Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής & Υπολογιστών
https://courses.softlab.ntua.gr/progtech/

Προγραμματιστικές Τεχνικές

Άσκηση 10 Μονοπάτια και κύκλοι Euler

Προθεσμία υποβολής στον grader: 14/5/2021

Δίνεται ένας μη κατευθυνόμενος γράφος N κορυφών (αριθμημένων από 0 έως $N-1$) και M ακμών. Θα είναι $2 \leq N \leq 10.000$ και $0 \leq M \leq 100.000$. Είναι πιθανό κάποια ακμή να υπάρχει πολλές φορές, όπως επίσης και να υπάρχουν κυκλικές ακμές — δηλαδή ακμές της μορφής (u, u). Θεωρήστε δεδομένο ότι ο γράφος θα είναι συνεκτικός, δηλαδή ότι θα είναι δυνατή η μετακίνηση από οποιαδήποτε κορυφή σε οποιαδήποτε άλλη, μέσω κάποιου μονοπατιού αποτελούμενου από ακμές.

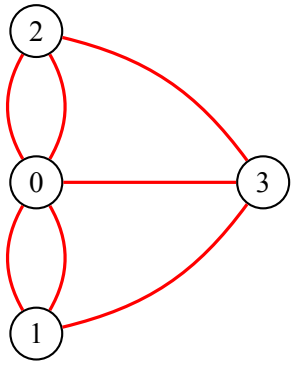
Γράψτε ένα πρόγραμμα που να διαβάζει τον γράφο από το standard input και να ελέγχει υπάρχει κύκλος ή μονοπάτι Euler στον γράφο. Συγκεκριμένα, το πρόγραμμά σας πρέπει να εκτυπώνει στο standard output μία γραμμή που να περιέχει ένα από τα εξής τρία:

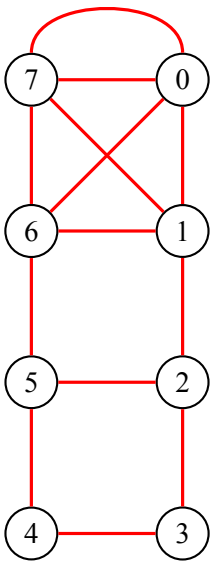
- "`CYCLE`", αν υπάρχει κύκλος Euler.
- "`PATH u v`", αν υπάρχει δεν υπάρχει κύκλος Euler, υπάρχει όμως μονοπάτι Euler από την κορυφή u στην κορυφή v — φροντίστε να είναι $u < v$.
- "`IMPOSSIBLE`", αν δεν υπάρχει ούτε κύκλος ούτε μονοπάτι Euler.

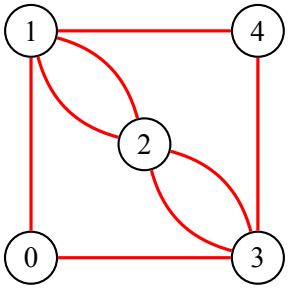
Η είσοδος θα περιέχει τα εξής. Η πρώτη γραμμή θα περιέχει δύο αριθμούς, χωρισμένους μεταξύ τους με ένα κενό διάστημα: το πλήθος των κορυφών N και το πλήθος των ακμών M. Οι επόμενες M γραμμές της εισόδου θα περιγράφουν τις ακμές του γράφου. Κάθε μία θα περιέχει δύο αριθμούς, χωρισμένους μεταξύ τους με ένα κενό διάστημα: η γραμμή που περιέχει τους αριθμούς u και v θα παριστάνει μία ακμή μεταξύ των κορυφών u και v του γράφου.

Μπορείτε να προσαρμόσετε κατάλληλα τους γνωστούς τρόπους αναπαράστασης γράφων ώστε να καλύπτουν την περίπτωση γράφων με πολλαπλές και με κυκλικές ακμές.

Παραδείγματα γράφων:







Είσοδος:

```
4 7
0 1
0 1
0 2
0 2
3 0
3 1
3 2
```

```
8 13
0 7
0 7
0 1
0 6
1 7
7 6
6 1
1 2
6 5
2 5
2 3
3 4
5 4
```

```
5 8
0 1
1 2
0 3
1 4
1 2
2 3
3 2
4 3
```

Έξοδος:

```
IMPOSSIBLE
```

```
PATH 2 5
```

```
CYCLE
```

Άσκηση 11 Κύκλος σε κατευθυνόμενο γράφο

Προθεσμία υποβολής στον grader: 14/5/2021

Υλοποιήστε την κλάση `Graph` για την αναπαράσταση κατευθυνόμενων γράφων. Χάριν ευκολίας, θα θεωρήσουμε ότι ένας γράφος περιέχει V κορυφές αριθμημένες από 0 έως $V-1$. Η κλάση σας πρέπει να υποστηρίζει τις εξής λειτουργίες, τις οποίες πρέπει να υλοποιήσετε:

```
class Graph {
 public:
  Graph(int V);
  ~Graph();
  void addEdge(int u, int v);
};
```

Θεωρήστε ότι είναι πιθανό να κληθεί η `addEdge` για την προσθήκη περισσότερων ακμών με τα ίδια άκρα. Με εξαίρεση την πρώτη, τις υπόλοιπες μπορείτε να τις αγνοείτε. Θεωρήστε επίσης ότι δε θα υπάρχουν κυκλικές ακμές – δηλαδή ακμές της μορφής (u, u).

Στη συνέχεια, προσθέστε την παρακάτω μέθοδο στην κλάση σας:

```
  bool cycle(vector<int> &path) const;
```

Η μέθοδος αυτή πρέπει να επιστρέφει **`true`** αν υπάρχει κύκλος στο γράφο, διαφορετικά **`false`**. Στην πρώτη περίπτωση, η μέθοδος πρέπει να προσθέτει στο `path` τους κόμβους που απαρτίζουν τον κύκλο. Δηλαδή, αν στο γράφο υπάρχουν οι ακμές $1 \to 4$, $4 \to 2$, $2 \to 7$ και $7 \to 1$, τότε η μέθοδός σας μπορεί να προσθέσει στο `path` τις τιμές $1, 4, 2, 7$ με αυτή τη σειρά. Εξίσου καλά μπορεί να προσθέσει τις τιμές $2, 7, 1, 4$, ή οποιονδήποτε άλλον έγκυρο κύκλο βρει.

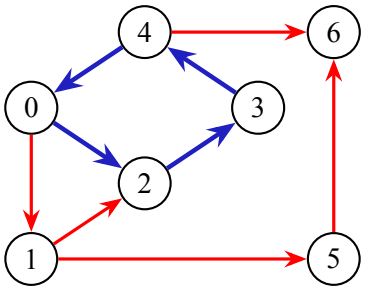
Στο αρχείο που θα ανεβάσετε στον grader θα πρέπει να συμπεριλάβετε (μόνο) τη δήλωση της κλάσης `Graph` και τις υλοποιήσεις των μεθόδων της.

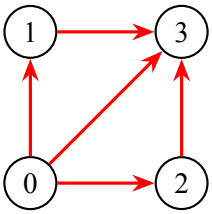
Μπορείτε να δοκιμάσετε την υλοποίησή σας με προγράμματα όπως το εξής (όπως ήδη γνωρίζετε από προηγούμενες ασκήσεις, μπορείτε να προσθέσετε τη συνάρτηση `main` ανάμεσα σε "`#ifndef CONTEST`" και "`#endif`", αν θέλετε να υποβάλλετε τον κώδικά σας στον grader χωρίς να τη σβήνετε):

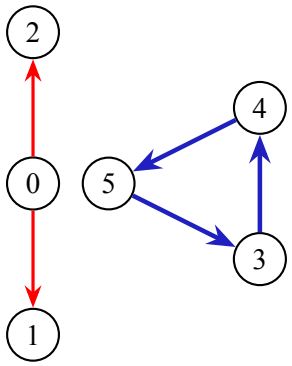
```
int main() {
  int V, E;
  cin >> V >> E;
  Graph g(V);
  for (int i = 0; i < E; ++i) { int u, v; cin >> u >> v; g.addEdge(u, v); }
  vector<int> path;
  bool c = g.cycle(path);
  if (c) {
    cout << "CYCLE: ";
    for (int i = 0; i < path.size(); ++i)
      cout << path[i] << (i == path.size()-1 ? "\n" : " ");
  } else {
    cout << "NO CYCLE" << endl;
  }
}
```

Η εκτέλεση του παραπάνω προγράμματος θα πρέπει να έχει την εξής συμπεριφορά, ανάλογα με την είσοδο που θα του δίνεται και που θα περιγράφει το γράφο:

Παραδείγματα γράφων:







Είσοδος:

```
7 9
0 1
0 2
1 2
2 3
3 4
4 0
4 6
1 5
5 6
```

```
4 5
0 1
0 2
1 3
2 3
0 3
```

```
6 5
0 1
0 2
3 4
4 5
5 3
```

Έξοδος:

```
CYCLE: 0 2 3 4
```

```
NO CYCLE
```

```
CYCLE: 3 4 5
```

Προσοχή! Θα προστεθεί μία ακόμη άσκηση, πάνω σε γράφους, με προθεσμία παράδοσης 21/5.