Kilts and CAS Extended

Session 267

11:00am – 12:00pm

Sat 8 March 2014

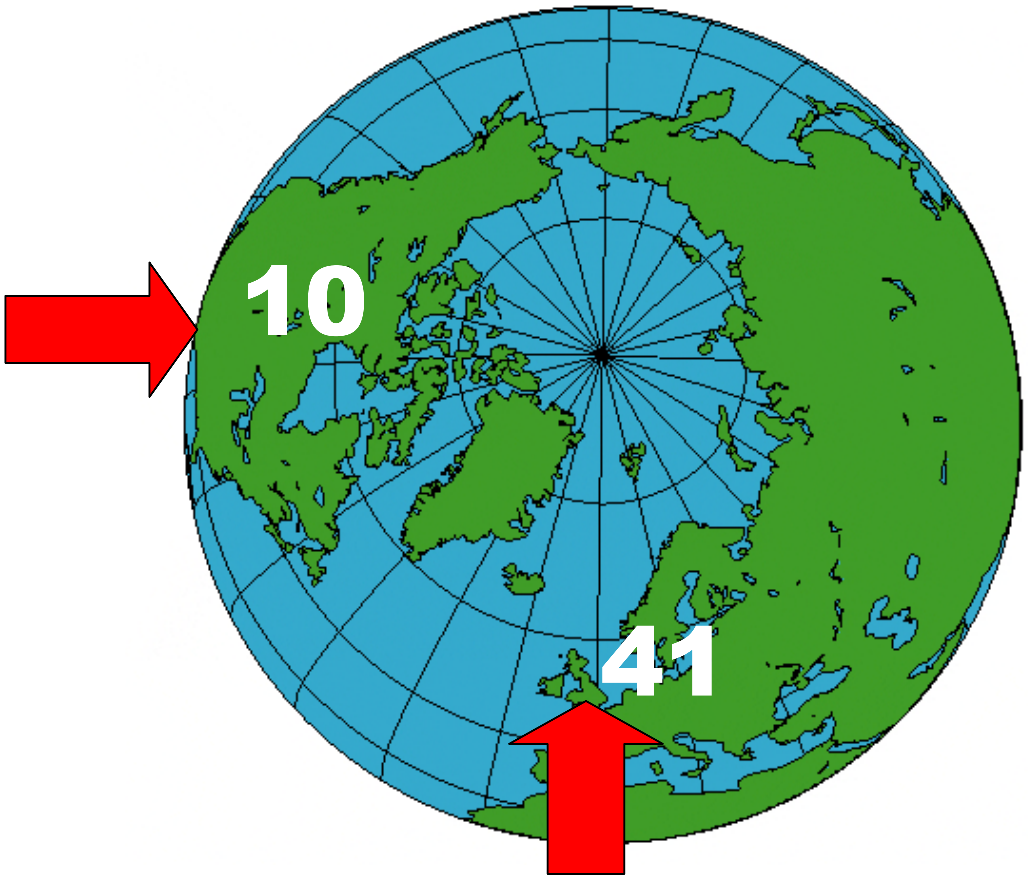
Tropical G
Rio Hotel and Casino

Nevil Hopley

T^3 National Trainer, Scotland & UK.
Head of Mathematics Department.

www.calculatorsoftware.co.uk/nspire

Nevil's from 4936 miles away on a Bearing of 033°



Strictly Limited Offer of Tartan TI-Nspire CX Cases at the end of this talk.

This talk will have a....

A Beginning

Background information about me & CAS, and the remit of this talk.

A Middle

Activities, Documents, Functions & Programs covering 7 different maths topics for students aged 12-18 years.

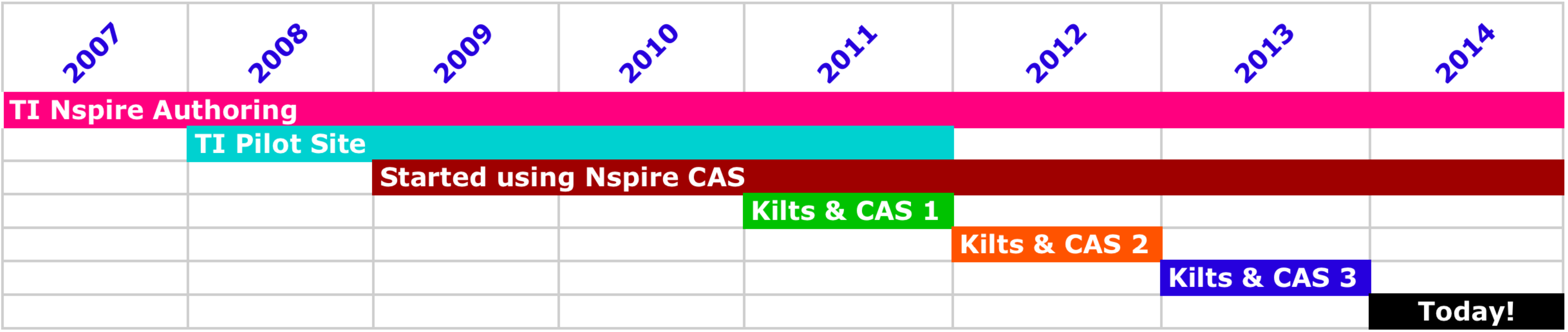
An End

Just before 12 noon!

And you can download all that you see today from

www.calculatorsoftware.co.uk/nspire

My CAS Timeline

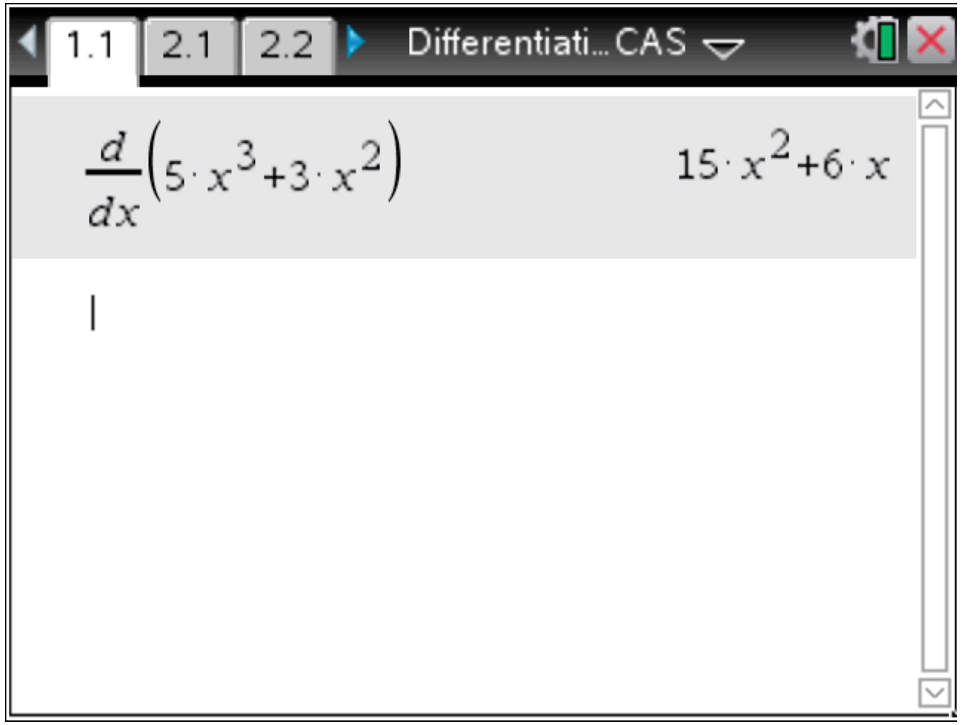


Kilts & CAS Talks @ T3 International Conferences

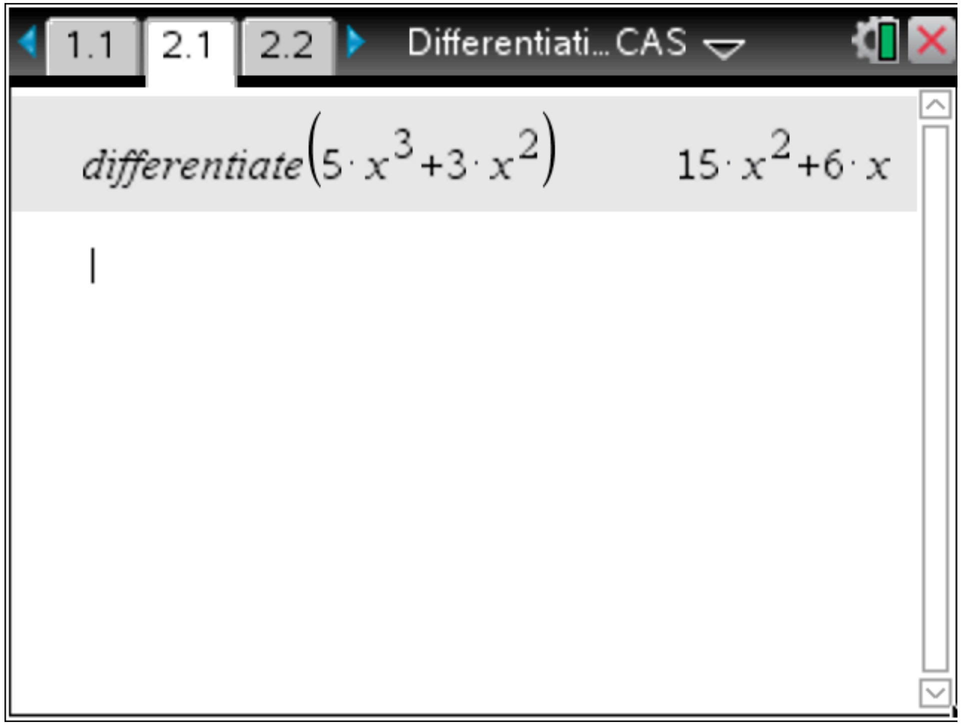
2011 My first 18 months of CAS usage
2012 The next 12 months of CAS usage
2013 Co-presented with Fred Ferneyhough
2014 Building CAS dependent functions & programs

Differentiation CAS.tns

The Issue

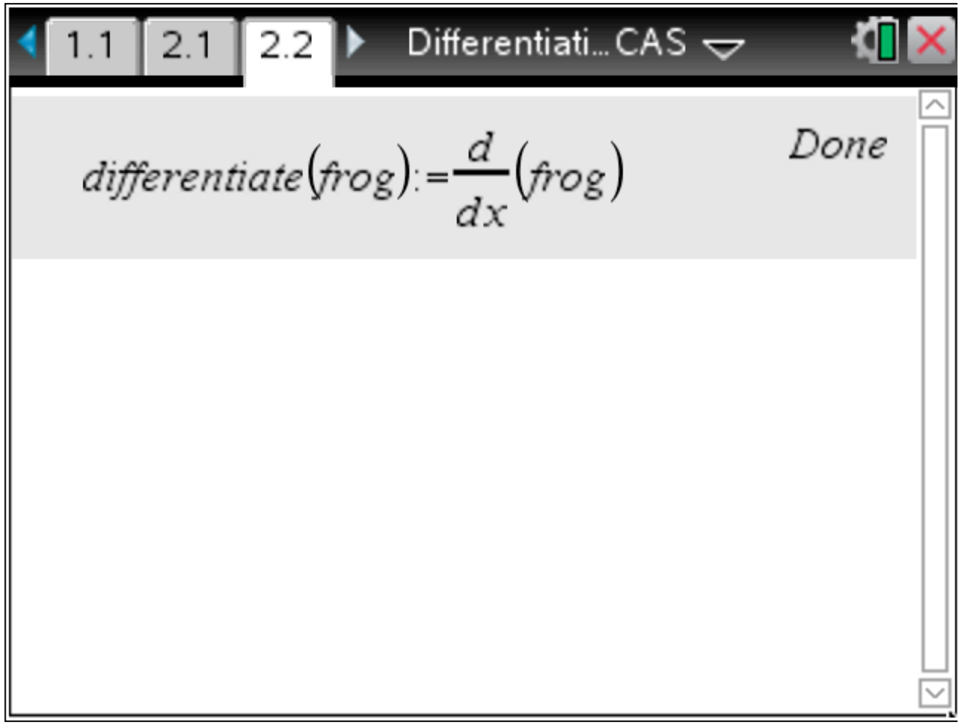


Much Nicer!



Differentiation CAS.tns

The Function



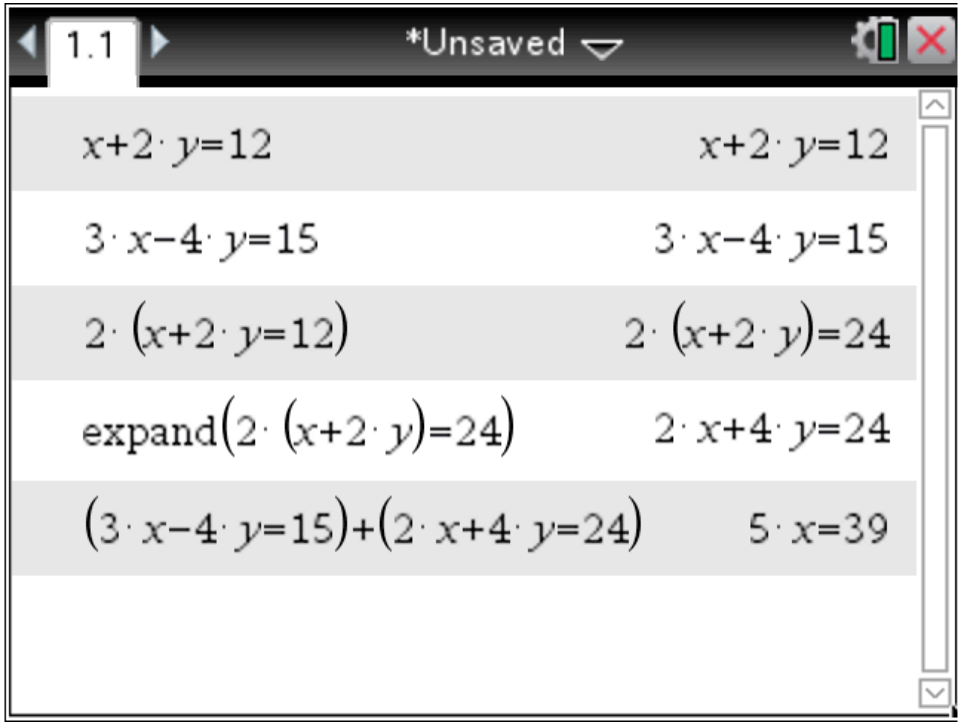
Notes

'frog' is a dummy variable. It can be anything, but not a system command.

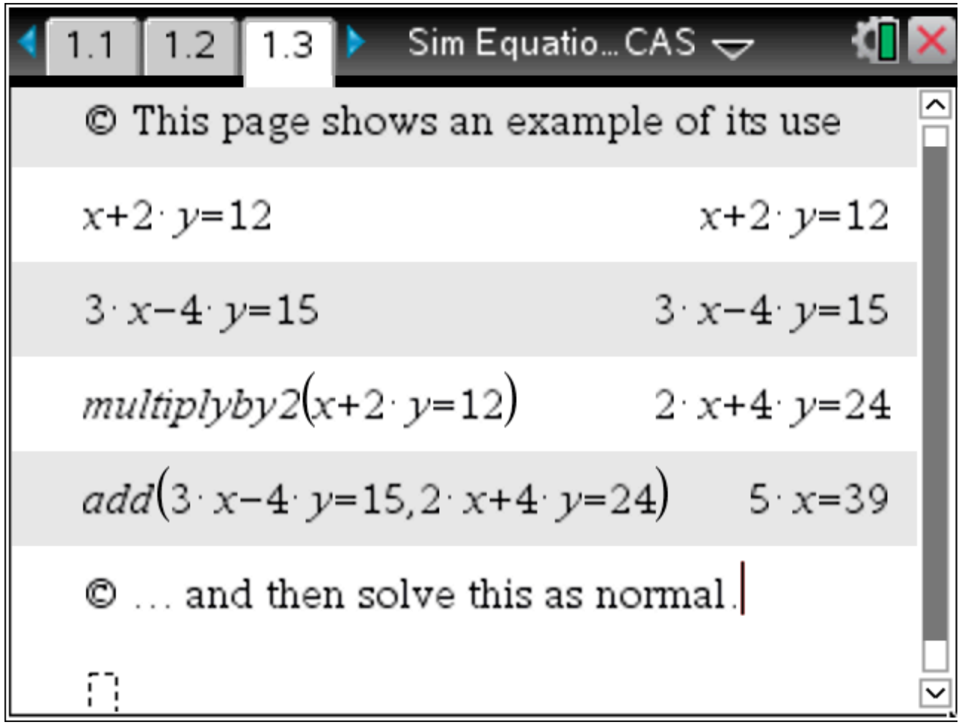
Limitation: no detection of the variable that's being differentiated with respect to. It's assumed to be x.

Sim Equations Toolkit CAS.tns

The Issue

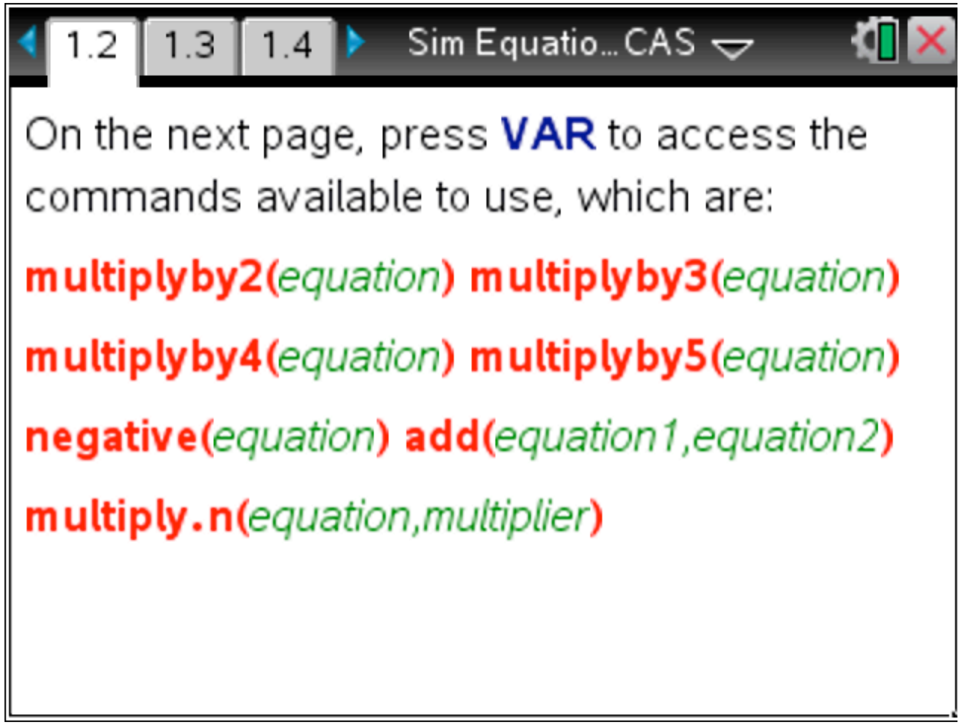


Much Nicer!



Sim Equations Toolkit CAS.tns

The Functions



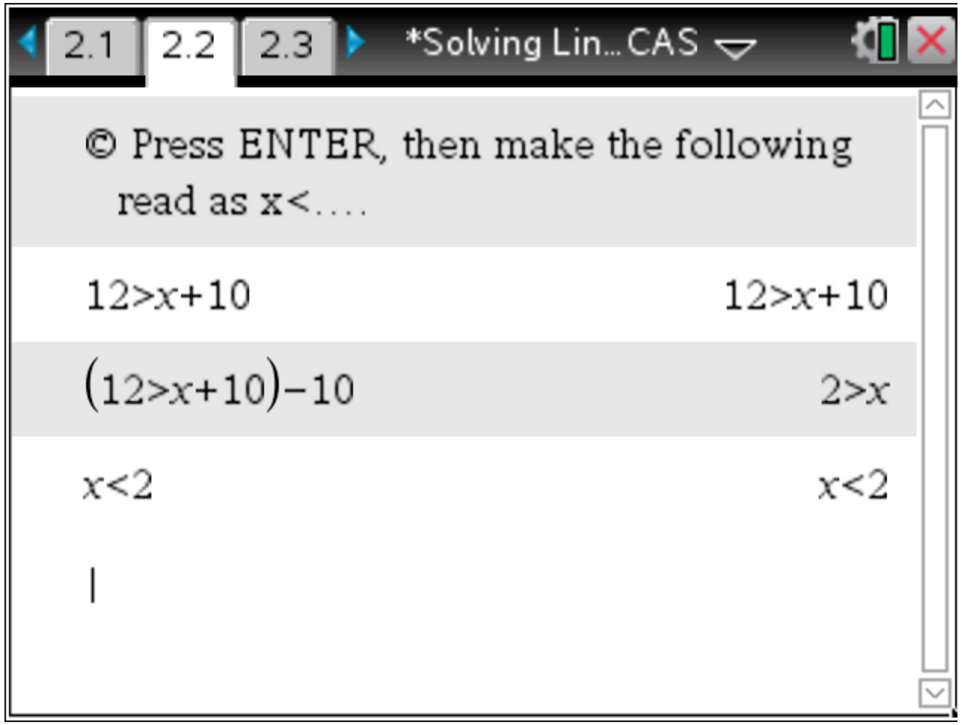
Notes

Go to pages 1.7 and 1.8 to view the underlying functions.

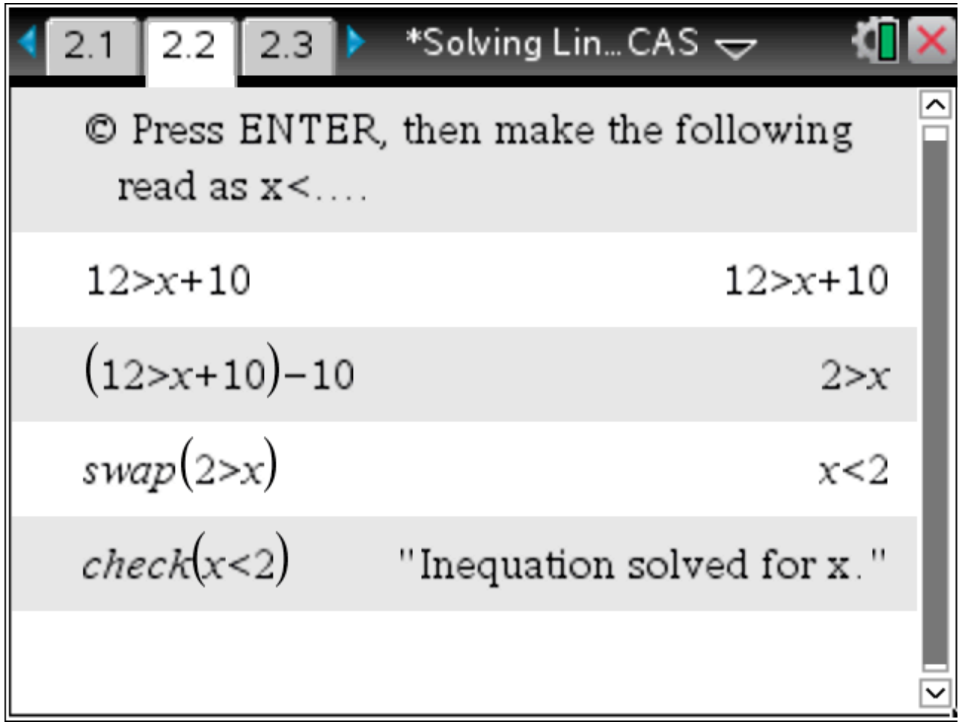
Ungroup each of these pages by pressing **CTRL** then **6**

Solving Linear Inequations CAS.tns

The Issue



Much Nicer!



Solving Linear Inequations CAS.tns

The Functions

page 1.2

swap(*expr*)

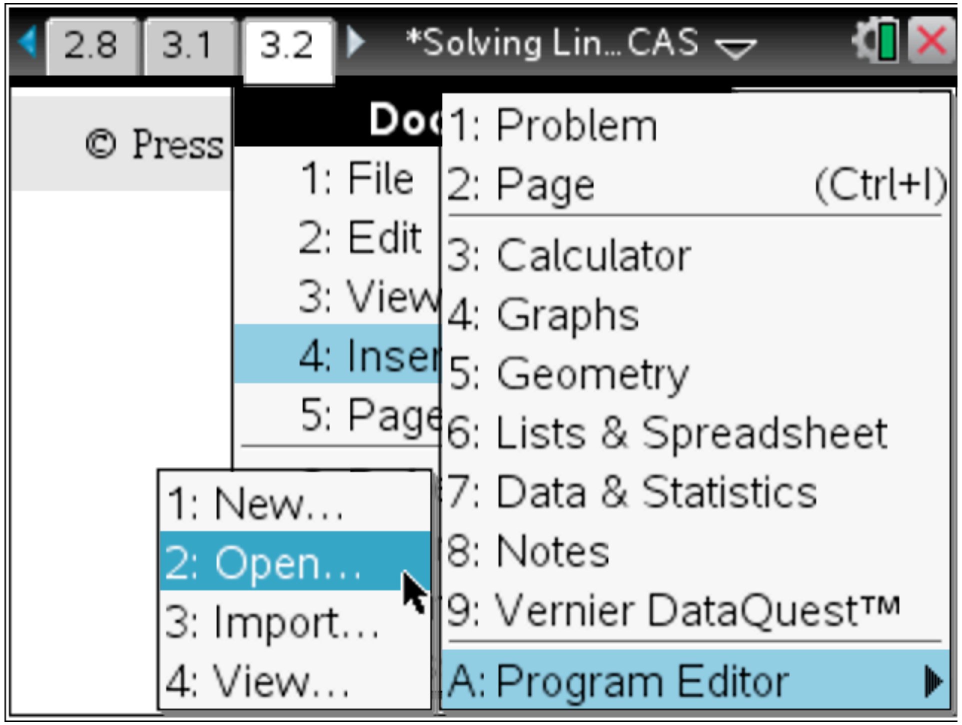
page 1.3

check(*expr*)

page 3.1

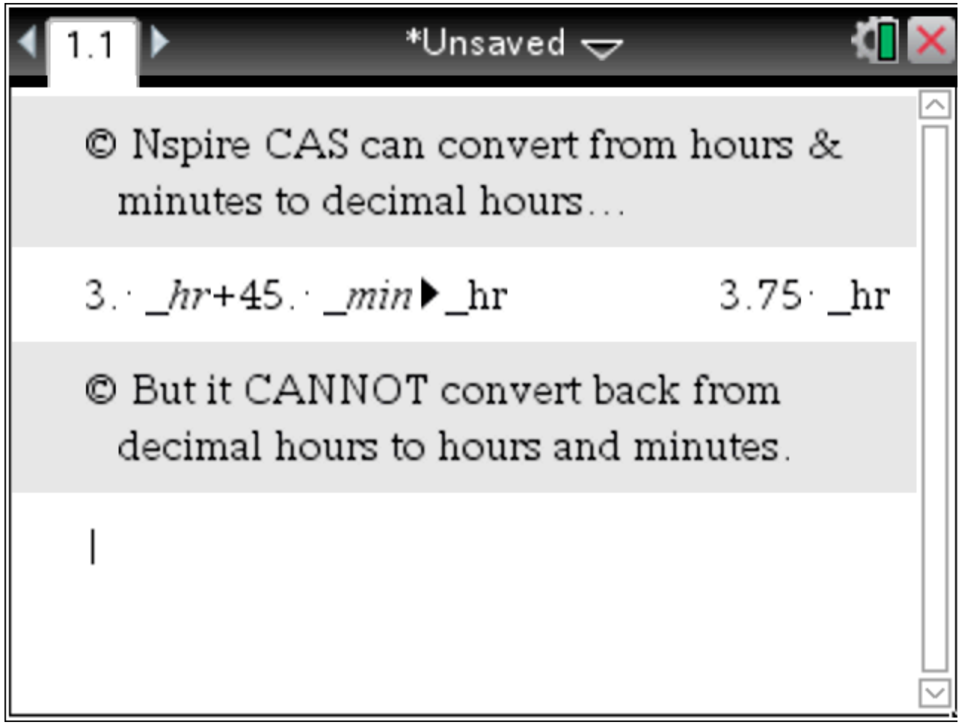
randominequation()

Notes

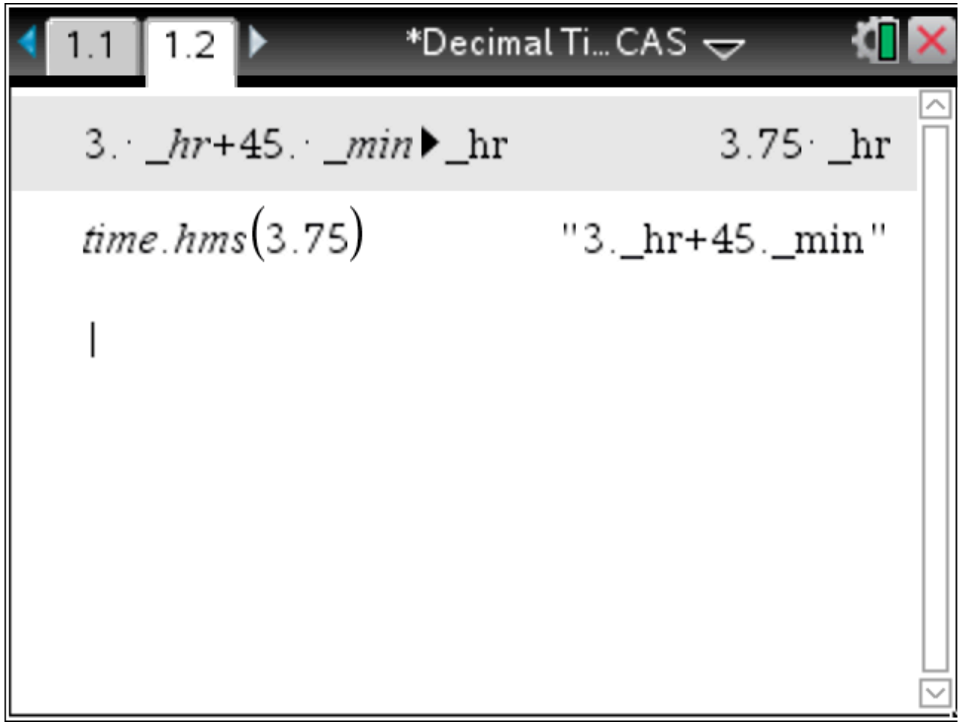


Decimal Time CAS.tns

The Issue



Much Nicer!



Decimal Time CAS.tns

The Function

time.hms 0/6

```
Define time.hms(time)=
Func
Local h,m,s,t
h:=iPart(time)
t:=60· fPart(time)
m:=iPart(60· fPart(time))
s:={0,                    m=t
   {round(60·(t−m),2),    m≠t
Return {string(h)&"_hr+"&string(m)&"_min",                           s=0
       {string(h)&"_hr+"&string(m)&"_min+"&string(s)&"_s",           s>0
EndFunc
```

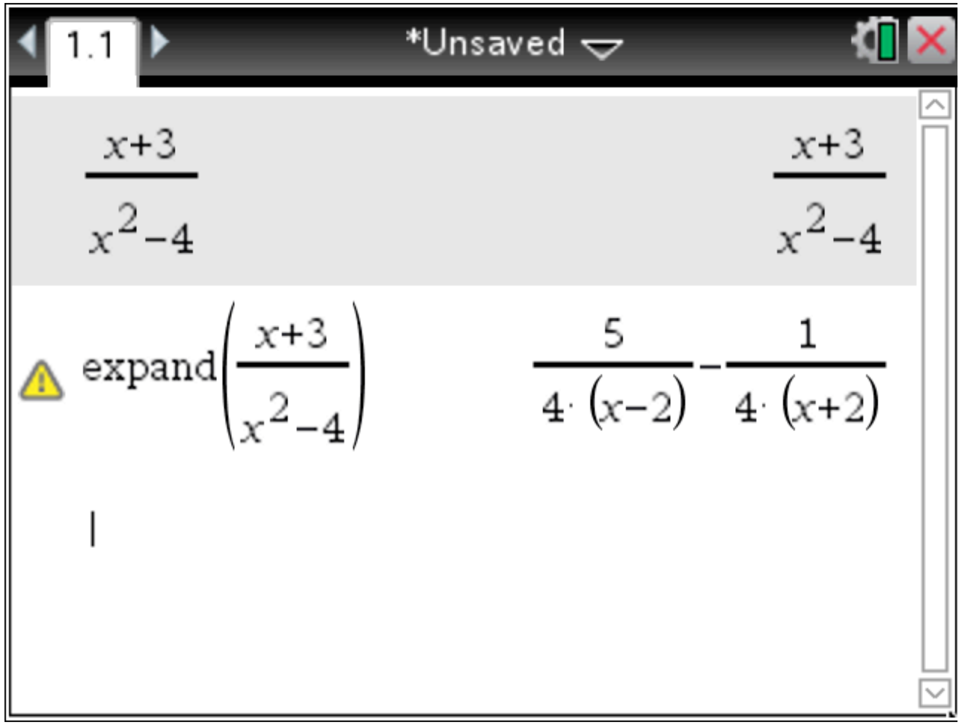
Notes

Piecewise functions are used instead of lots of 'if' statements

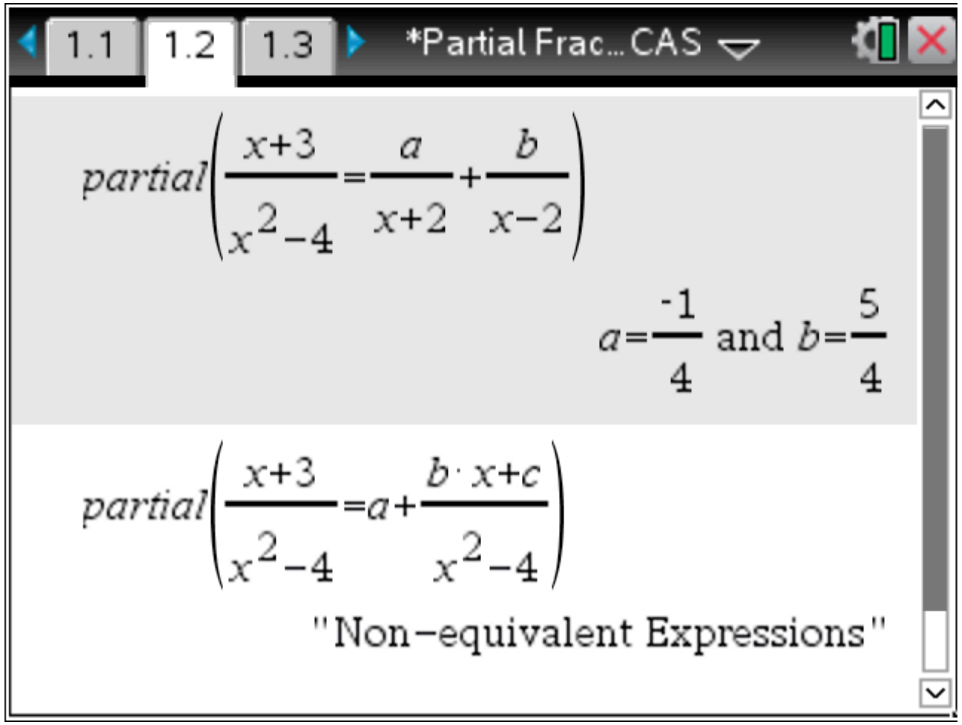
Text strings were needed at the end to prevent auto evaluation of the answer back to the original form!

Partial Fractions Checker CAS.tns

The Issue



More Helpful?



Partial Fractions Checker CAS

The Function

```
Define partial(expression)=
Func
Local lhslist,rhslist,eqstr,varlist,i,str
polyCoeffs(getNum(comDenom(left(expression))),x) → lhslist
polyCoeffs(getNum(comDenom(right(expression))),x) → rhslist
{ } → varlist
" " → eqstr
If dim(lhslist)=dim(rhslist) Then
varlist[1]:=expr(right(string(rhslist[1]),1))
© previous line finds just one of the variables used
For i,1,dim(lhslist)
eqstr&string(lhslist[i])&"="&string(rhslist[i])&" and " → eqstr
EndFor
left(eqstr,dim(eqstr)−5) → eqstr
"solve("&eqstr&","&string(varlist)&")" → str
Return expr(str)
Else
Return "Non−equivalent Expressions"
EndIf
EndFunc
```

Brief Notes

Separate LHS and RHS of expression to establish coeffs of each polynomial.

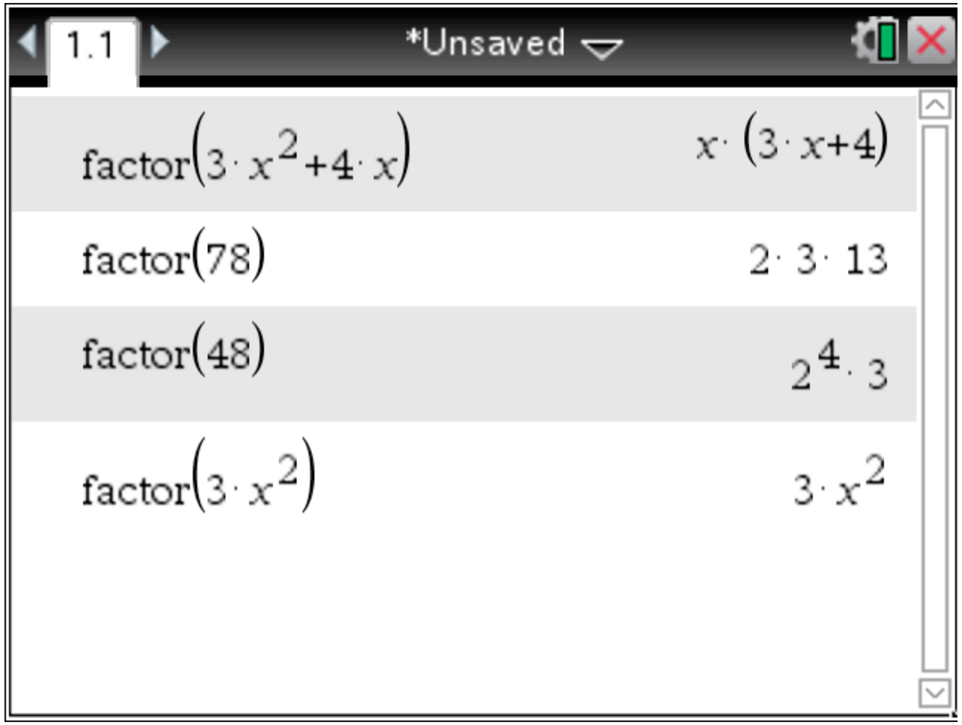
If they <u>are</u> equivalent powers, then pick just one of the variables used, for use later with the **solve(** command.

Cycle through the polynomials' coefficients, equating corresponding coeffs and building up an expression to then solve

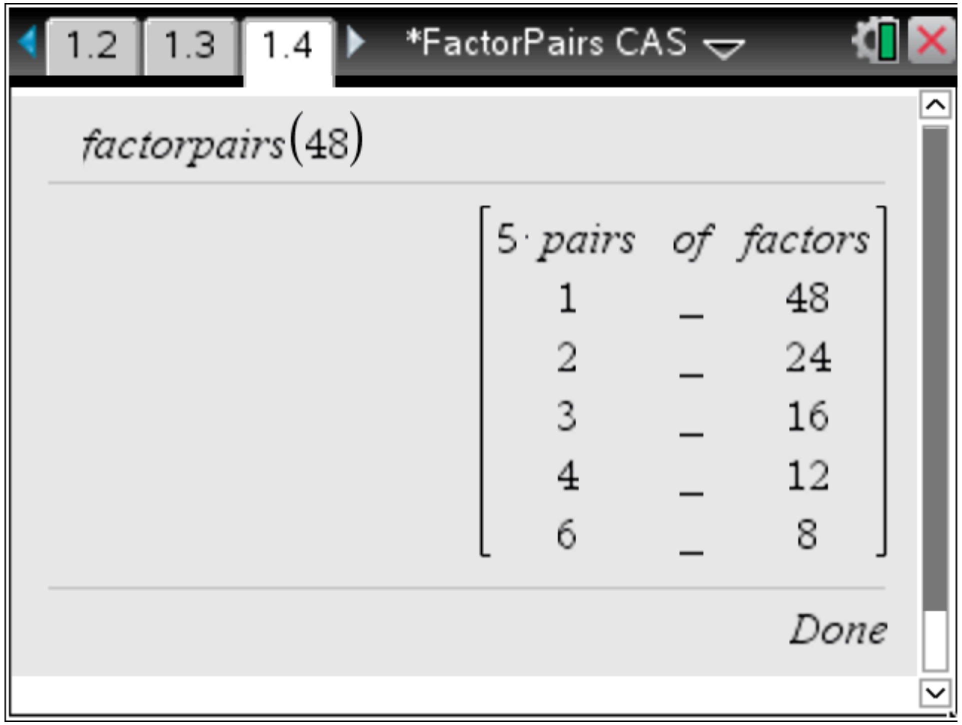
If powers were <u>not</u> equivalent, then return error message.

FactorPairs CAS.tns

The Issue



More Helpful



Teach…

Primes & Times Tables Backwards & Basics of Factorising & many, many other things!

FactorPairs CAS.tns

The Function

```
factorpairs                                                          0/76
Define factorpairs(term)=
Prgm
© Display factor pairs for any single term numeric or algebraic expression that has integer coefficients and powers.
© Written by Nevil Hopley, 22 July 2013
© More at www.calculatorsoftware.co.uk "/" nspire

Local ii,jj,kk,alpha,vectsum,constant,cfactors,tfactors,term1,term2,nextpair
alpha:=[[_][_][0]]

For ii,1,26
If zeros(term,expr(char(96+ii)))={0}
alpha:=augment(alpha,[[expr(char(96+ii))][0][0]])
EndFor

vectsum:=[1]
constant:=string(term)&"| true"
For ii,2,colDim(alpha)
vectsum:=augment(vectsum,[1])
constant:=constant&" and "&string(alpha[1,ii])&"=1"
While zeros(term/(alpha[1,ii]^(alpha[2,ii])),alpha[1,ii])={0}
alpha[2,ii]:=alpha[2,ii]+1
```



```
alpha[2,ii]:=alpha[2,ii]+1
```



```
EndWhile
EndFor

alpha[1,1]:=expr(constant)
cfactors:={ }
For ii,1,iPart(√(alpha[1,1]))
If mod(alpha[1,1],ii)=0
cfactors:=augment(cfactors,{ii})
EndFor
alpha[2,1]:=dim(cfactors)-1
tfactors:=[pairs  of  factors]
Loop

term1:=string(cfactors[alpha[3,1]+1])

© construct first factor, using row 3 of matrix
For ii,2,colDim(alpha)
term1:=term1&"·"&string(alpha[1,ii]^(alpha[3,ii]))
EndFor

term1:=expr(term1)
term2:=term/term1
tfactors:=(augment(tfactors^T,[term1  _  term2]^T))^T
If dotP(sum(alpha,3,3),vectsum)=dotP(sum(alpha,2,2),vectsum)
Exit
jj:=1
nextpair:=false

While nextpair=false

If alpha[3,jj]<alpha[2,jj] Then
© if can increase current register, then do so and conclude
  alpha[3,jj]:=alpha[3,jj]+1
  nextpair:=true
Else
© increase next register if allowed and reset previous registers to zero
  If alpha[3,jj+1]<alpha[2,jj+1] Then
    alpha[3,jj+1]:=alpha[3,jj+1]+1
    For kk,1,jj
    alpha[3,kk]:=0
    EndFor
    nextpair:=true
  Else
    jj:=jj+1
  EndIf
EndIf

EndWhile

EndLoop
If rowDim(tfactors)>2 Then
```



```
If rowDim(tfactors)>2 Then
```



```
tfactors[1,1]:=(rowDim(tfactors)-1)· pairs
Else
tfactors[1,1]:=only· two
tfactors[1,2]:=distinct
EndIf
Disp tfactors
EndPrgm
```

Phew!

FactorPairs CAS.tns

Broad Description of How It Works …

- Find all the single letter variables in the entered expression, using the zeros command and set them all up in a matrix.
- Find all the powers of each variable and record them in the 2^{nd} row of the matrix.
- Find the value of any constant coefficient, by setting all the variables to have value 1.
- Find all the integer factors of the constant, from 1 up to √constant and store them in the list called *cfactors*.
- Construct the first factorpair using the first member of *cfactors* and all the variables from the matrix with each of their powers initially set to 0 (call it *term1*) and dividing the entered term by it, to give *term2*.
- Cycle through the members of *cfactors* and each of the variables in the matrix, increasing their powers in turn, up to their maximums. Then ‘reset’ each power and move onto the next variable.
- Closing commands adjust the phrasing of the output if there was only one factorpair for the entered expression.

FactorPairs CAS.tns

See the algorithm in action …

1.2 | 1.3 | 1.4 — *FactorPairs CAS

$factorpairs\left(6\cdot a^{3}\cdot b^{2}\right)$

24· pairs	of	factors
1	_	$6\cdot a^{3}\cdot b^{2}$
2	_	$3\cdot a^{3}\cdot b^{2}$
a	_	$6\cdot a^{2}\cdot b^{2}$
$2\cdot a$	_	$3\cdot a^{2}\cdot b^{2}$
a^{2}	_	$6\cdot a\cdot b^{2}$
$2\cdot a^{2}$	_	$3\cdot a\cdot b^{2}$
a^{3}	_	$6\cdot b^{2}$
$2\cdot a^{3}$	_	$3\cdot b^{2}$
b	_	$6\cdot a^{3}\cdot b$
$2\cdot b$	_	$3\cdot a^{3}\cdot b$
$a\cdot b$	_	$6\cdot a^{2}\cdot b$
$2\cdot a\cdot b$	_	$3\cdot a^{2}\cdot b$
$a^{2}\cdot b$	_	$6\cdot a\cdot b$
$2\cdot a^{2}\cdot b$	_	$3\cdot a\cdot b$
$a^{3}\cdot b$	_	$6\cdot b$
$2\cdot a^{3}\cdot b$	_	$3\cdot b$
b^{2}	_	$6\cdot a^{3}$
$2\cdot b^{2}$	_	$3\cdot a^{3}$
$a\cdot b^{2}$	_	$6\cdot a^{2}$
$2\cdot a\cdot b^{2}$	_	$3\cdot a^{2}$
$a^{2}\cdot b^{2}$	_	$6\cdot a$
$2\cdot a^{2}\cdot b^{2}$	_	$3\cdot a$
$a^{3}\cdot b^{2}$	_	6
$2\cdot a^{3}\cdot b^{2}$	_	3

Only 30 in the whole world!

Suggested minimum
donation of $15……

All proceeds to

CANCER RESEARCH UK

Want Copies of Everything?

www.CalculatorSoftware.co.uk/nspire

Thank you for coming to my talk.

Nevil Hopley

T^3 National Trainer, Scotland & UK.

Head of Mathematics Department

CAS user on Handhelds and TI-Nspire iPad App
TI-Basic and Lua Programmer

Mountain Unicycler

Source for Highland Dancer Image Acknowledged from:
http://www.scotsman.com/what-s-on/theatre-comedy-dance/matthew-bourne-on-the-scottish-ballet-s-highland-fling-1-2907738