Pensieve header: The package “Perm”.

Topics (in no particular order). Whatever you may suggest; whatever comes to my mind; ~~the Fibonacci numbers~~; ~~the Catalan numbers~~; ~~the Jones polynomial~~; ~~a more efficient Jones algorithm~~; ~~a riddle on spheres~~; Khovanov homology; Γ-calculus; the Hopf fibration; Hilbert’s 13th problem; non-commutative Gaussian elimination; free Lie algebras; the Baker-Campbell-Hausdorff formula; wacky numbers; ~~an order 4 torus~~; the Schwarz Lantern; knot colourings; the Temperley-Lieb pairing; the dodecahedral link; ~~sound experiments~~; barycentric subdivisions; ~~some Peano curves~~; braid closures and Vogel’s algorithm; ~~the insolubility of the quintic~~; phase portraits; the Mandelbrot set; shadows of the Cantor aerogel; quilt plots; some image transformations; De Bruijn graphs; the Riemann series theorem; finite type invariants and the Willerton fish; ~~the Towers of Hanoi~~; ~~Hochschild homology of (some) coalgebras~~; convolutions and image improvements; the 8-5-3 milk jug problem; ~~a cow problem,~~ a permutations package.

```
gcd[0, n_] := n;
gcd[n_, m_] /; m < n := gcd[m, n];
gcd[n_, m_] /; m ≥ n := gcd[n, m - n];
```

```
gcd[42, 54]
```

6

```
? GCD
```

GCD[n_1, n_2, ...] gives the greatest common divisor of the n_i. ≫

An Image Manipulation Challenge

The image at http://drorbn.net/bbs/show?shot=17-1750-171016-111042.jpg is pathetic. Can you improve it? Whatever you do, should also work well with all other images at http://drorbn.net/bbs/show.php?prefix=17-1750.

The 8-5-3 Milk Jug Problem

Challenge. Draw the state graph of this problem (no spilling allowed!).

NCGE Challenge

Update the NCGE program to contain “backtracking information”. Use it to find how to turn the lower face of a Rubik’s cube

by turning all but the lower face of that cube.

The Mathematica Package "Perm"

Challenge. Re-implement permutations, though using the standard "list of images" notation for permutations: Perm[5,2,3,1,4], etc. Your package should know $\sigma\circ\tau$, σ^{-1}, $\sigma[[i]]$, Pivot[σ], IdentityPermutation[n], it should interact well with Cycles, and its internals should be hidden.

```
σ_Perm ∘ τ_Perm /; Length[σ] == Length[τ] := Perm @@ Table[σ[[τ[[i]]]], {i, Length[σ]}];
```

```
Perm[1, 3, 4, 2] ∘ Perm[2, 3, 1, 4]
```

```
Perm[3, 4, 1, 2]
```

```
{a, b, c}[[2]]
```

```
b
```

```
{a, b, c}[[{3, 2}]]
```

```
{c, b}
```

```
σ_Perm ∘ τ_Perm /; Length[σ] == Length[τ] := σ[[List @@ τ]];
```

```
?? SmallCircle
```

```
SmallCircle[x, y, ...] displays as x ∘ y ∘ .... >>
```

```
σ_Perm ∘ τ_Perm /; Length[σ] == Length[τ] := σ[[List @@ τ]]
```

```
Perm[1, 3, 4, 2] ∘ Perm[2, 3, 1, 4]
```

```
Perm[3, 4, 1, 2]
```

```
? Position
```

```
Position[expr, pattern] gives a list of the positions at which objects matching pattern appear in expr.
Position[expr, pattern, levelspec] finds only objects that appear on levels specified by levelspec.
Position[expr, pattern, levelspec, n] gives the positions of the first n objects found.
Position[pattern] represents an operator form of Position that can be applied to an expression. >>
```

```
σ = Perm[3, 4, 1, 2]
```

```
Perm[3, 4, 1, 2]
```

```
Position[σ, 1]
```

```
{{3}}
```

```
Expand[(a + b)^2]
```

$$a^2 + 2\,a\,b + b^2$$

```
Position[Expand[(a + b)^2], b]
```

```
{{2, 3}, {3, 1}}
```

```
(σ_Perm)^-1 := Table[Position[σ, i][[1, 1]], {i, Length[σ]}]
```

SetDelayed: Tag Power in $\frac{1}{\sigma\_\text{Perm}}$ is Protected.

```
Perm /: (σ_Perm)^-1 /; PermutationQ[σ] := Perm @@ Table[Position[σ, i][[1, 1]], {i, Length[σ]}]
```

Can you make this more efficient?

```
Perm /: (σ_Perm)^-1 /; PermutationQ[σ] := (
  τ = σ;
  Do[τ[[σ[[i]]]] = i, {i, Length[σ]}];
  τ
 )
```

```
Perm[2, 3, 1]^-1
```

```
Perm[3, 1, 2]
```

```
σ
```

```
Perm[3, 4, 1, 2]
```

```
Perm[2, 3, 1]^-1
```

```
Perm[3, 1, 2]
```

```
σ^-1
```

```
Perm[3, 4, 1, 2]
```

```
PermutationQ[σ_Perm] := Sort[List @@ σ] === Range[Length[σ]]
```

```
PermutationQ[Perm[2, 3, 1]]
```

```
True
```

```
PermutationQ[Perm[1, 2, 2]]
```

```
False
```

```
? Context
```

```
Context[] gives the current context.
Context[symbol] gives the context in which a symbol appears.  ≫
```

```
Context[]
```

```
Global`
```

```
cow = 7
```

```
7
```

```
Context[cow]
```

```
Global`
```

```
Context[Plus]
```

```
System`
```

```
? Contexts
```

```
Contexts[] gives a list of all contexts.
Contexts["string"] gives a list of the contexts that match the string.  ≫
```

```
Contexts[] // Short
```

```
{Algebra`, ≪822≫, $CellContext`}
```

? $ContextPath

```
$ContextPath is a global variable that gives a list of contexts
   to search, before $Context, in trying to find a symbol that has been entered.  >>
```

$ContextPath

```
{DocumentationSearch`, ResourceLocator`, WolframAlphaClient`,
 DrorBarNatan`, CloudObjectLoader`, InterpreterLoader`, IntegratedServicesLoader`,
 IconizeLoader`, HTTPHandlingLoader`, GeneralUtilitiesLoader`, AuthenticationLoader`,
 SystemTools`, StreamingLoader`, SVTools`, PacletManager`, System`, Global`}
```

? $Context

```
$Context is a global variable that gives the current context.  >>
```

$Context

```
Global`
```

$Context = "Horse`"

```
Horse`
```

cow

```
7
```

sheep = 8

```
8
```

Context[sheep]

```
Horse`
```

Context[cow]

```
Global`
```

? BeginPackage

```
BeginPackage["context`"] makes context` and System` the only active contexts.
BeginPackage["context`", {"need1`", "need2`", ...}] calls Needs on the needi.  >>
```

Perm::usage = "Perm[5,2,3,1,4] means the permutation that maps 1→5, 2→2, 3→3, 4→1, 5→4."

```
Perm[5,2,3,1,4] means the permutation that maps 1→5, 2→2, 3→3, 4→1, 5→4.
```

? Perm

```
Perm[5,2,3,1,4] means the permutation that maps 1→5, 2→2, 3→3, 4→1, 5→4.
```

? Begin

```
Begin["context`"] resets the current context.  >>
```

? End

```
End[] returns the present context, and reverts to the previous one.  >>
```

? EndPackage

```
EndPackage[] restores $Context and $ContextPath to their values before the
   preceding BeginPackage, and prepends the current context to the list $ContextPath.  >>
```