Задача С2. ПЛАТКА

Пешо, нашият прочут програмист, отишъл да учи в технически университет. Една от курсовите му задачи била да поправи една счупена платка. Платката представлявала точки, свързани с пътечки от алуминий, направени от поялник. За нещастие тези пътечки били разкъсани на някои места и за да я поправи трябвало да мине повторно с поялника върху тях. Платката работи само, ако всички точки са свързани с една непрекъсната пътечка от алуминий и заради това Пешо трябва да си избере една от точките, да сложи поялника на нея и без да го вдига да обходи всички останали точки. Проблемът бил, че ако мине два пъти по една и съща връзка между две точки – тя се повреждa. Помогнете на Пешо да се справи с тази трудна задача, като напишете програма **`platka`**, която по зададени връзките между точките, извежда реда, в който Пешо трябва да обходи точките.

Програмата приема входните данни от стандартния си вход. На първия ред са записани две числа N ($1 \leq N \leq 10000$) – броя на точките по платката и M ($1 \leq M \leq 100000000$) – броя на съществуващите пътечки между точките. Следващите M реда съдържат също по две числа – номерата на точките, свързани с пътечка.

Програмата трябва да изведе на стандартния изход на един ред начина на обхождане на точките, така, че след това платката да работи. Ако не съществува начин, при който да се обходят всички точки наведнъж, да се изведе съобщението “Sorry, Pesho”, следвано от броя на точките, който няма да бъдат достигнати, тръгвайки от избраната точка.

Пример:

Вход:
```
6 10
1 2
1 3
1 4
2 3
2 5
3 4
3 5
4 5
4 6
5 6
```

Изход:
```
1 3 5 6 4 1 2 3 4 5 2
```

Вход:
```
6 6
1 2
1 3
1 4
2 3
3 4
5 6
```

Изход:
```
Sorry, Pesho 2
```

Решение:

Решението на задачата прилага известния алгоритъм за намиране на Ойлеров път. (описанието на този алгоритъм може да се намери в различни книги за алгоритми). Единственото “усложнение” е изиксването за проверка дали графът има повече от една компонента:

```
#include <iostream>
using namespace std;

int a[10000][10000];
```

```
int p[10000], vis[10000];
int n, m, u, v;

void read_data()
{
     cin >> n >> m;
     for (int i = 1; i <= m; i++)
     {
         cin >> u >> v;
         a[u][v] = 1;
         a[v][u] = 1;
         a[u][0]  =  !a[u][0];  //контролира  се  четността  на
върховете
         a[v][0] = !a[v][0]; //за четен връх – 0, а за нечетен –
1
     }
}

void dfs(int u) // за намиране броя на свързаните компоненти
{
     vis[u] = 1;
     for (int i = 1; i <= n; i++)
         if (a[u][i] && !vis[i])
            dfs(i);
}

void Euler()
{
     p[0] = u; //слага се първия връх в началото на пътя
     int l = 0, k = m; //l – края на началото, k – началото на
края
     while (l < k) //докато не се срещнат
     {
           u  =  p[l];  //взема  се  последния  връх  от  началото  на
пътя
           v = 1;
           while (v <= n && !a[u][v])
             //търси се има ли на къде да се ходи
                 v++; //цикъла спира или когато сме намерили
           if (v <= n) //съсед (v не е минало n)
           { //тогава
                 l++;  //добавяме  намерения  съсед  в  началото  на
пътя
                 p[l] = v;
                 a[u][v] = 0; //премахваме връзките
                 a[v][u] = 0; //между двата върха
                 }
           else // ако сме обходили всички върхове
                 // и не сме намерили съсед
           {
               p[k] = u; //тогава този връх се маха от началото
               l--; k--; //и се слага в края на пътя
```

```
            }
       }
}

void print_data() // отпечатва пътя
{
     for (int i = 0; i <= m; i++)
         cout << p[i] << " ";
 }

int main()
{
    read_data();
    int i, br = 0;
    for (i = 1; i <= n; i++)
      if (a[i][0]) { u = i; break;} // намира се първия нечетен
връх
   dfs(u);
        for (i = 1; i <= n; i++)
      if (!vis[i]) br++;
     //преброяват се всички необходени от DFS върхове
     if (!br) //ако няма такива
     {
           Euler(); //се намира ойлеровия път
           print_data(); // и се отпечатва
     }
     else
           cout << "Sorry, Pesho " << br << endl;
           //иначе се отпечатва броя
    return 0;
}
```

Антония Йовчева

Задача С3. DARTS

Дартс е игра, в която играчите хвърлят къси стрели в кръгла мишена, окачена на стената. Играта се е зародила преди няколко столетия на Британските острови. И досега дартс е традиционна игра във Великобритания, Нидерландия, Скандинавските страни, Съединените Щати и др.

Стандартната дъска за дартс е квадрат със страна 48 cm, центърът на който съвпада с центъра на мишената – окръжност с радиус 20 cm. Ми-