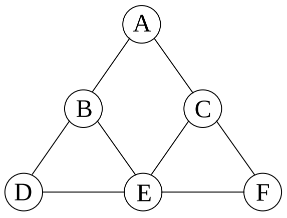
Algorithms and Programming
09 September 2016
Part II: Program (18 point version)

No books or notes are allowed. Examination time: 100 minutes. Final program due by midnight of Wednesday the 14th; use the course portal page ("Elaborati" section) to up-load it.

In graph theory, an *independent set* or *stable set* is a set of vertices in a graph, no two of which are adjacent. That is, it is a set S of vertices such that for every two vertices in S, there is no edge connecting the two. An independent set that is not the subset of another independent set is called a *maximum independent set.*
For example, given the following graph G



$\{A, D, E, F\}$ is not an independent set, $\{D, F\}$ is an independent set but not maximum, $\{A, D, F\}$ is a maximum independent set.

A file specifies an undirected graph $G = (V, E)$ by storing the list of its edges E with the format:

```
IDvertex1   IDvertex2
```

where (`IDvertex1`, `IDvertex2`) $\in E$, `IDvertex1` $\in V$, and `IDvertex2` $\in V$. Each `IDvertex` is a string of at most 20 characters. The number of edges is unknown as their order within the file. Moreover, no edge is duplicated in the file.
For example, the previous graph may be stored as:

```
F E
E D
A B
A C
B D
B E
C E
C F
```

Write a C program which is able to:

- Receive three file names on the command line. The first two files are input files, whereas the third one is an output file.
- Read the graph from the first file and store it in a proper data structure.
- Read the second file, which contains a set of edges (such as $\{A, D, E, F\}$), each one written on a separate line of the file), and verify that this set is an independent set for the previous graph.
- Find a maximum independent set and store it in the third file with the same format used to store the original graph in the first file. The program has to print-out on standard output the number of edges belonging to this maximum independent set.

Algorithms and Programming

09 September 2016

Part II: Program (12 point version)

No books or notes are allowed. Examination time: 100 minutes. Final program due by midnight of Wednesday the 14th; use the course portal page ("Elaborati" section) to up-load it.

1 (2.0 points)
Write function:

```
int subMatMax (int **mat, int r, int c, int n);
```

which receives a matrix of integer values `mat` containing `r` rows and `c` columns, and it searches the matrix to find the square sub-matrix of size `n` with the largest sum of all elements, and it returns this value.
For example, given the following matrix:

$$\begin{array}{c|c|c|c|c|}
 & 0 & 1 & 2 & 3 \\ \hline
0 & 5 & 2 & 3 & 1 \\ \hline
1 & 3 & 1 & 6 & 4 \\ \hline
2 & 3 & 0 & 5 & 2 \\ \hline
\end{array}$$

with `r=3` and `c=4`, if `n=2` the sub-matrix with the largest sum of its elements is the one highlighted (and the sum is equal to 17).

2 (4.0 points)
Write function:

```
int distance (node_t *root, int key1, int key2);
```

that receives a BST root referenced by `root`, storing integer keys, and two integer values `key1` and `key2`, and it returns the number of edges that it is necessary to traverse to reach the node of key `key1` starting from the one of key `key2` (or vice-versa).
Suggestion: function `distance` proceeds recursively along the BST as long as the path to reach keys `key1` and `key2` is the same. Then it generates two separate visits, one for key `key1` and the other for key `key2`, each one returning an integer value.

3 (6.0 points)
For his girlfriend's birthday a boy decided to buy a present. He knows she loves books so he goes to the local bookshop, where there are `n` books on sale from one of `m` genres. In the bookshop, he decides to buy `k` books (with `k` $\leq$ `m`) of different genres. Based on the genre of books on sale in the shop, write function

```
int birthday (int *vet, int n, int m, int k);
```

able to print-out all possible ways in which the boy can choose `k` books within a set of `n` books of `m` different genres. `vet[i]` equals the genre of the i-th book. Options are considered different if they differ in at least one book.
Example
If `vet = (2 1 1 4 3)`, `n=5`, `m=4`, and `k=3`, all possible book selections (numbering books starting from 0) are the following: $(0,1,3)$, $(0,1,4)$, $(0,2,3)$, $(0,2,4)$, $(0,3,4)$, $(1,3,4)$, $(2,3,4)$. Notice that selection $(0,1,3)$ means that the books in position 0 (genre 2), 1 (genre 1), and 3 (genre 4) have been selected by the boy. At the same time, selecting the book $(0,1,2)$ would be wrong because book 1 and 2 have the same genre (1).
Example
If `vet = (1 2 3 1 2 3)`, `n=6`, `m=3`, and `k=2`, all possible book selections (numbering books starting from 0) are the following: $(0,1)$, $(0,2)$, $(0,4)$, $(0,5)$, $(1,2)$, $(1,3)$, $(1,5)$, $(2,3)$, $(2,4)$, $(3,4)$, $(3,5)$, $(4,5)$.
Suggestion: Generate all possible set of k books selecting them from n books, and for each of those set check the acceptance condition (different genres).