Code Generation

The CodeGeneration package offers new support for translating Maple code to the Julia programming language.

Julia

With CodeGeneration[Julia], you can translate expressions to code fragments:

```
> with(CodeGeneration):
```

```
> Julia(sqrt(a^2+b^2+c^2))
```

```
cg = sqrt(a ^ 2 + b ^ 2 + c ^ 2)
```

You can also translate procedures and larger programs.

```
> Julia(m->add(i, i=1..m))
```

```
function cg0(m)
    r = 0
    for i = 1:m
        r = r + i
    return(r)
end
```

CodeGeneration[Julia] translates many Maple data structures and functions, including many routines for linear algebra and special functions, to equivalents in Julia.

```
> Julia(Matrix([[1,2],[3,4]]))
```

```
cg1 = [1 2; 3 4]
```

```
> Julia(piecewise(v<1, -v^3+3*v+1, v<2, 2*v^3-9*v^2+12*v-2, -v^3+9*v^2-24*v+22))
```

```
cg2 = -v ^ 3 + 3 * v + 1 ? v < 1 : 2 * v ^ 3 - 9 * v ^ 2 + 12
```

```
* v - 2 ? v < 2 : -v ^ 3 + 9 * v ^ 2 - 24 * v + 22
```

> $Julia((M, n) \rightarrow M - x\, LinearAlgebra{:}\text{-}IdentityMatrix(n))$

```
cg3(M,n) = M - x * eye(n)
```

> $Julia(\_C1\ \mathrm{BesselJ}(\nu, x) + \_C2\ \mathrm{BesselY}(\nu, x))$

```
cg4 = _C1 * besselj(nu, x) + _C2 * bessely(nu, x)
```

Code Generation for Julia in Maple can also translate some key commands from Statistics:

> $Julia('Statistics{:}\text{-}Mean([5, 2, 1, 4, 3])')$

```
cg5 = mean([5,2,1,4,3])
```

> $Julia('Statistics{:}\text{-}Median([5, 2, 1, 4, 3])')$

```
cg6 = median([5,2,1,4,3])
```

> $Julia('Statistics{:}\text{-}StandardDeviation([5, 2, 1, 4, 3])')$

```
cg7 = std([5,2,1,4,3])
```

When possible, Maple attempts to return an equivalent command to commands dealing with many distributions. In this example, the evaluated probability density function is translated.

> $Julia(Statistics{:}\text{-}PDF(LogNormal(0, 1), x))$

```
cg9 = 0 ? x < 0 : 1 / x * sqrt(2) * pi ^ (-1//2) * exp(-log
(x) ^ 2 / 2) / 2
```