

数据结构与算法实验报告

实验五



学 生：

学 号：

年 级：2023

专 业：软件工程

重庆大学大数据与软件学院

2024 年 11 月 24 日

1. 实验目的：

练习 Dijkstra 算法的实现。
掌握调试程序的方法，跟踪程序的执行过程。

2. 实验要求

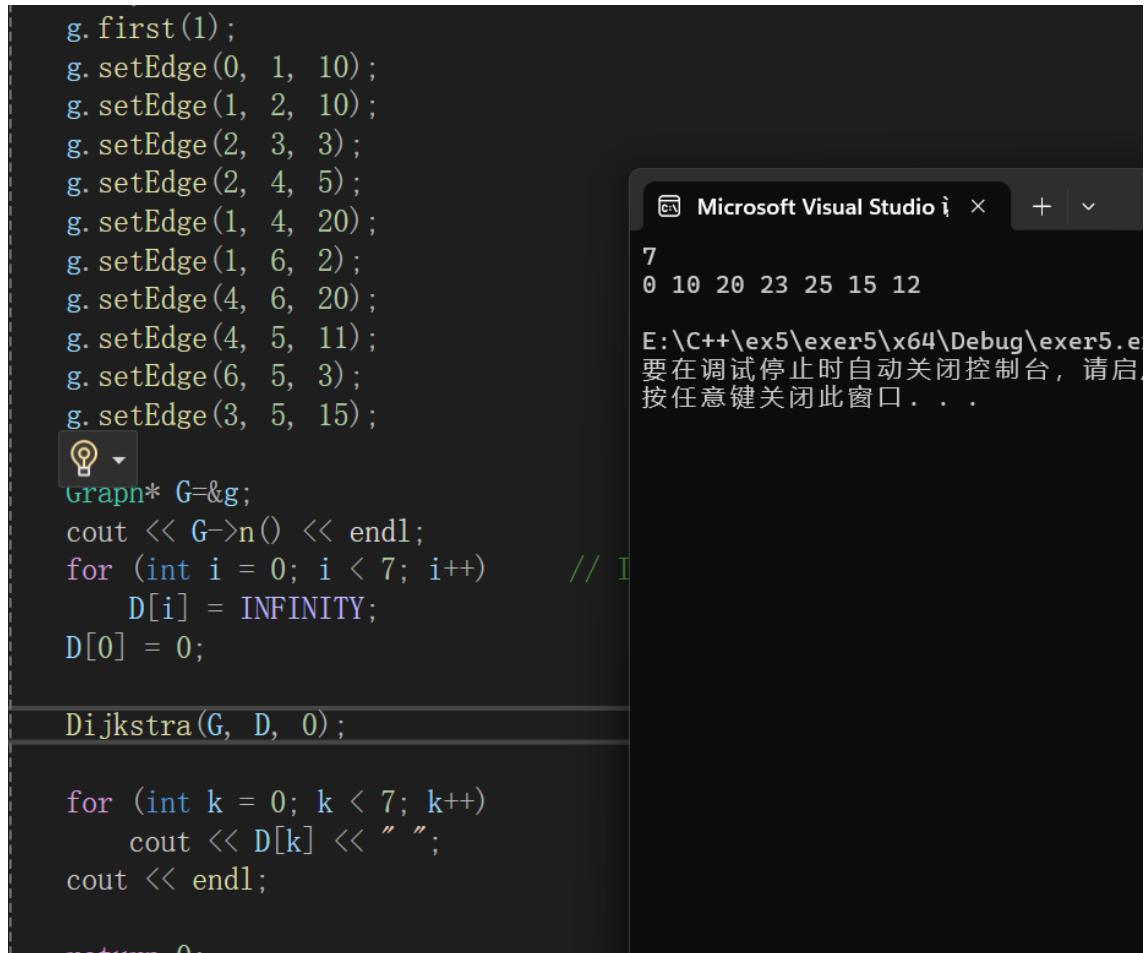
完成实验中要求的内容，实验报告书写格式尽量美观整洁。
所有程序需经过上机调试通过。
注意程序编写规范，如：必要的注释行，缩进排列等。
提交项目压缩包和实验报告。

3. 实验内容：

实现 Dijkstra 算法，并编写测试程序进行验证。

4. 核心代码和实验结果

代码见压缩包



The screenshot shows the Microsoft Visual Studio interface. On the right, the Output window displays the command-line output of a C++ program. The output shows the shortest distances from the source vertex (0) to all other vertices (1, 2, 3, 4, 5, 6). The distances are: 0, 10, 20, 23, 25, 15, 12. A message at the bottom of the window says: "要在调试停止时自动关闭控制台，请按任意键关闭此窗口..." (When the debug session ends, the console will close automatically. Please press any key to close this window...).

The code in the editor is a C++ implementation of the Dijkstra algorithm. It starts by defining a Graph class with a constructor g.first(). Then it initializes the graph with edges: (0, 1, 10), (1, 2, 10), (2, 3, 3), (2, 4, 5), (1, 4, 20), (1, 6, 2), (4, 6, 20), (4, 5, 11), (6, 5, 3), and (3, 5, 15). It then creates a Graph pointer G=&g and prints the number of vertices n. It initializes a distance array D[7] with INFINITY for all vertices except the source (0, 0). Finally, it calls the Dijkstra function with parameters G, D, and 0, and prints the resulting distances.

```
g.first();
g.setEdge(0, 1, 10);
g.setEdge(1, 2, 10);
g.setEdge(2, 3, 3);
g.setEdge(2, 4, 5);
g.setEdge(1, 4, 20);
g.setEdge(1, 6, 2);
g.setEdge(4, 6, 20);
g.setEdge(4, 5, 11);
g.setEdge(6, 5, 3);
g.setEdge(3, 5, 15);

Graph* G=&g;
cout << G->n() << endl;
for (int i = 0; i < 7; i++) // I
    D[i] = INFINITY;
D[0] = 0;

Dijkstra(G, D, 0);

for (int k = 0; k < 7; k++)
    cout << D[k] << " ";
cout << endl;

return 0;
```