10.图和树基础

1、邻接矩阵的实现

```
#include <iostream>
#include <cstring>

using namespace std;
int main()
{
  int G[6][6];
  memset(G,0,sizeof(G));
  int m;
  cin >> m;
  for (int i = 0; i < m ;i++)
  {
    int a,b;
    cin >> a >> b;
    G[a][b] = 1;
  }
  for (int i = 1; i <= 5; i ++)
  {
    int sum = 0;
    for (int j = 1; j <= 5; j ++)
    {
      if (G[i][j] == 1 && G[j][i] == 1)
      {
        sum++;
      }
    }
    cout << i << "有" << sum <<"个真正的朋友"<<endl;
  }
  return 0;
}
```

2、邻接表的实现

```
#include <iostream>
#include <vector>

using namespace std;
int main()
{
    vector<int> G[11];
```

```
    int m;
    cin >> m;
    for (int i = 0; i < m; i++)
    {
        int a,b;
        cin >> a >> b;
        G[a].push_back(b);
        G[b].push_back(a);
    }
    for(int i = 1; i <= 10; i++)
    {
        cout << i << ":";
        for (int j = 0; j < G[i].size();j++)
        {

            cout << G[i][j] << " ";
        }
        cout <<endl;
    }

  return 0;
}
```

3、带权值的图

```
#include <iostream>
#include <vector>

using namespace std;

struct node
{
    int v,w;
};
vector<node> G[11];
void insert1(int u, int v, int w)
{
    node temp;
    temp.v = v;
    temp.w = w;
    G[u].push_back(temp);
}
void insert2(int u, int v, int w)
{
    insert1(u,v,w);
    insert1(v,u,w);
}

int main()
{
    int m;
```

```
    cin >> m;
    for (int i = 0; i < m; i++)
    {
        int u,v,w;
        cin >> u >> v >> w;
        insert2(u,v,w);
    }
    for(int i = 1; i <= 10; i++)
    {

        for (int j = 0; j < G[i].size();j++)
        {
             cout << "("<< i << "," << G[i][j].v << "," <<G[i][j].w <<")" <<
        }
    }

  return 0;
}
```

4、 关系查询

```
#include <iostream>
#include <map>

using namespace std;
int G[500][500];
map<string,int> dict;
int ids;
int find(string a)
{
    if(dict.find(a) == dict.end())
    {
        dict[a] = ++ ids;
    }
    else
    {
        return dict[a];
    }
}

int main()
{
    memset(G,0,sizeof(G));
    int n;
    for (int i = 0; i < n; i ++)
    {
        string a, b;
        cin >> a >> b;
        int x = find(a);
        int y = find(b);
        G[x][y] = G[y][x] = 1;
```

```
    }
    return 0;
}
```