

1.题目名称

地铁修建

2.代码行数

69行

3.算法思想

并查集

利用并查集来判断两个节点是否已经在一个树中。

Krustal

利用Krustal算法生成最小生成树，当加入某一条边使得两个目标节点存在于同一颗树时直接返回该边的权值即可。

4.主要/核心函数分析

Krustal

```
1 void Krustal(){
2     Initial();
3     for(int i=0;i<m;i++){
4         int x= find(e[i].u);
5         int y= find(e[i].v);
6         if(x!=y){
7             if(Union(x,y)){
8                 if(find(1)== find(n)){ //找到最大边 直接退出
9                     cout<<e[i].w;
10                    break;
11                }
12            }
13        }
14    }
15 }
```

利用了Krustal算法的思想,边从小到大插入，因此可以直接找到题目要求的边。

5.测试数据(规模,测试次数)

规模: $1 \leq n \leq 100000$, $1 \leq m \leq 200000$, $1 \leq a, b \leq n$, $1 \leq c \leq 1000000$ 。

测试次数:自行测试1次+参考CSP官网该题测试次数。

测试用例:见自行输入

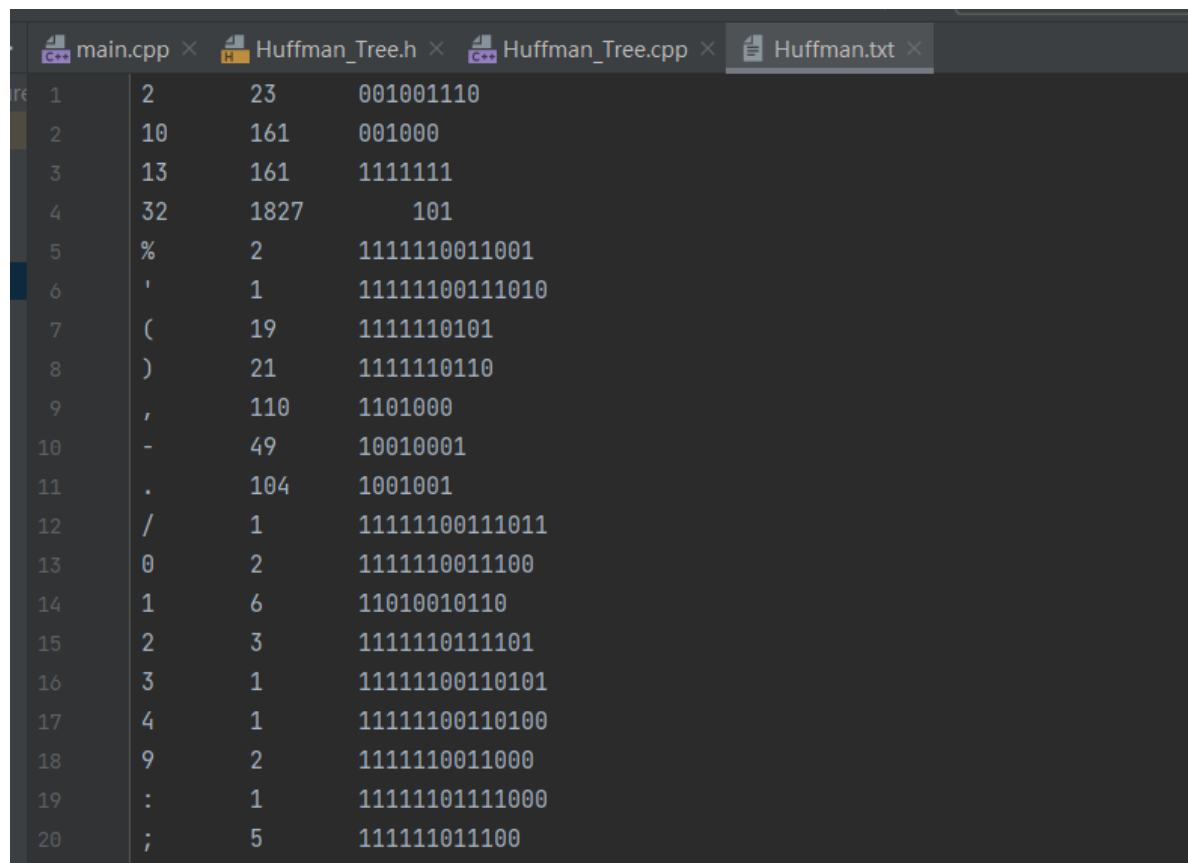
6.运行结果

```
1 F:\data_structure\Must\question_5\cmake-build-debug\question_5.exe
2 6 6
3 1 2 4
4 2 3 4
5 3 6 7
6 1 4 2
7 4 5 5
8 5 6 6
9 6
10 进程已结束,退出代码0
11
```

7.时间复杂度分析

该代码主要利用了Kruskal算法，时间复杂度为 $O(n \log n)$ 。

8.结果截屏图片



1	2	23	001001110
2	10	161	001000
3	13	161	1111111
4	32	1827	101
5	%	2	1111110011001
6	'	1	11111100111010
7	(19	1111110101
8)	21	1111110110
9	,	110	1101000
10	-	49	10010001
11	.	104	1001001
12	/	1	11111100111011
13	0	2	1111110011100
14	1	6	11010010110
15	2	3	1111110111101
16	3	1	11111100110101
17	4	1	11111100110100
18	9	2	1111110011000
19	:	1	11111101111000
20	;	5	111111011100

9.心得体会

对最小生成树以及Kruskal算法有了更为深入的理解。