Homework Assignment 4

Given 04/17/2019, due 05/15/2019

Write code that finds a maximum flow in a directed graph, using the Ford-Fulkerson algorithm. The graph is given as adjacency matrix with `cap`[`i`][`j`] being the capacity of the directed edge from `i` to `j`. The function received another argument, the matrix `flow`[][], which is used to return the flow values in the maximum flow from `s` to `t`

```
void maximum_flow(int n, int s, int t, int *cap, int *flow)
```

Your function has the following arguments:

- `n`: the number of vertices of the graph,
- `s`: the start vertex,
- `t`: the target vertex
- `cap`: the matrix of edge capacities.
- `flow`: the matrix used to return the maximum flow.

The vertices are numbered from 0 to `n-1`, so `s` and `t` are numbers in that range.

`capacity, flow` are a pointers to `n` $\times$ `n` matrices of nonnegative integers; in standard C the size of a matrix cannot be a variable, so we use pointer arithmetic, and treat the matrix as a one-dimensional matrix. The array element `cap[i][j]` can be accessed as `*(cap + i*n + j)`. Your function should return in the matrix `flow` the flow values of the maximum flow from `s` to `t`. The `flow` variable of your function points to space allocated for the `flow` matrix.

Your function will need at least the following auxiliary arrays:

- an `n` $\times$ `n` matrix to hold the current flow,
- an `n` $\times$ `n` matrix to hold the current residual capacities,
- an array to maintain which vertices are already visited in the search of an augmenting path from `s` to `t` with positive residual capacity.

You have to allocate and deallocate the auxiliary arrays in your function. Do not use global variables. You can use either BFS or DFS for the search of the augmenting path.