**Problemas no uso do GOTO**

Não é útil para tornar o código mais fácil de ler pois há outras opções mais legíveis ou pelo menos tão legíveis quanto.

Pode ser útil para reduzir o tamanho do código, mas código expressivo e de fácil manutenção é mais importante que código curto.

Não é mais útil para evitar repetição de código do que o encapsulamento do código em loops formais, em métodos e em objetos.

Não é mais útil para sair de loops aninhados do que declarar uma variável de controle e testar seu estado na declaração de cada loop.

Não é mais útil para implementar máquina de estado do que a orientação a objetos.

Veja, **programadores que não nasceram experientes em algum momento vão produzir código ruim com qualquer recurso da ferramenta**.

Estes programadores podem sim se orientar por boas práticas bem argumentadas para poder desde cedo produzir bom código e ir compreendendo melhor os porquês e questionando estes porquês ao longo da carreira.

Portanto, a fim de ajudar estes profissionais, algumas vezes chamamos de "má prática" as práticas que historicamente mostraram trazer mais malefícios que benefícios. Por exemplo: disponibilizar globais para alteração de estado, capturar exceções sem saber por que, micro-otimizar código sem evidências da necessidade, dar nomes pouco significativos aos artefatos, usar herança como primeira opção...

Por fim, se o GOTO é um destes elementos com maior potencial de causar prejuízo e se existem sempre ótimas opções que o tornam dispensável, então usar GOTO pode sim ser considerado uma má prática.