```
//silnie spojne skladowe
//Rafał Pytko
//www.algorytm.org (c) 2009

#include <cstdio>
#include <set>
#include <list>
using namespace std;

const int max_n = 10000;
list<int> gS[max_n];          // wyjscie: graf SSS
int sss[max_n];               // wyjscie: numer SSS wierzcholka i
int ns;                       // wyjscie: ilosc SSS
int sssF[max_n],ttt;
bool sssvis[max_n];

list<int> adj[max_n];         // wejscie: graf
list<int> adjT[max_n];        // wejscie: graf transponowany
int n;                        // wejscie: ilosc wierzcholkow

void sss_dfs(int v) {
    sssvis[v] = 1;
    for(list<int>::iterator it=adj[v].begin(); it != adj[v].end();
       ++it)
      if(!sssvis[*it]) sss_dfs(*it);
    sssF[ttt++] = v;
}

void sss_dfs2(int v) {
    sss[v] = ns;
    sssvis[v] = 1;
    for(list<int>::iterator it=adjT[v].begin(); it !=
       adjT[v].end(); ++it)
      if(!sssvis[*it]) sss_dfs2(*it);
}

void compute_sss() {
    for(int i = 0; i < n; ++i) {
        sssvis[i] = 0;
        gS[i].clear();
    }
    ttt=0;
    for(int i = 0; i < n; ++i)
      if(!sssvis[i]) sss_dfs(i);
    for(int i = 0; i < n; ++i)
      sssvis[i] = 0;
    ns = 0;
    for(int i = ttt-1; i >= 0; --i)
```

```
      if(!sssvis[sssF[i]]) {
        sss_dfs2(sssF[i]);
        ++ns;
      }

}

void create_sss_graph() {
    // tworzenie grafu silnie spójnych składowych
    set<pair<int,int> > W;
    for(int i = 0; i < n; ++i)
      for(list<int>::iterator it=adj[i].begin(); it !=
         adj[i].end(); ++it)
        if(sss[i] != sss[*it]) {
          if(W.find(make_pair(sss[i],sss[*it])) == W.end()) {
              gS[sss[i]].push_back(sss[*it]);
              W.insert(make_pair(sss[i],sss[*it]));
          }
        }
}

void transpose_graph() {
  for(int i = 0; i < n; ++i)
    for(list<int>::iterator it = adj[i].begin(); it !=
       adj[i].end(); ++it)
      adjT[*it].push_back(i);
}

int main() {
  n = 7; // liczba wierzcholkow

  // graf jako lista sasiadow
  adj[0].push_back(1);
  adj[1].push_back(2);
  adj[2].push_back(3);
  adj[3].push_back(1);
  adj[3].push_back(4);
  adj[3].push_back(6);
  adj[4].push_back(5);
  adj[5].push_back(4);
  adj[6].push_back(1);

  // obliczamy graf transponowany
  transpose_graph();

  // obliczamy silnie spojne skladowe
  compute_sss();
  printf("Liczba SSS: %d\n", ns);
```

```
  for(int i = 0; i < n; ++i) {
    printf("wierz: %d, skladowa: %d\n", i+1, sss[i]+1);
  }

  // wypisywanie skladowych
  for(int i = 0; i < ns; ++i) {
    printf("Skladowa %d: ", i+1);
    for(int v = 0; v < n; ++v)
      if(sss[v] == i) {
        printf("%d, ", v+1);
      }
    printf("\n");
  }

  // tworzymy graf SSS
  create_sss_graph();
  printf("\n\ngraf SSS (lista sasiadow)\n");
  for(int i = 0; i < ns; ++i) {
    printf("%d: ", i+1);
    for(list<int>::iterator it = gS[i].begin(); it != gS[i].end();
       ++it)
      printf("%d, ", *it + 1);
    printf("\n");
  }
}
```