```
savemarked (generic function with 1 method)
```

```
begin
    include("../shared/savemarked.jl")
    #savemarked()
end
```

graphcol_3

- graph loader/converter for dimacs `col` format: `loadcol`
    - the `tolg` parameter for converting the data into the default `lg` format
- the data is from Michael Trick's page
- dimacs is an old/simple format described here

```
begin

md"""
#### graphcol_3
* graph loader/converter for dimacs `col` format: `loadcol`
  * the `tolg` parameter for converting the data into the default `lg` format
* the data is from [Michael Trick's page](https://mat.gsia.cmu.edu/COLOR/instances.html)
* dimacs is an old/simple format described [here](https://mat.gsia.cmu.edu/COLOR/general/ccformat.ps)"""

end
```

```
begin
    import Pkg
    Pkg.activate(".")


    Pkg.add.(
        [
            "Graphs",
            "StatsBase",
        ]
    )
    Pkg.instantiate()

    using
        Graphs,
        StatsBase
end
```

```
  Activating project at `~/Asztal/git/plnotebooks/graphcol_3`
   Resolving package versions...
  No Changes to `~/Asztal/git/plnotebooks/graphcol_3/Project.toml`
  No Changes to `~/Asztal/git/plnotebooks/graphcol_3/Manifest.toml`
   Resolving package versions...
  No Changes to `~/Asztal/git/plnotebooks/graphcol_3/Project.toml`
  No Changes to `~/Asztal/git/plnotebooks/graphcol_3/Manifest.toml`
```

```
loadcol (generic function with 1 method)
```

```
#--->loadcol

# .col extension is a must
function loadcol(gfile::String; tolg=false, toopt=false)
    _e(msg)=error("loadcol: $(msg)")

    !isfile(gfile) && _e("no such file")
    sfile=split(gfile,'.')
    (sfile[end]!="col") && _e("wrong extension")
    gstring=split(read(gfile,String),'\n',keepempty=false)

    num_colors=-1
    E=[]
    nV,nE,tV=-1,-1,-1
    for line in gstring
        sline=split(line,keepempty=false)
        (sline[1]=="c") && continue

        if sline[1]=="p" # only the last counts, but must precede the first 'p'
        line, bcos it  is used in a sanity check
            nV,nE=parse.(Int,sline[3:end])
            continue
        end
        if sline[1]=="e"
            a,b=parse.(Int,split(line)[2:end])
            if a<1 || a>nV || b<1 || b>nV
                _e("vertex is out of range")
            end
            push!(E,(a,b))
            continue
        end
        if sline[1]=="num_colors"
            num_colors=parse(Int,sline[2])
            continue
        end
    end
    if nV<0 || nE<0 || length(E)!=nE
        _e("wrong data")
    end
    G=Graph()
    add_vertices!(G,nV)
    for (a,b) in E
        add_edge!(G,a,b)
    end
    if tolg==true
        sfile=join(sfile[1:end-1],'.')
        savegraph("$(sfile).lg",G)
        printstyled(stderr,"saved $(sfile).lg\n"; color=:green)
    end
    if num_colors>0 && toopt==true
        open("$(sfile).opt","w") do f
            println(f,num_colors)
        end
        printstyled(stderr,"saved $(sfile).opt\n"; color=:yellow)
    end
    G
end
```

```
57
58 #--->loadcol
```

```
1 #### convert the col's to lg's
2 for f in readdir("../data/col-instances/"; join=true)
3     sf=split(f,'.')
4     if sf[end]=="col"
5         jf=join(sf[1:end-1],'.')
6         isfile("$(jf).lg") && isfile("$(jf).opt") && continue
7         loadcol(f; tolg=true, toopt=true)
8     end
9 end
```

```
1 begin
2
3 # test
4 include("../shared/graphcol_bt.jl")
5 G=loadgraph("../data/col-instances/queen5_5.lg")
6 opt=parse(Int,read("../data/col-instances/queen5_5.opt",String))
7 @time graphcol_bt(G,opt-1)|>println
8 @time graphcol_bt(G,opt)|>println
9 @time greedy_color(G,reps=33)|>println
10
11 end
```

```
(num_colors = -1, colors = nothing)
  0.632118 seconds (450.44 k allocations: 31.074 MiB, 99.95% compilation ti
me)
(num_colors = 5, colors = [1, 2, 3, 4, 5, 3, 4, 5, 1, 2  …  2, 3, 4, 5, 1,
4, 5, 1, 2, 3])
  0.240531 seconds (214.69 k allocations: 14.645 MiB, 7.67% gc time, 99.70%
compilation time)
Graphs.Coloring{Int64}(5, [2, 5, 1, 3, 4, 3, 4, 2, 5, 1  …  4, 2, 5, 1, 3,
1, 3, 4, 2, 5])
  0.277017 seconds (326.44 k allocations: 21.705 MiB, 99.61% compilation ti
me)
```