1. 问题描述：

给定一个赋权无向图G=(V,E)，每个顶点v∈V都有一个权值w(v)。如果U∈V，且对任意（u,v）∈E有u∈U或v∈U，就称U为图G的一个顶点覆盖。G的最小权顶点覆盖是指G中所含顶点权之和最小的顶点覆盖。

1. 算法描述：

用优先队列分支限界方法解最小权顶点覆盖，在算法的搜索的进程中保存当前已构造出的部分解空间树，在算法搜索达到叶节点时，其最优值对应的最优解同时保存下来。优先队列的优先级为每个活节点的权值，权值最小的活节点成为下一个扩展节点。

设求解最大化问题，[解向量](https://baike.baidu.com/item/%E8%A7%A3%E5%90%91%E9%87%8F" \t "_blank)为X=(x1,…,xn)，xi的[取值范围](https://baike.baidu.com/item/%E5%8F%96%E5%80%BC%E8%8C%83%E5%9B%B4" \t "_blank)为Si，|Si|=ri。在使用[分支限界搜索](https://baike.baidu.com/item/%E5%88%86%E6%94%AF%E9%99%90%E7%95%8C%E6%90%9C%E7%B4%A2" \t "_blank)问题的解空间树时，先根据限界函数估算[目标函数](https://baike.baidu.com/item/%E7%9B%AE%E6%A0%87%E5%87%BD%E6%95%B0/10829077)的界[down, up]，然后从[根结点](https://baike.baidu.com/item/%E6%A0%B9%E7%BB%93%E7%82%B9/9795570)出发，扩展根结点的r1个孩子结点，从而构成分量x1的r1种可能的取值方式。

对这r1个孩子结点分别估算可能的目标函数bound(x1)，其含义：以该结点为根的子树所有可能的取值不大于bound(x1)，即：

bound(x1)≥bound(x1,x2)≥…≥ bound(x1,…,xn)

若某孩子结点的目标函数值超出目标函数的下界，则将该孩子结点丢弃；否则，将该孩子结点保存在待处理结点表PT中。

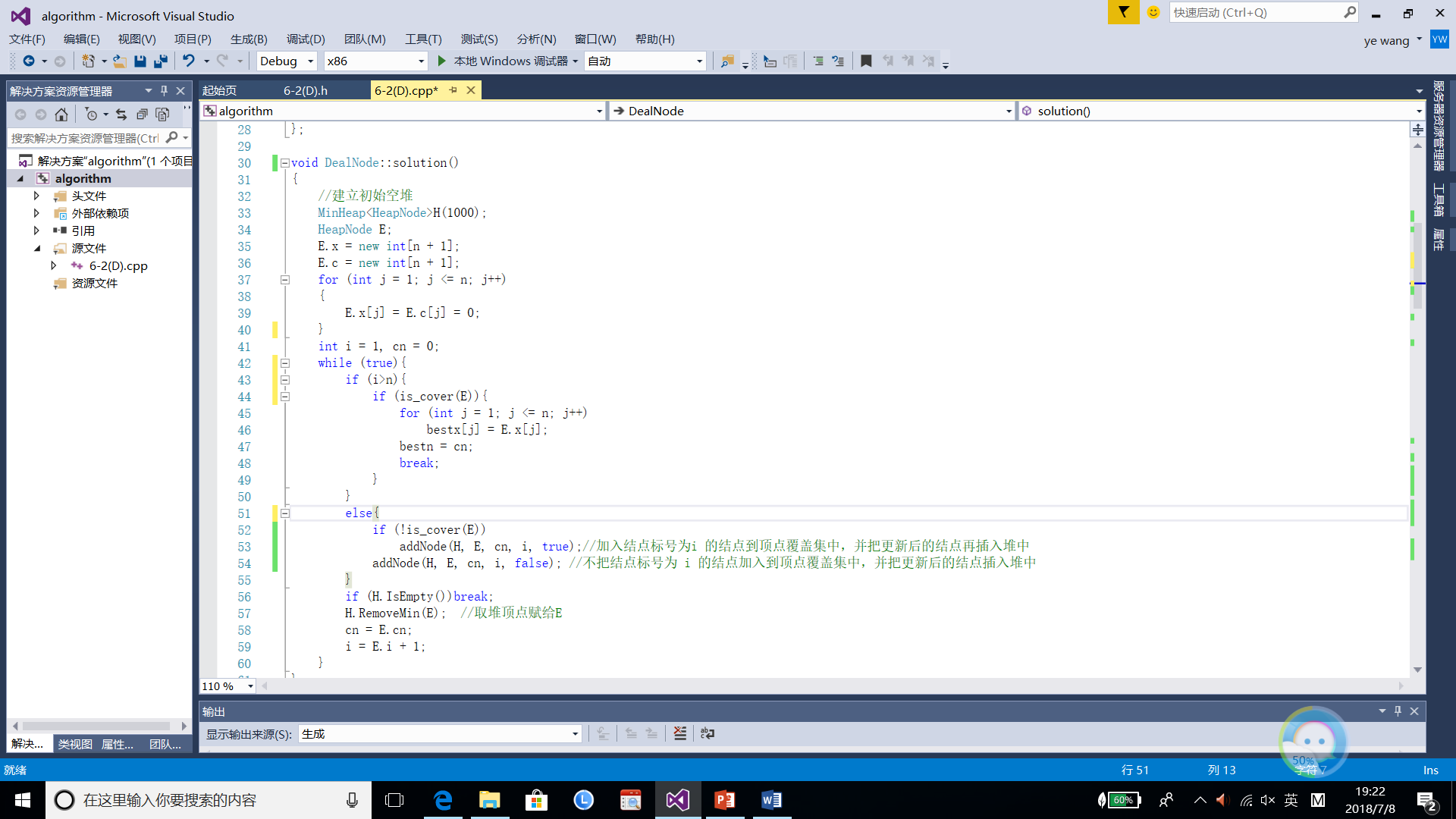
再取PT表中目标函数[极大值](https://baike.baidu.com/item/%E6%9E%81%E5%A4%A7%E5%80%BC/7667035" \t "_blank)结点作为扩展的[根结点](https://baike.baidu.com/item/%E6%A0%B9%E7%BB%93%E7%82%B9/9795570)，重复上述。

直到一个[叶子结点](https://baike.baidu.com/item/%E5%8F%B6%E5%AD%90%E7%BB%93%E7%82%B9" \t "_blank)时的[可行解](https://baike.baidu.com/item/%E5%8F%AF%E8%A1%8C%E8%A7%A3)X=(x1,…,xn)，及目标函数值bound(x1,…,xn) 。

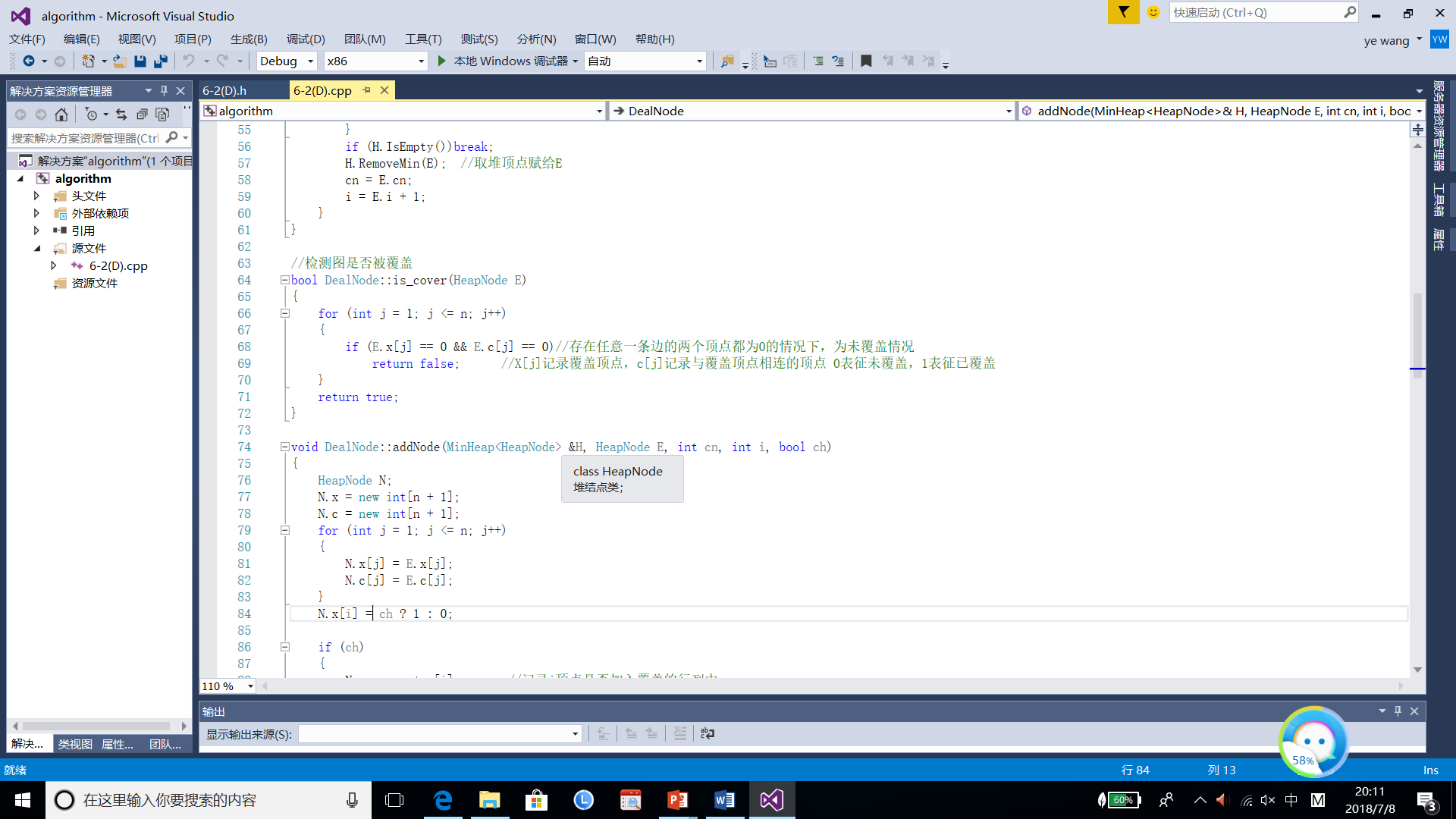
1. 代码展示

在头文件中是一套实现优先队列的函数。

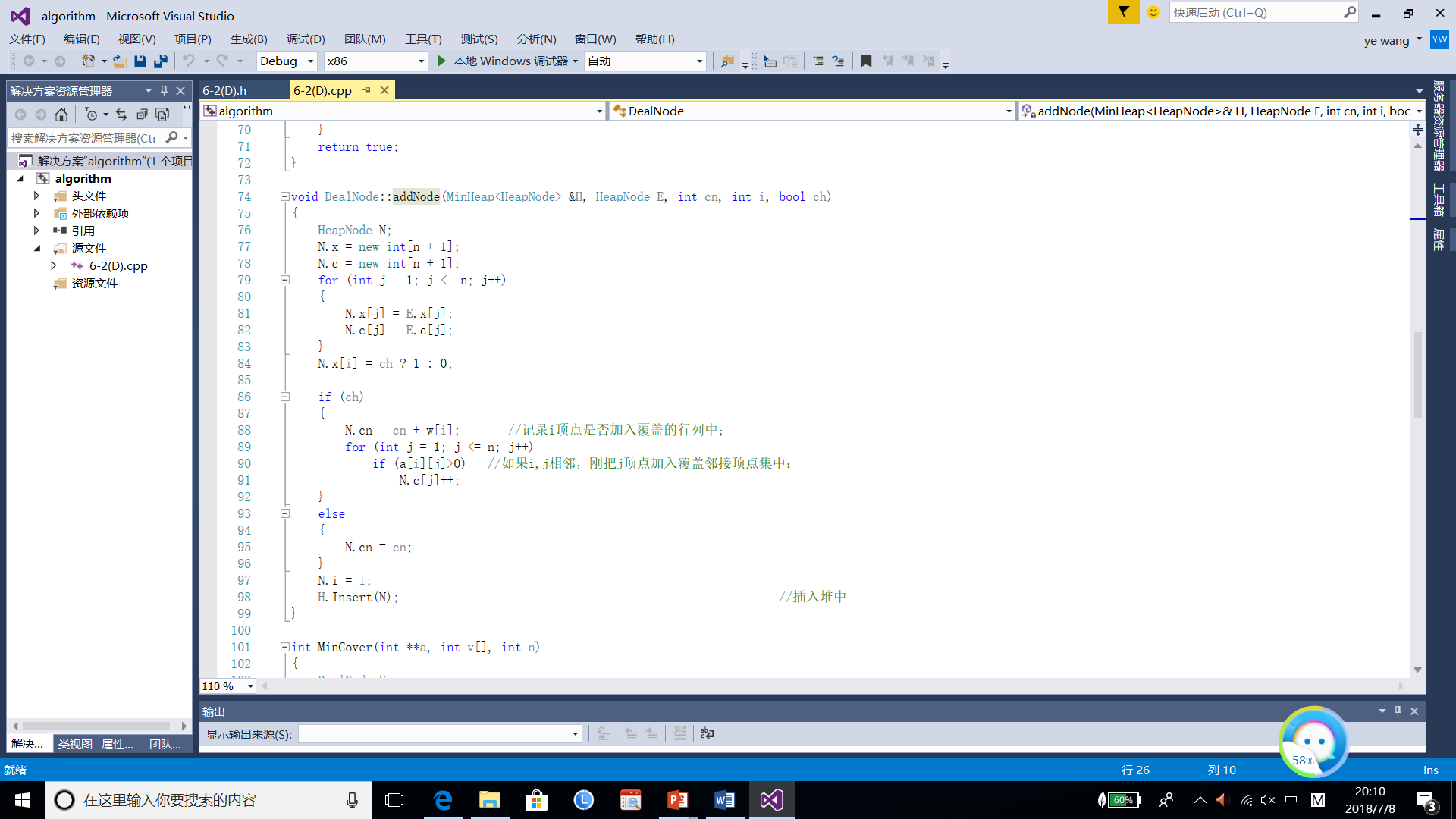
函数solution求取最小权顶点覆盖的最优值和最优解；



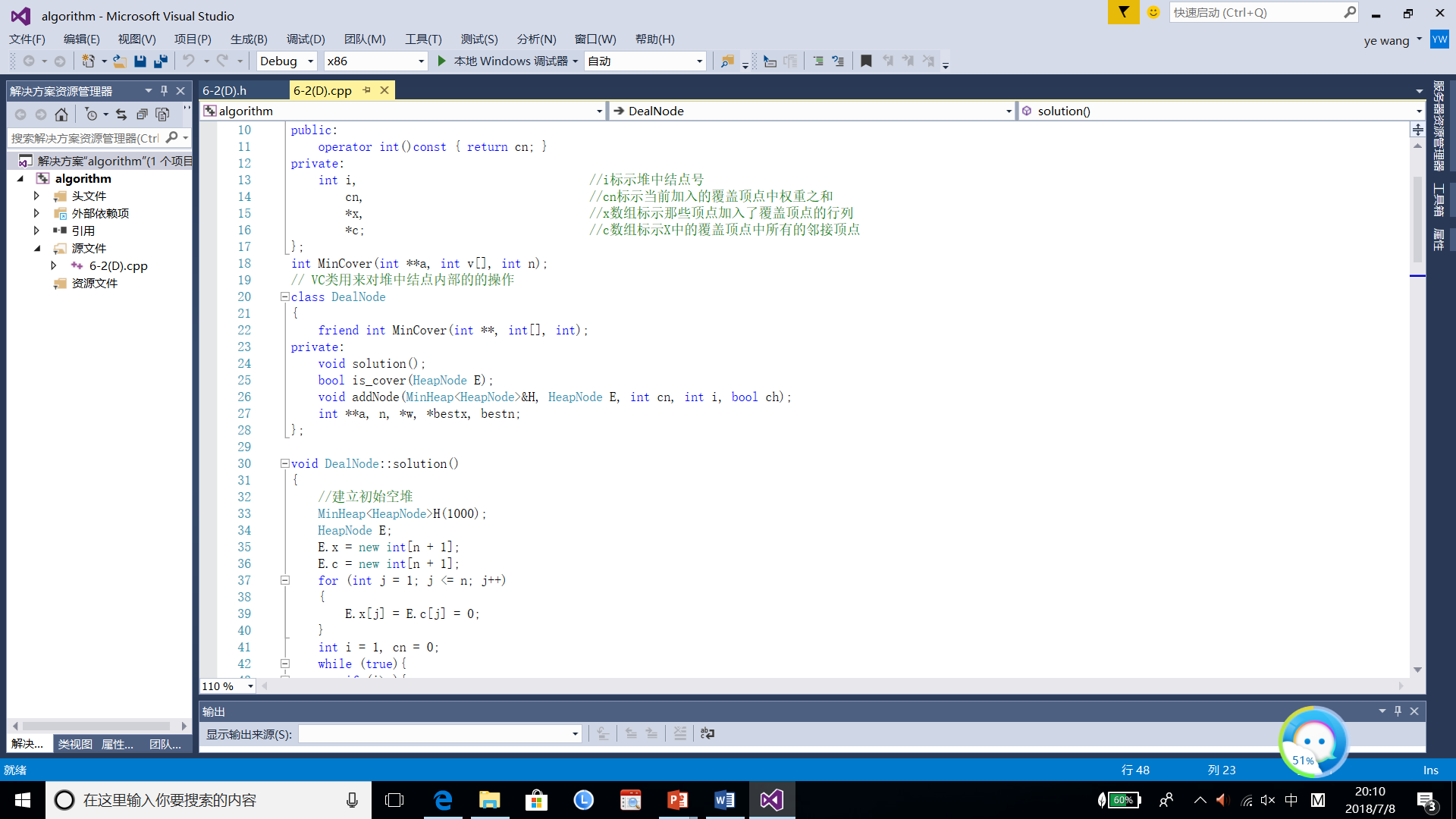
函数is\_cover判断图中顶点是否被完全覆盖；



算法addNode将产生的子节点加入到活节点优先队列的最小堆中；



函数MinCover返回最小权顶点覆盖的最优值和最优解。

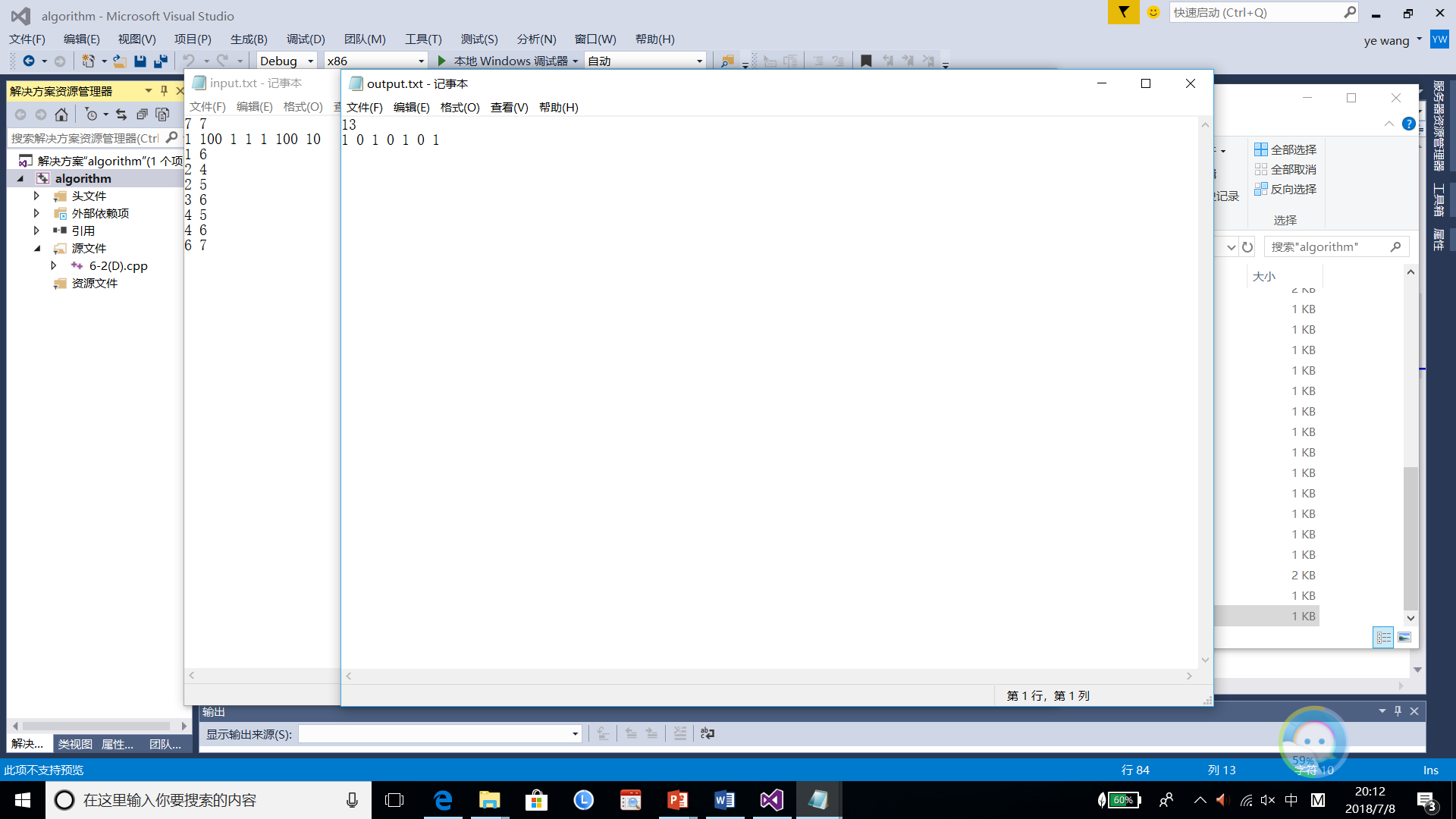


1. 时间复杂度分析

每一个节点均为二叉，最差情况下要完成所有节点的搜索，时间复杂度 即

算法空间存放优先队列，空间复杂度为

1. 结果



经验证结果正确