Code utility

Code

Sintaxis

Sintaxis Code take the script in the description or in a string and try to convert it into a SMath function

```
test(a,b∘) // fn name

x:a+1  // use ∘ for local
y:b∘-1 // varialbes

// use do for mask
// multiple lines
do u:1 , v∘:2

while cond1
  for k,rng
    r:19
    s:20
    try a
      b
  if cond2 // if(#2)
    t:21
  if cond3 // if(#3)
    if qq  // nested if
      t:22
  else
    u:22
  if cond4 // if(#4)
    t:23
  elif cond4'
    u:23
  else
    v∘:23

f(u):u*cos(u) // one line fnc

let g(u):     // multiple
  if u>0      // lines fnc
    u^2
  else
    3*u-1

m:mat( ... // continue
 x,y, ...  // in
 r,s, ...  // next line
 2,2)

Code >> p // Another def
if cond
  a∘
else
  b
```

$test := test$

$C := Code(test) =$ "line(test(a,b.∘test∘):line(x:a+1,y:b.∘test∘-1,

=>

$test(a, b_{\circ test\circ}) :=$
- $x := a + 1$
- $y := b_{\circ test\circ} - 1$
- $do_{\circ test\circ}(u := 1, v_{\circ test\circ} := 2)$
- while *cond1*
  - for $k \in rng$
    - $r := 19$
    - $s := 20$
    - try
      - a
    - on error
      - b
  - if *cond2*
    - $t := 21$
  - if *cond3*
    - if *qq*
      - $t := 22$
  - else
    - $u := 22$
  - if *cond4*
    - $t := 23$
  - else if *cond4'*
    - $u := 23$
  - else
    - $v_{\circ test\circ} := 23$
- $f(u) := u \cdot \cos(u)$
- $g(u) :=$
  - if $u > 0$
    - u^2
  - else
    - $3 \cdot u - 1$
- $m := \begin{bmatrix} x & y \\ r & s \end{bmatrix}$

$p :=$
- if *cond*
  - $a_{\circ p\circ}$
- else
  - b

2

Pasting the string without quotes

Use ∘ (unicode char 2092h) for get "local" variables as the variable name with a subscript with the function name.

Further Examples

```
Code >> IsPrime(n) // basic
sq:trunc(sqrt(n))  // prime test
for k:2,k≤sq,k:k+1
  if mod(n,k)≡0
    break
k ≡ (1+sq)*(n≥2)

Code >> IsMersennePrime(p)
do m:2^p-1 , s:4
for k:2,k≤p-1,k:k+1 // Lehmer
  s:s^2-2
  s:s-m*trunc(s/m)
s ≡ 0

Code >> IsCircularPrime(p)
do n:trunc(log10(p)) , m:10^n , q:p
for k:0,k≤n,k:k+1
  if q<p | ¬IsPrime(q)
    break
  else
    q:10*mod(q,m)+trunc(q/m)
k ≡ n+1

Code >> Mobius(n)  // Mobius number
p:0
for k,range(1,n+1)
  if mod(n,k)≡0 & IsPrime(k)
    if mod(n,k^2)≡0
      break
    else
      p:p+1
(-1)^mod(p,2)*(k≡n+1)

Code >> Catalan(n) // Catalan number
C:mat(1,1,1)       // from 1 to n
for k:1,k≤n,k:k+1
  el(C,k+1):2*(2*k-1)/(k+1)*el(C,k)
C

Code >> Find(M)       // index
k:range(1,length(M)) // where M=1
col(findrows(eval( ...
  augment(el(M,k),k)),1,1),2)
```

$NumTh := NumTh$

$\text{str2num}(Code(NumTh)) = 6$

$N := [1..50]$ $\quad P := N$ $\quad Find(\overrightarrow{IsPrime(N)})$

$N := [1..400]$ $\quad C := N$ $\quad Find(\overrightarrow{IsCircularPrime(N)})$

$N := [1..20]$ $\quad M := N$ $\quad Find(\overrightarrow{IsMersennePrime(N)})$

$$P = \begin{bmatrix} 2 \\ 3 \\ 5 \\ 7 \\ 11 \\ 13 \\ 17 \\ 19 \\ 23 \\ 29 \\ 31 \\ 37 \\ 41 \\ 43 \\ 47 \end{bmatrix} \quad C = \begin{bmatrix} 2 \\ 3 \\ 5 \\ 7 \\ 11 \\ 13 \\ 17 \\ 37 \\ 79 \\ 113 \\ 197 \\ 199 \\ 337 \end{bmatrix} \quad M = \begin{bmatrix} 3 \\ 5 \\ 7 \\ 13 \\ 17 \\ 19 \end{bmatrix}$$

$$Catalan(15) = \begin{bmatrix} 1 \\ 1 \\ 2 \\ 5 \\ 14 \\ 42 \\ 132 \\ 429 \\ 1430 \\ 4862 \\ 16796 \\ 58786 \\ 208012 \\ 742900 \\ 2674440 \\ 9694845 \end{bmatrix}$$

$\mu := \text{matrix}(5, 20)$ $\quad N := [1..100]$ $\quad \mu_N := Mobius(N)$

$$\mu = \begin{bmatrix} 1 & -1 & -1 & 0 & -1 & 1 & -1 & 0 & 0 & 1 & -1 & 0 & -1 & 1 & 1 & 0 & -1 & 0 & -1 & 0 \\ 1 & 1 & -1 & 0 & 0 & 1 & 0 & 0 & -1 & -1 & -1 & 0 & 1 & 1 & 1 & 0 & -1 & 1 & 1 & 0 \\ -1 & -1 & -1 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & 1 & 0 & 1 & 1 & -1 & 0 \\ -1 & 1 & 0 & 0 & 1 & -1 & -1 & 0 & 1 & -1 & -1 & 0 & -1 & 1 & 1 & 0 & 0 & 1 & -1 & -1 & 0 \\ 0 & 1 & -1 & 0 & 1 & 1 & 1 & 0 & -1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & -1 & 0 & 0 & 0 \end{bmatrix}$$

```
Code >> nIsZ(xₒ,Tₒ)
num2str(eval(normi(stack(xₒ))<Tₒ))≡"1"

Code >> nSort(Aₒ)
for rₒ,range(1,length(Aₒ))
  qₒ:el(Aₒ,rₒ)
  if num2str(IsDefined(qₒ))≡"1"
    el(Vₒ,rₒ):normi(stack(el(Aₒ,rₒ)))
  else
    el(Vₒ,rₒ):10^300
col(eval(csort( ...
  augment(Vₒ,range(1,length(Aₒ))),1)),2)

Code >> nRREF(Mₒ,Tₒ,Pₒ) // Row reduced echelon form
do Aₒ:Mₒ , mₒ:rows(Aₒ) , nₒ:cols(Aₒ) , Bₒ:Aₒ
do krₒ:range(1,mₒ) , Pₒ:eval(matrix(0,1))
do rₒ:1 , cₒ:1
while (rₒ≤mₒ)&(cₒ≤nₒ)
  ρₒ:el(nSort(el(Aₒ,range(rₒ,mₒ),cₒ)),mₒ-rₒ+1)+rₒ-1
  if nIsZ(el(Aₒ,ρₒ,cₒ),Tₒ)
    kρₒ:range(ρₒ,mₒ)        // fill with 0
    el(Aₒ,kρₒ,cₒ):0         // above the ρ column
    cₒ:cₒ+1
  else
    Pₒ:eval(stack(Pₒ,ρₒ))   // store ρ
    // x(P) are the bound vars in Ax=b
    // columns P of M are a basis for its range
    for kcₒ,range(1,nₒ)     // swap rows ρ & r
      tmpₒ:el(Aₒ,ρₒ,kcₒ)
      el(Aₒ,ρₒ,kcₒ):el(Aₒ,rₒ,kcₒ)
      el(Aₒ,rₒ,kcₒ):tmpₒ
    kₒ:range(cₒ,nₒ)     // divide & substract
    pₒ:el(Aₒ,rₒ,cₒ)     // pivot value
    el(Aₒ,rₒ,kₒ):el(Aₒ,rₒ,kₒ)/pₒ
    el(Bₒ,krₒ,kₒ):el(Aₒ,krₒ,kₒ)- ...
      if(krₒ≠rₒ,el(Aₒ,krₒ,cₒ)*el(Aₒ,rₒ,kₒ),0)
    el(Aₒ,krₒ,kₒ):el(Bₒ,krₒ,kₒ)  // update
    do rₒ:rₒ+1 , cₒ:cₒ+1
Aₒ

Code >> nRank(Mₒ,Tₒ)
Aₒ:nRREF(Mₒ,Tₒ,Rₒ) ¦ length(Rₒ)

Code >> nIsZ(xₒ) ¦ nIsZ(xₒ,'ZTOL)
Code >> nRREF(Mₒ) ¦ nRREF(Mₒ,'ZTOL,0)
Code >> nRank(Mₒ) ¦ nRank(Mₒ,'ZTOL)
```

$nRREF_{Code} := nRREF_{Code}$

$\text{str2num}\left(Code\left(nRREF_{Code}\right)\right) = 7$

$ZTOL := 10^{-12}$

$$A := \begin{bmatrix} 0 & 2 & 0 & 1 & 0 & 0 \\ 0 & x & 6 & 0 & 1 & 0 \\ 7 & 0 & 9 & 0 & 0 & 1 \end{bmatrix} \quad D := \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \quad B := \begin{bmatrix} 0 & 2 & 0 & 1 & 0 \end{bmatrix} \quad C := B^{T}$$

$$nRREF(A) = \begin{bmatrix} 1 & 0 & 0 & \frac{3 \cdot x}{28} & -\frac{3}{14} & \frac{1}{7} \\ 0 & 1 & 0 & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 1 & -\frac{x}{12} & \frac{1}{6} & 0 \end{bmatrix} \quad nRank(A) = 3$$

$$nRREF(B) = \begin{bmatrix} 0 & 1 & 0 & \frac{1}{2} & 0 \end{bmatrix} \quad nRREF(C) = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$nRank(B) = 1$ $\quad$ $nRank(C) = 1$

$$nRREF(D) = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 0 \end{bmatrix} \quad nRank(D) = 2$$

$$E := \begin{bmatrix} 0 & 1 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & -1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad nRREF(E) = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$\text{rank}(E) = 4$ $\quad$ $Wrong!$ $\quad$ $nRank(E) = 3$

```
Code >> nBisection(fs◦,α◦,β◦,ε◦)
// Bisection method
str2num(strrep( ...
  "f◦(x◦):@◦(x◦)","@◦",num2str(fs◦)))
do a◦:α◦ , b◦:β◦ , ε◦:UoM(b◦-a◦)*ε◦
if (sa◦:sign(f◦(a◦)))≡0
  a◦
elif (sb◦:sign(f◦(b◦)))≡0
  b◦
elif sa◦*sb◦>0
  error("nBisection_Wrong_interval.")
else
  for iter◦ , range(1, ...
  1+trunc(log(abs((b◦-a◦)/ε◦),2)))
    do  c◦:{a◦+b◦}/2 , sc◦:sign(f◦(c◦))
    if sc◦≡0
      break
    elif sa◦*sc◦>0
      do a◦:c◦ , sa◦:sc◦
    else
      do b◦:c◦ , sb◦:sc◦
  c◦

Code >> nNR.1(fs◦,x◦,εox◦,εoy◦,ho◦,MI◦)
str2num(strrep( ...
  "f◦(x◦):@◦(x◦)","@◦",num2str(fs◦)))
do a◦:x◦ , εx◦:εox◦*UoM(a◦)
do h◦:ho◦*UoM(a◦) , εy◦:εoy◦*UoM(f◦(a◦))
for iter◦,range(1,MI◦)
  y◦:f◦(a◦)
  if abs(y◦)>εy◦
    b◦:eval(a◦-(h◦*y◦)/(f◦(a◦+h◦)-y◦))
    if abs(a◦-b◦)>εx◦
      a◦:b◦
    else
      break
  else
    break
if(MI◦≥iter◦,a◦,error("nNR.1_MaxIt_Rached"))
```

$nBisection_{Code} := nBisection_{Code}$

$\text{str2num}\left(Code\left(nBisection_{Code}\right)\right) = 2$

$TOL := 10^{-7}$ $\quad h := 10^{-5}$

$f(x) := x^2 - 2$ $\quad nBisection(f, 0, 2, TOL) = 1.4142$

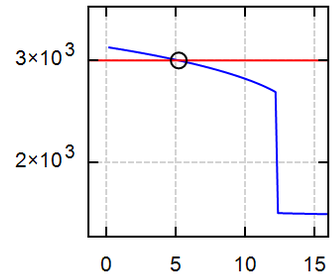
$nNR_1(f, 1, TOL, ZTOL, h, 50) = 1.4142$

$h(p) := \text{CoolProp\_Props}("H", "P", p, "T", 600\,K, "H2O")$

$ho := 3000\,\frac{kJ}{kg}$

$po := nBisection\left(\begin{vmatrix}\varphi(p) := ho - h(p), 1\,MPa, 9\,MPa, TOL\\ \varphi\end{vmatrix}\right)$

$po = 5.1974\,MPa$



$po := nNR_1\left(\begin{vmatrix}\varphi(p) := ho - h(p), 4\,MPa, TOL, TOL, h, 50\\ \varphi\end{vmatrix}\right)$

$po = 5.1974\,MPa$

```
nRKA(Ds◦,xo◦,to◦,te◦,N◦,ε◦)  // Runge-Kutta method
                              // with adaptive step
str2num(strrep( ...
  "D◦(t◦,x◦):@◦(t◦,x◦)","@◦",num2str(Ds◦)))
X◦:el(xo◦,range(1,length(xo◦)))
h◦:(te◦-to◦)/N◦
do kt◦:range(1,N◦+1) , To◦:eval(to◦+h◦*(kt◦-1))
for t◦:to◦+h◦,t◦≤te◦,t◦:t◦+h◦
  x◦:col(X◦,cols(X◦))
  k1◦:h◦/3*D◦(t◦,x◦)
  k2◦:h◦/3*D◦(t◦+h◦/3,x◦+k1◦)
  k3◦:h◦/3*D◦(t◦+h◦/3,x◦+{k1◦+k2◦}/2)
  k4◦:h◦/3*D◦(t◦+h◦/2,x◦+{3*k1◦+9*k3◦}/8)
  k5◦:h◦/3*D◦(t◦+h◦,x◦+{3*k1◦-9*k3◦}/2+6*k4◦)
  δ◦:k1◦-9/2*k3◦+4*k4◦-1/2*k5◦
  if normi(δ◦)≤5*ε◦
    X◦:augment(X◦,x◦+1/2*(k1◦+4*k4◦+k5◦))
    el(T◦,cols(X◦)):t◦
    h◦:2*h◦
  else
    t◦:t◦-h◦
    h◦:h◦/2
X◦:eval(transpose(X◦))
for k◦,range(1,cols(X◦))
  el(RK◦,kt◦,k◦):cinterp(T◦,col(X◦,k◦),el(To◦,kt◦))
eval(augment(To◦,RK◦))
```

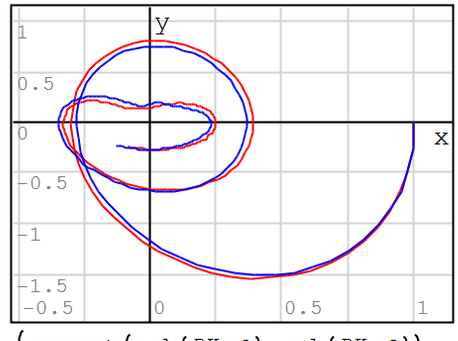
$nRKA_{Code} := nRKA_{Code}$

$\text{str2num}\left(Code\left(nRKA_{Code}\right)\right) = 1$ $\quad \text{Clear}(D) = 1$

$$D(t, x) := \begin{bmatrix} x_2 \\ -\left(5 \cdot x_1 + x_2 - \sin(t)\right) \end{bmatrix}$$

$RK := nRKA\left(D, [1\ 0], 0, 10, 200, 10^{-5}\right)$

$RK' := \text{Rkadapt}\left(\begin{bmatrix}1\\0\end{bmatrix}, 0, 10, 200, D\right)$



$\begin{cases}\text{augment}(\text{col}(RK, 2), \text{col}(RK, 3))\\ \text{augment}(\text{col}(RK', 2), \text{col}(RK', 3))\end{cases}$

new if

```
if2(x)

if x>0
  ln(x)
```

$if2 := if2$

```
if3(x)

if x>0
  ln(x)
else
  ln(-x)
```

$if3 := if3$

```
if4(x)

if x>0
  ln(x)
elif x<0
  ln(-x)
else
  -∞
```

$if4 := if4$

$\text{str2num}(Code(\text{description}(if2))) = 1$

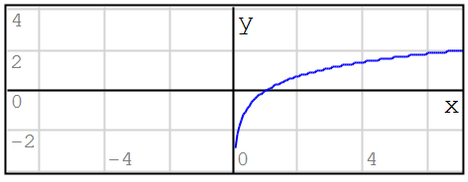
$\text{str2num}(Code(\text{description}(if3))) = 1$

$\text{str2num}(Code(\text{description}(if4))) = 1$

$if2(3) = 1.0986$

$if2(-3) = $

$if2(0) = $

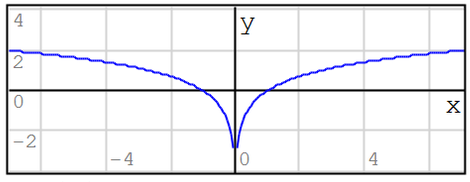


$if2(x)$

$if3(3) = 1.0986$

$if3(-3) = 1.0986$

$if3(0) = $

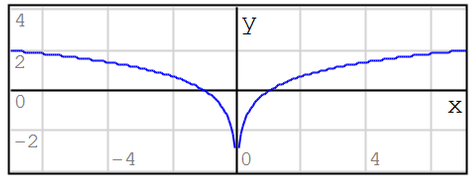


$if3(x)$

$if4(3) = 1.0986$

$if4(-3) = 1.0986$

$if4(0) = -\infty$



$if4(x)$

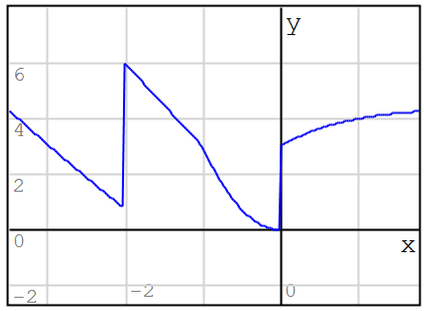
fnc

Compound functions inside the main body. Also shows it using ; as argument separator in the script, and using mathcad block.

```
Fun(x;a)
let f(u):  // declare f
  if u≥0
    a+x*atan(a;2*x)
  elif u>-2
    a*if(u>-1;u^2;-u)
  else
    -a-x*atan(a;x/2)
f(x)       // call it
```

Fun_{Code}

$\text{str2num}(Code(Fun_{Code})) = 1$



$Fun(x, 3)$

SMath code for the script:

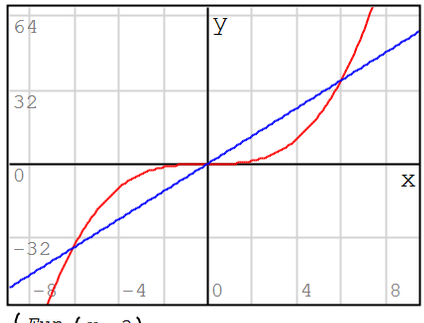
$$Fun(x,a) := \left| \begin{array}{l} f(u) := \left| \begin{array}{l} \text{if } u \geq 0 \\ \quad a + x \cdot \text{atan}(a, 2 \cdot x) \\ \text{else if } u > -2 \\ \quad a \cdot \text{if } u > -1 \\ \qquad u^2 \\ \qquad \text{else} \\ \qquad -u \\ \text{else} \\ \quad -a - x \cdot \text{atan}\left(a, \frac{x}{2}\right) \end{array} \right. \\ f(x) \end{array} \right.$$

1

```
Fun(x;a)
do s:0 ; p:1
if a≥0
  let f(u):
    for k;range(1;a)
      s:s+k*u
    s
else
  let f(u):
    for k;range(1;-a)
      p:p*u/k
    p
f(x)
```

Fun_{Code}

$\text{str2num}(Code(Fun_{Code})) = 1$



$Fun(x, 3)$

$Fun(x, -3)$

$$Fun(x,a) := \left| \begin{array}{l} do_{\circ Fun_{\circ}}(s := 0, p := 1) \\ \text{if } a \geq 0 \\ \quad \left| \begin{array}{l} f(u) := \left| \begin{array}{l} \text{for } k \in [1..a] \\ \quad s := s + k \cdot u \end{array} \right. \\ s \end{array} \right. \\ \text{else} \\ \quad \left| \begin{array}{l} f(u) := \left| \begin{array}{l} \text{for } k \in [1..-a] \\ \quad p := p \cdot \frac{u}{k} \end{array} \right. \\ p \end{array} \right. \\ f(x) \end{array} \right.$$

1

$\text{appVersion}(4) = \text{"1.1.8763.0"}$