

# printme

April 18, 2023

This file is part of CasADi.

CasADi -- A symbolic framework for dynamic optimization.  
Copyright (C) 2010-2023 Joel Andersson, Joris Gillis, Moritz Diehl,  
KU Leuven. All rights reserved.  
Copyright (C) 2011-2014 Greg Horn

CasADi is free software; you can redistribute it and/or  
modify it under the terms of the GNU Lesser General Public  
License as published by the Free Software Foundation; either  
version 3 of the License, or (at your option) any later version.

CasADi is distributed in the hope that it will be useful,  
but WITHOUT ANY WARRANTY; without even the implied warranty of  
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU  
Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public  
License along with CasADi; if not, write to the Free Software  
Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

```
[1]: from casadi import *
```

```
[2]: a = SX.sym("a")  
     b = SX.sym("b")
```

```
[3]: c = a+b  
     c = c.printme(13)
```

```
[4]: d = c**2
```

```
[5]: print(d)
```

```
sq(printme((a+b),13))
```

```
[6]: f = Function("f", [a,b],[d])
```

When the graph is evaluated, a printout of c will occur (if you have set WITH\_PRINTME to ON in CMakeCache.txt) Printout reads '|> 13: 7' 13 is an identifier of choice, 7 is the numerical value

of c

```
[7]: f(4,3)
```

```
|> 13 : 7.000000000000000e+00
```

```
[7]: DM(49)
```

```
[8]: dd_da = jacobian(d, a)
      J = Function('J', [a,b], [dd_da])
```

The first derivative still depends on c Printout reads '|> 13: 11'

```
[9]: J(2,9)
```

```
|> 13 : 1.100000000000000e+01
```

```
[9]: DM(22)
```

```
[10]: d2d_da2 = jacobian(dd_da, a)
       J = Function('J', [a,b], [d2d_da2])
```

second derivative doesn't, so we don't get a printout

```
[11]: J(2,9)
```

```
[11]: DM(2)
```