Контролно по СДП-практикум на 7-ма група
18.12.2015 г.

Задача 1 (25 точки)

Даден е неориентиран граф с **n** върха, в който всеки връх е номериран с число от **0** до **n**. Да се реализира **C++** клас представящ графа като свързан списък от въховете му, подредени според номерата си, като всеки връх съдържа число представляващо номерът му и стек съдържащ индексите на неговите съседи.
За графа да се реализират следните методи:

- **addEdge(u, v)** - добавя към графа ребро свързващо върховете **u** и **v**.
- **removeEdge(u, v)** - премахва реброто свързващо върховете **u** и **v**.
- **areNeighbours(u, v)** - проверява дали върховете **u** и **v** са съседни.

Да се напише приятелска функция **isConnected(G)**, която проверява дали графа **G** е свързан.

Задача 2 (25 точки)

Клас Tree представя двоично дърво с методи: `**addNode(Tree::Node*, Tree::Node*)**` - добавя възел дете на първият параметър, `**removeNode(Tree::Node*)**` - трие възел от дървото и `**orderTree(char mode)**` - пренарежда стойностите на дървото в ред, описан от параметъра *mode* по следният начин: ако *mode* е равно на 1 го нарежда в инфиксен ред, ако е 2 го нарежда в префиксен ред и ако е 4 го нарежда в постфиксен ред *[приемаме, че за елементите във възлите има дефиниран оператор <].* Дървото също има методи `**maxDepth()**` и `**hasPathSum(Type sum)**` *[тук приемаме, че типът Type има дефиниран оператор за събиране]* – който връщат съответно най-голямата дълбочина в дървото и дали дървото има път от корена до листо със сума *sum*.

Направете имплементация на дървото използвайки C++.

<u>За всички класове да се осигури правилното функциониране на методите от “голямата четворка”.</u>