数据结构与算法分析

作业 10

本次作业共有 1 道题，需要将代码提交至在线评测系统

1. 编程实现 Kruskal's Algorithm，详见评测系统

在之前的作业中，我们发现有些同学对于编程和调试还不太熟悉，我们总结了一些相关的技巧，如果你在之前的作业中感觉编程调试比较困难，建议仔细阅读，全文如下：

这门课在课上给大家讲解算法和数据结构的思路，在理解了课上的内容之后，给你一组数据，你就应该能够在纸上，按照课上介绍的算法和数据结构一步一步算出结果。但是在实际中，面对大量的数据，我们通常需要通过计算机编程实现算法来解决实际问题。因此在作业中，会让大家编程实现课上介绍的算法和数据结构，一方面希望加深大家对课堂知识的理解，另一方面也希望大家练习提高编程实现算法的能力，让这门课程的知识能够真正为你之后的学习和工作提供帮助。

在理解了算法的思路之后，如果你已经掌握了一门计算机语句的语法，就可以进行编程实现了。但是对于大部分人面对稍微复杂的算法，一开始写出的代码都难免会有 bug，这时就需要调试（debug）。这里提供一些与（C 语言）编程以及调试相关的技巧和建议：

1. 在编程过程中，保证代码质量，努力做到一遍写对或者只有少量的 bug，这里总结了以下几点：

- 在开始编程之前，先有一个整体的构思，例如需要用到哪些变量，可以定义哪些函数，如果编写边想有时可能无法顾及全局，导致需要反复修改
- 在编写代码的过程中保持清晰一致的风格，例如缩进要正确，可以添加适量的空格换行等让代码更清晰易读
- 合理定义函数，让代码更为结构化，让自己思路更清晰，例如一些基本操作，比如链表的插入删除都可以定义成函数，而不是在代码中复制粘贴再修改，否则会让代码变得复杂难读，可能会引入错误而且不易发现
- 定义一个变量后，应该立即考虑是否需要初始化这个变量
- 使用数组和指针，需要注意检查是否已经分配了足够的空间，下标是否超出了之前定义的空间
- 编程是有技巧的，如果你觉得自己代码中某个部分写的不够清晰简洁，那么可以学习别人是如何编写的（比如同学的、网络学堂中的参考解答、或者从网上寻找相关代码），因为同样的功能在实际编程实现的过程中可能有不同的写法，有的写法简

单易写，有的则比较复杂。例如链表在实现的过程中，就可以通过一些技巧，来避免特殊处理第一个或者最后一个节点，这一点在之前作业的参考解答中有涉及到。通过学习别人的编程技巧，运用在自己的编程过程中，能够让你编程更轻松，代码更清晰，更不容易出错

2. 在之前的作业中，我们发现有的同学比较依赖评测系统，在编写完代码之后，自己测试了小规模的数据发现没有问题，就提交到了评测系统，结果发现只通过了一部分测试，然后就不知道该如何调试代码了。更好的做法是，在代码编写完成，没有语法错误，已经可以正常编译之后，你应该先对自己的代码进行充分的测试。想要测试代码是否正确，需要有相应的测试数据。在作业中，首先可以使用题目中提供的样例进行测试，样例的数据规模较小，便于理解和检查。如果在样例上能够得到正确的结果，只能说明你的代码基本正确，为了进一步测试代码，你应该按照题目中要求的规模，按照从小规模到大规模的顺序，准备不同的数据对代码进行测试。如果可能的话，你的测试数据应该包含随机数据，以及涉及边界情况的极端数据，例如排序算法要求 n <= 10,000，你的测试中应该包含 n = 10,000 的情况，同时除了随机数据以外，你还应该测试已经排好序，以及逆序的数据。评测系统内置了一组测试数据，主要用于辅助大家提交作业，以免出现因为开发环境、编译器版本不同等原因导致作业中的代码无法正常运行，难以评判作业成绩的问题。希望大家不要过度依赖评测系统，不要只依靠评测系统帮自己检查代码的正确性。在你之后的学习和工作中如果遇到实际的编程问题，很多时候都不会有人给你提供评测工具，不会帮你提供数据检查代码是否正确，通常需要你自己设计数据，验证代码的正确性。

3. 如果你已经找到了一组数据，发现你的代码在这组输入数据上无法正常运行，例如程序异常退出、程序长时间运行无输出、或者输出的结果不正确。这说明你的代码存在 bug，而这组数据能够重现这个 bug。那么应该如何找到 bug 的具体位置呢？核心的思路就是在代码中增加 printf 语句，把关键步骤相关的信息都输出出来，这样你就能够知道你的代码在什么位置停止了，或者在什么位置的逻辑有错误以及大致是什么样的错误。例如归并排序，你可以将每次均分两段的起止点输出，检查拆分过程是否正确；如果拆分正确，但是结果不正确，你可以检查合并前后的数据，看看是因为合并之前的数据没有排好序还是合并操作有问题。对于 dijkstra 算法，你可以将每次从 C 中找出的最短距离和对应的顶点标号输出，此外还可以将更新前后的距离和顶点编号输出，检查是否正确。在大致找到 bug 的位置之后，你可以有针对性的增加 printf 语句，输出更多的相关信息，定位 bug 的位置。可能有的同学习惯使用开发工具中的单步跟踪、断点、变量监视等功能，其实和上面的 printf 作用类似，你可以按照自己的习惯选择顺手的调试方法。这里介绍 printf 主要是考虑到这种方法简单通用，对于不同的语言和不同的开发环境普遍适用。

最后总结一下编程和调试的流程，首先参考第 1 点，写出质量较高的代码；然后参考第 2 点，准备测试数据对自己的代码进行测试，如果经过各种测试都没有问题，那么就可以尝试提交作业，如果在某组数据上遇到了问题，可以参考第 3 点，找出代码中的错误。