

## [7.8] Return program results to home screen

A TI-89/TI-92 Plus *function* can return a result to the command line, but a *program* usually cannot. However, there are two methods to get around this limitation.

*Method 1: Samuel Stearly's copyto\_h():*

Samuel Stearly has written an ASM program that copies program results to the home screen. For the latest version, search for *copyto\_h* on ticalc.org. The call syntax for *copyto\_h()* is

```
copyto_h([name1[,name2]])
```

*name1* and *name2* are variable name strings. If no arguments are passed, *copyto\_h()* displays the call syntax in the status line. If *name1* is passed, the contents of *name1* are copied to the entry and answer areas of the home screen. If *name1* and *name2* are passed, the contents of *name1* and *name2* are copied to the entry and answer areas, respectively. This feature provides a very convenient way to label multiple results. *name1* and *name2* may include a folder specification. This simple test routine demonstrates the usage:

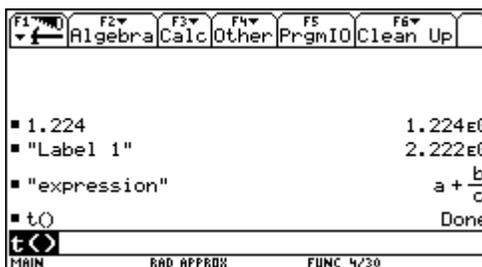
```
t()
Prgm
local v1,v2,v3,v4,v5

1.224→v1
"Label 1"→v2
2.222→v3
"expression"→v4
a+b/c→v5

util\copyto_h("v1")
util\copyto_h("v2","v3")
util\copyto_h("v4","v5")

EndPrgm
```

For this example, *copyto\_h()* is installed in the *util* folder on the calculator, so the function calls specify that folder. After running *t()*, the home screen looks like this:



The first call to *copyto\_h()* has only one argument (1.224), so that argument is placed in both the entry and answer columns of the history area. The next two calls have two arguments, so the first argument (the strings) is placed in the entry column, and the second argument is placed in the answer column. The last call shows that expressions are pretty-printed, as expected. Note that multiple calls to *copyto\_h()* push each result up in the history display.

*copyto\_h()* seems to run reliably on both HW1 and HW2 calculators, running AMS 2.05. I have had no problem with this routine, but there is always the risk that this type of program could cause unreliable

calculator operation or a crash. I know of *no* problems that have been reported, and this program is so useful that I accept the small amount of risk that is involved. Mr. Stearly is to be congratulated for this accomplishment.

Mr. Stearly has also written:

*sendstr()* which works only with keyboard programs (*kbdprgmx()*) to send a string to the entry line:  
<http://www.ticalc.org/archives/files/fileinfo/166/16623.html>

*copytobc()* which works with keyboard programs to copy an expression to the calculator clipboard:  
<http://www.ticalc.org/archives/files/fileinfo/146/14628.html>

*Method 2: expr(result & ":stop")*

If the last line of your program is

```
expr(result & ":stop")
```

where *result* is a string, then *result* will be returned to the history area. For example,

```
expr("10"&":stop")
```

will return

```
10 : Stop
```

Now, this isn't exactly what we want, since the ":Stop" is tagged onto the result, but at least it works. The ":Stop" can be edited out.

Timité Hassan provides this routine *tohome()* to return results to the history area:

```
tohome(lres)
Prgm
Local zkk,execstr
""->execstr
If dim(lres)=0:return
For zkk,1,dim(lres)
  lres->wstr
  If instring(lres[zkk],"@")=1then
execstr&string(mid(lres[zkk],2))&":->execstr
  Else
  execstr&lres[zkk]&":->execstr
  endif
Endfor
expr(execstr&":stop")
EndPrgm
```

*lres* is a list of strings to return to the history area. You can use the "@" character to add labels to the results. Note that the input string *lres* is stored to the global variable *wstr*, so that you can also recall *wstr* to get the results.

Some examples:

```
tohome>{"45"} returns 45 : Stop
```

`tohome({"45","x=6*po"})` returns `45 : x=6*po : : Stop`

`tohome({"@res1","45","@res2","x=6*po"})` returns `"res1" : 45 : "res2" : x=6*po : : Stop`

cj points out that *tohome()* does not work if called from a keyboard program; those executed with `[DIAMOND][n]`, where *n* is a number from 1 to 9. However, *tohome()* works if *kbdprgmn()* is called from the command line.

*(Credit to Timité Hassan and cj for Method 2)*