算法基础 hw9

1. 正确的。

要实现 floyd 算法，必须生成正确的路径矩阵，必须正确计算每个点的最短路径长。

<1> k=0　　D=W

<2> k=k时　D中 d_{ij} 为节点 i 到 j 的最短长度

<3> k=k+1 时　在 k+1 次迭代中　D中的 $d_{ij} = \min(d_{ij}, d_{ik}+d_{kj})$

证 ① 路径不经过 k，则 d_{ij} 不变。

② 路径经过 k，则如果存在经过 k 的路径使比上次迭代中的路径更短，则通过 k 的路径被更新为最短路径。

综上，NEW 算法正确计算了 i×j 个节点的最短路径长，则正确生成路径长度矩阵。　正确。

2.

```
Shortest-Path (G, s, k)
    init  dist[] , path[]
    init  min-priority queue  Q.
    for v in G
        dist[v] = ∞
        path[v] = empty.
    dist[s] = 0
    put (s, 0) in Q.
```

While (Q)

u = Q中的最小距离的节点

for v in u. neighbour.

if (dist[u] + weight(u, v) < dist[v]

and edges(u, v) ≤ k)

dist[v] = dist[u] + weight(u, v)

path[v] = path[u] + [u, v]

在队列中 decrease key (v, dist[v])

return dist, path.

1> dist[v] 为源节点 s 到 v 的最短路径长度，初始化为 ∞

path[v] 为 s → v 的最短路径。初始化为空

Q 为最小优先队列。

将 s 放入 Q，距离为 0

2> 在队列空之前，重复：

取出最小距离节点 u.

对 u 的邻居节点 v（多个），若有更短路径，且加边后边数不超 k，则更新到 v 的 dist[v] 和 path[v]，更新 v 的距离（Q中）

最小优先队列操作次数为 O(|V|)，对边的遍历次数为 O(k|E|)，则

O(|V| + k|E|)

~~3. 对于环路 C 上的边 (u,v)，原始权重为 $w(u,v)$~~

~~则 $w(u,v)=w(C)=0$~~

~~引入新一个新顶点 s，使 s, u, v, s 为环。~~

~~$w(s,u)+w(u,v)+w(v,s)=0$~~

~~即 $w(s,u)+w(v,s)=0$~~

3. 对于环路 C 上的边 (u,v)

$\hat{w}(u,v)=w(u,v)+h(u)-h(v)$

取 C 上一点 s. $s\to\cdots\to u\to v$

$h(v)\le h(u)+w(u,v)$

由 $s\to\cdots\to u\to v\to\cdots\to s\to\cdots\to u$

$h(u)\le h(v)-w(u,v)\le h(u)$

若 $h(v)\neq h(u)+w(u,v)$ 则 $h(u)<h(u)$ ×

即 ~~$h(u)=h(u)$~~ $h(v)=h(u)+w(u,v)$

则 $\hat{w}(u,v)=w(u,v)+h(u)-h(u)-w(u,v)=0$

即 $\hat{w}(u,v)=0$

4. ① 维护一个优先队列，每个节点内容包含站点，体力值，到达时间和到达路径。

~~② 将初始状态站点s，体力值g，到达时间t=0~~

② 使（s，g，0，NULL）入队。

③ 使用 Dijkstra 算法循环处理队列中最小时间的节点，考虑相邻站点。

<1> x_i 到 x_j 上坡（$e_j > e_i$），则更新体力值为 now-(e_j-e_i)，更新到达时间

<2> x_i 到 x_j 下坡（$e_i > e_j$），则体力不变，更新到达时间。

<3> x_j 为补给，则休息并更新体力，更新时间为到达时间+k。

（更新体力用循环，由0到n循环，直到不丢弃路径）

更新时同时更新站点和路径。

若体力为0或负，则丢弃路径，若时间超过最快时间，也丢弃

④ 当t出现在优先队列，则找到。

优先队列操作为 $O(\log n)$ 处理边可能所有边，$O(E \log n)$

又 $E \le 5n$ $O(n \log n)$ 又在 Dijkstra 中对休息站休息时间 $O(n)$

则 $O(n^2 \log n)$