**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

Факультет прикладной математики и информатики

Лабораторная работа №3 по курсу “ИСО”

Вариант №8

Выполнил: Ёда Никита

3 курс, 6 группа

Преподаватель: Лепин В.В., Кваша Д.Ю.

2024

**Задача 8. Станки**

Конвейер состоит из N различных станков. Есть N рабочих. Известна матрица C размера N × N, где элемент cij задаёт производительность i-го рабочего на j-м станке. Необходимо определить, каким должно быть распределение рабочих по станкам (каждый рабочий может быть назначен только на один станок, на каждом станке может работать только один рабочий), чтобы производительность всего конвейера была максимальной. Производительность конвейера при некотором распределении рабочих по станкам равна минимальной производительности рабочих на назначенных им на конвейере станках. Если решение не единственно, вывести решение, первое в лексикографическом порядке среди всех решений.

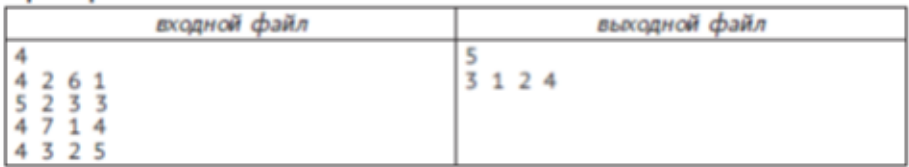
**Формат входных данных**

Первая строка содержит число рабочих (станков) N (1 ≤ N ≤ 500). Затем идут N строк файла, которые задают матрицу C производительностей (1 ≤ cij ≤ 1000).

**Формат выходных данных**

В первой строке выведите максимальную возможную производительность конвейера. Во второй строке — номера станков, на которые должны быть распределены рабочие 1, 2, . . . , N соответственно.

**Пример**



**Содержимое *“input.txt”* файла**

|  |
| --- |
| 4  4 2 6 1  5 2 3 3  4 7 1 4  4 3 2 5 |

**Код:**

|  |
| --- |
| #include <iostream>  #include <fstream>  #include <vector>  using namespace std;  ifstream fin("input.txt");  ofstream fout("output.txt");  int n;  int border;  vector<int> mt;  vector<bool> used;  vector<bool> block;  vector<vector<int>> vec;  int result[256];  int e[256][256];  int parent[256];  int mtr[256][256];  bool try\_kuhn(int v) {  if (used[v])  {  return false;  }  used[v] = true;  for (int i = 0; i < vec[v].size(); i++)  {  int to = vec[v][i];  if (mt[to] == -1 || try\_kuhn(mt[to]))  {  mt[to] = v;  return true;  }  }  return false;  }  bool function(int m)  {  vec.clear();  for (int i = 0; i < n; i++)  {  vector<int> cur;  for (int j = 0; j < n; j++)  {  if (mtr[i][j] >= m) cur.push\_back(j);  }  vec.push\_back(cur);  }  mt.assign(n, -1);  for (int v = 0; v < n; ++v)  {  used.assign(n, false);  try\_kuhn(v);  }  int ans = 0;  for (int i = 0; i < n; ++i)  {  if (mt[i] != -1) ans++;  }  if (ans == n)  {  return true;  }  else  {  return false;  }  }  void dfs(int v, int p)  {  if (used[v])  {  return;  }  used[v] = true;  parent[v] = p;  for (int i = 0; i < 2 \* n; i++)  {  if (e[v][i] == 1 && !block[i]) dfs(i, v);  }  }  void findLex(int v)  {  for (int i = 0; i < n; i++)  {  e[i][result[i] + n] = 1;  }  used.assign(2 \* n, false);  for (int i = 0; i < n; i++)  {  parent[i] = -1;  }  dfs(v, v);  for (int i = 0; i < n; i++)  {  e[i][result[i] + n] = 0;  }  for (int i = n; i < 2 \* n; i++)  {  if (used[i] && !block[i] && mtr[v][i - n] >= border)  {  block[i] = true;  result[v] = i - n;  int u = parent[i];  while (u != parent[u])  {  result[u] = parent[u] - n;  u = parent[parent[u]];  }  break;  }  }  block[v] = true;  }  void lex()  {  block.assign(2 \* n, false);  for (int i = 0; i < n; i++)  {  result[mt[i]] = i;  }  for (int i = 0; i < n; i++)  {  for (int j = 0; j < n; j++)  {  if (mtr[i][j] >= border) e[n + j][i] = 1;  }  }  for (int i = 0; i < n; i++)  {  findLex(i);  }  for (int i = 0; i < n - 1; i++)  {  fout << result[i] + 1 << " ";  }  fout << result[n - 1] + 1 << endl;  }  int main()  {  fin >> n;  int left = 1;  int right = 0;  for (int i = 0; i < n; i++)  {  for (int j = 0; j < n; j++)  {  fin >> mtr[i][j];  if (mtr[i][j] > right)  {  right = mtr[i][j];  }  }  }  while (right - left > 1)  {  int mid = (left + right) / 2;  if (function(mid))  {  left = mid;  }  else  {  right = mid;  }  }  if (function(right))  {  border = right;  }  else  {  function(left);  border = left;  }  fout << border << endl;  lex();  } |

**Результат:**

|  |
| --- |
|  |

**Содержимое *“output.txt”* файла**

|  |
| --- |
| 5  3 1 2 4 |