# 综合练习

## 人员

赵熙羽 到课, 杨瑾硕、于子珈、周子一 线上

## 上周作业检查

上周作业链接: https://cppoj.kids123code.com/contest/2454


| # | 用户名 | 姓名 | 编程分 | 时间 | A | B | C |
|---|---|---|---|---|---|---|---|
| 1 | yuzijia1 | 于子珈 | 300 | 1559 | 100 | 100 | 100 |
| 2 | zhaoxiyu | 赵熙羽 | 300 | 2166 | 100 | 100 | 100 |
| 3 | qinxiansen | 秦显森 | 300 | 2537 | 100 | 100 | 100 |
| 4 | lizihan | 李子瀚 | 200 | 1529 | 100 | 100 | |
| 5 | chenluoran | 陈洛冉 | 134 | 509 | 100 | 34 | |
| 6 | zhouziyi | 周子一 | 100 | 245 | 100 | | |
| 7 | yangjinshuo | 杨谨硕 | 100 | 270 | 100 | | 0 |

## 本周作业

https://cppoj.kids123code.com/contest/2584 (课上讲了 A ~ C 题, 课后作业是 C 题)

## 课堂表现

今天的第三题建图会相对复杂一些, 需要同学们拿着笔多画一画, 搞清楚建图的逻辑, 才能做这道题。

## 课堂内容

**小 K 的农场 (上周作业)**

按照题目的要求建边, 可以把原问题转化为一个差分约束问题

```cpp
#include <bits/stdc++.h>

using namespace std;

const int maxn = 5e3 + 5;
const int inf = 0x3f3f3f3f;
struct node {
  int to, value;
};
vector<node> vec[maxn];
int dis[maxn], cnt[maxn];
bool st[maxn];
```

```cpp
int main()
{
  int n, m; cin >> n >> m;
  while (m -- ) {
    int op; cin >> op;
    if (op == 1) {
      int a, b, c; cin >> a >> b >> c;
      vec[a].push_back({b,-c});
    } else if (op == 2) {
      int a, b, c; cin >> a >> b >> c;
      vec[b].push_back({a,c});
    } else {
      int a, b; cin >> a >> b;
      vec[a].push_back({b,0}), vec[b].push_back({a,0});
    }
  }
  for (int i = 1; i <= n; ++i) vec[0].push_back({i,0});

  memset(dis, 0x3f, sizeof(dis));
  queue<int> q; q.push(0); dis[0] = 0; st[0] = true;
  while (!q.empty()) {
    int u = q.front(); q.pop();
    st[u] = false;
    for (node it : vec[u]) {
      if (dis[u]+it.value < dis[it.to]) {
        dis[it.to] = dis[u]+it.value; ++cnt[it.to];
        if (cnt[it.to] == n) { cout << "No" << endl; return 0; }
        if (!st[it.to]) { q.push(it.to); st[it.to] = true; }
      }
    }
  }

  cout << "Yes" << endl;
  return 0;
}
```

**[CSP-J 2019 江西] 道路拆除**

从 A 点、s1 点、s2 点 分别跑 bfs, 然后枚举一个中间 i 点, 找 d[i]+d1[i]+d2[i] 的最小值

```cpp
#include <bits/stdc++.h>

using namespace std;

const int maxn = 3000 + 5;
const int inf = 0x3f3f3f3f;
vector<int> vec[maxn];
int d[maxn], d1[maxn], d2[maxn];

void bfs(int id, int a[]) {
```

```
  for (int i = 0; i < maxn; ++i) a[i] = 0x3f3f3f3f;
  queue<int> q; q.push(id); a[id] = 0;
  while (!q.empty()) {
    int u = q.front(); q.pop();
    for (int i : vec[u]) {
      if (a[u]+1 < a[i]) {
        a[i] = a[u]+1; q.push(i);
      }
    }
  }
}

int main()
{
  int n, m; cin >> n >> m;
  for (int i = 1; i <= m; i++) {
    int u, v; cin >> u >> v;
    vec[u].push_back(v), vec[v].push_back(u);
  }

  int s1, t1, s2, t2; cin >> s1 >> t1 >> s2 >> t2;
  bfs(1, d), bfs(s1, d1), bfs(s2, d2);

  int res = inf;
  for (int i = 1; i <= n; ++i) {
    if (d[i]+d1[i]<=t1 && d[i]+d2[i]<=t2) {
      res = min(res, d[i]+d1[i]+d2[i]);
    }
  }
  if (res == inf) cout << -1 << endl;
  else cout << m - res << endl;
  return 0;
}
```

**[CQOI2005] 新年好**

以 1,a,b,c,d,e 这 6 个点为起点每个点跑一遍 bfs, 然后全排列 a,b,c,d,e 这 5 个点, 看怎么走最近

```
#include <bits/stdc++.h>

using namespace std;

typedef long long LL;
const int maxn = 50000 + 5;
const int inf = 0x3f3f3f3f;
struct node {
  int to, value;
  bool operator > (const node& p) const { return value > p.value; }
};
vector<node> vec[maxn];
int a[10], dis[10][maxn];
```

```
bool st[maxn];

void bfs(int id) {
  memset(st, false, sizeof(st));
  priority_queue<node, vector<node>, greater<node>> q;
  q.push({a[id], 0}); dis[id][a[id]] = 0;

  while (!q.empty()) {
    node u = q.top(); q.pop();
    int u_id = u.to, u_dis = u.value;
    if (st[u_id]) continue;
    st[u_id] = true;

    for (node it : vec[u_id]) {
      int to = it.to, value = it.value;
      if (!st[to] && u_dis+value<dis[id][to]) {
        dis[id][to] = u_dis+value; q.push({to, dis[id][to]});
      }
    }
  }
}

int main()
{
  int n, m; cin >> n >> m;
  map<int, int> mp;
  a[1] = 1; mp[1] = 1;
  for (int i = 2; i <= 6; ++i) cin >> a[i], mp[a[i]] = i;

  for (int i = 1; i <= m; ++i) {
    int u, v, w; cin >> u >> v >> w;
    vec[u].push_back({v,w}), vec[v].push_back({u,w});
  }

  memset(dis, 0x3f, sizeof(dis));
  for (int i = 1; i <= 6; ++i) bfs(i);

  LL res = 1e18;
  sort(a+2, a+6+1);
  do {
    LL sum = 0;
    for (int i = 2; i <= 6; ++i) sum += dis[mp[a[i-1]]][a[i]];
    res = min(res, sum);
  } while (next_permutation(a+2, a+6+1));
  cout << res << endl;
  return 0;
}
```

**[SHOI2012] 回家的路**

分层图, 把每个点拆分成两个点, 一个为横向的点, 一个为竖向的点

横向的点之间每行彼此建边, 竖向的点之间每列彼此建边, 同一个点的横线与竖向也建边

然后从起点往终点跑最短路即可

```
#include <bits/stdc++.h>

using namespace std;

const int maxn = 2e5 + 100;
const int inf = 0x3f3f3f3f;

int getId(int id, int c) { return id*2 + c; }
struct Point {
  int x, y;
} w[maxn];
struct Info {
  int id, c;
  bool operator < (const Info& p) const { return c < p.c; }
};
vector<Info> row[maxn], col[maxn];

struct node {
  int to, value;
  bool operator > (const node& p) const { return value > p.value; }
};
vector<node> vec[maxn];
int dis[maxn];
bool st[maxn];

int dijkstra(int _st, int _ed) {
  memset(dis, 0x3f, sizeof(dis));
  priority_queue<node, vector<node>, greater<node>> q; q.push({_st,0}); dis[_st] =
0;
  while (!q.empty()) {
    node u = q.top(); q.pop();
    int u_id = u.to, u_dis = u.value;
    if (st[u_id]) continue;
    st[u_id] = true;

    for (node it : vec[u_id]) {
      if (u_dis+it.value < dis[it.to]) {
        dis[it.to] = u_dis+it.value; q.push({it.to, dis[it.to]});
      }
    }
  }

  return (dis[_ed]==inf ? -1 : dis[_ed]);
}

int main()
{
  int n, m; cin >> n >> m;
  for (int i = 1; i <= m+2; ++i) {
```

```
    int x, y; cin >> x >> y; w[i] = {x, y};
    row[x].push_back({i,y}), col[y].push_back({i,x});
  }

  for (int i = 1; i <= n; ++i) {
    sort(row[i].begin(), row[i].end());
    int len = row[i].size();
    for (int j = 1; j < len; ++j) {
      Info a = row[i][j-1], b = row[i][j];
      int a_id = getId(a.id,1), b_id = getId(b.id,1);
      vec[a_id].push_back({b_id,(b.c-a.c)*2}), vec[b_id].push_back({a_id,(b.c-
a.c)*2});
    }
  }

  for (int j = 1; j <= n; ++j) {
    sort(col[j].begin(), col[j].end());
    int len = col[j].size();
    for (int i = 1; i < len; ++i) {
      Info a = col[j][i-1], b = col[j][i];
      int a_id = getId(a.id,2), b_id = getId(b.id,2);
      vec[a_id].push_back({b_id,(b.c-a.c)*2}), vec[b_id].push_back({a_id,(b.c-
a.c)*2});
    }
  }

  for (int i = 1; i <= m+2; ++i) {
    int id1 = getId(i,1), id2 = getId(i,2);
    vec[id1].push_back({id2,(i<=m)}), vec[id2].push_back({id1,(i<=m)});
  }

  cout << dijkstra(getId(m+1,1), getId(m+2,2)) << endl;
  return 0;
}
```