解。

问题描述

旅行商问题是一个非常经典的算法设计问题。这里描述的是旅行商问题的一个基本的版本：有 N 座城市，两两之间均有道路连接，某个旅行商要将这些城市遍历一次，每个城市都访问且仅访问一次，最后要回到出发点；要求给出一种遍历顺序，使得旅行商经过的路径总长度最短。

显然，旅行商问题的数学模型是有 N 个顶点的无向完全网（因为边上有权值），旅行商问题的解则是其中 N 条边所组成的一个简单回路（因为遍历不能重复访问）。我们怎样求解旅行商问题？可能有这几种策略：

1. 穷举法。列出所有长度为 N 的简单回路，从其中找出路径总长度最短的。
2. 迭代法。先随机生成一个长度为 N 的简单回路，再通过迭代找更短的。
3. 递归法。假设 $N-1$ 时的旅行商问题能解，如何递推求出 N 时的旅行商问题的解？
4. 贪心法。请自己设计。
5. 回溯法。请自己设计。
6. 分枝定界法。请自己设计。

还有其他策略吗？

实验内容

选做实验，助教不检查。完成实验后，请提交文档和源代码作为助教给分依据。

在文档中，需要为旅行商问题设计至少 3 种算法，并分析你设计的算法是否能够保证得到最优的结果，算法的时间、空间复杂度如何。

在源代码中，需要实现你设计的至少 2 种算法，随机生成测试数据，比较不同算法的输出结果和运行时间。

完成本实验的同学可获得实验加分。

实验 7 统计话费

实验时间

选做。

实验目的

综合运用所学知识，为实际问题设计高效算法。

问题描述

给定一份通话记录文件 `records.txt`，包含约 10 万用户一天共计约 100 万条通话记录。文件是纯文本格式，每行为一条记录，格式如下：

手机号码（11 位）、呼叫类型（2 位，00 表示主叫，01 表示被叫）、通话时长（4 位，以秒为单位）、呼叫发生小区（4 位）、换行符（2 位，即 `'r\n'`）

例如，文件的第一行：`13955191490010225JQHK`

表示手机号码为 13955191490 的用户在 JQHK 小区被叫，时长为 225 秒。

计费规则如下：

1. 通话时长以分钟为单位计费，不足一分钟按一分钟计算。
2. 主叫每分钟 0.40 元（40 分）、被叫每分钟 0.20 元（20 分）。

请生成话费账单，输出一份文件，要求输出文件是纯文本格式，每行为一条记录，格式如下：

手机号码（11 位）、总费用（8 位，以分为单位）、换行符（2 位，即 `'r\n'`）

如输出文件的第一行：`1395519149000001240`

表示手机号码为 13955191490 的用户费用是 12.40 元。

实验内容

选做实验，助教不检查。完成实验后，请提交文档和源代码作为助教给分依据。

请设计计算复杂度尽可能低的算法。

在文档中，需要描述你设计的算法。源代码需要对算法进行完整实现。

完成本实验的同学可获得实验加分。