# Basic Algorithms, Assignment 5

Due, Thursday, Oct 18

1. Page 107. Problem 4.
   Solution: Suppose the data is given as arrays $\texttt{SAME}[v]$, $\texttt{DIFFERENT}[v]$ giving those $v$ definitely the same and definitely different from $v$ respectively. Lets use colors $\texttt{white}, \texttt{grey}, \textit{black}$ as usual for not having seen, in process, and completed respectively. We add an array $\texttt{TYPE}[v]$ with values $A, B$. The outside of the shell will be that for each vertex $v$ if $\texttt{COLOR}[v] = \texttt{white}$ then $\texttt{TYPE}[v] \leftarrow A$ and $\texttt{DFS}[v]$. (We can arbitrarily label a first butterfly as species A.) In doing $\texttt{DFS}[v]$ we go through $\texttt{SAME}[v]$ and give all those $w$ the same type as $v$ and go through $\texttt{DIFFERENT}[v]$ and give all those $w$ the other type than $v$. In both cases, however, when the found $w$ is not white (so it has already been given a type), if its already given type is different from what we want to give it then we return that the judgements are inconsistent. If this never happens the judgements will be consistent and we will have given a consistent typing.

2. Modify the topological ordering algorithm of §3.6, or the version given in class, so that there will be a stacks (or queues, you decide) BADCYCLE and GOODORDER and a Boolean Variable NOPROBLEMO. The input to your algorithm will be a directed graph with the adjacency list representation. When the algorithm is over there are two cases. If there is a topological order then NOPROBLEMO = **true** and GOODORDER gives such an order. If there is no topological order then NOPROBLEMO = **false** and BADCYCLE gives a directed cycle. (When NOPROBLEMO = **true** it doesn't matter what BADCYCLE is and when NOPROBLEMO = **false** it doesn't matter what GOODORDER is.)

   Give your algorithm in detailed pseudocode with copious commentary Solution (Outline): Keep a CALLSTACK of which $\texttt{DFS}[v]$ are in process. This starts NIL and when $\texttt{DFS}[v]$ is called ($v$ turns grey) $v$ is added to it and when $\texttt{DFS}[v]$ is completed ($v$ turns black) $v$ is removed from it. So in the middle if it has (say) $u, z, w, v$ it means $\texttt{DFS}[v]$ has called $\texttt{DFS}[w]$ which has called $\texttt{DFS}[z]$ which has called $\texttt{DFS}[u]$. These are the current grey points. Now say $w \in \mathbf{Adj}[z]$, as $\texttt{COLOR}[w] = \texttt{grey}$ we now want to spit out a cycle. The cycle will consist of everything in the stack *until* you reach $w$, in this case $u, z, w$. Save $w$ as some special symbol BINGO and go until you reach it. However, the arrows are in reverse order, $u \in \mathbf{Adj}[z], z \in \mathbf{Adj}[w], w \in \mathbf{Adj}[u]$. One way to get this is to put the elements of `CALLSTACK` into a *stack* `BADCYCLE` (originally `NIL`) until reaching `BINGO`. That re-reverses the order, as you want.

Why did you come to Casablance anyway, Rick?
    I came for the waters.
Waters, what waters? Casablanca is in the desert.
    I was misinformed
Claude Rains and Humphrey Bogart in *Casablanca*