[7.42] Find variable names used in expressions

It is often useful to extract the variable names from an expression or equation. For example, the variables can be deleted, archived, initialized and so on. Timité Hassan and Bhuvanesh Bhatt, have both written functions to extract variable names. This tip discusses both methods.

Timité's TI Basic version

Timité's function *getvars*() returns the variable names of an expression passed as a string. I have slightly modified his function, and changed the name to *exprvars()* to avoid confusion. My modifications change the external subroutines to local routines, and convert some nested-If expressions to *when()* functions. The basic algorithm and code remains unchanged.

```
exprvars(g_str)
Func
©("expr") return list ofvariables in "expr"
©Timité Hassan, modified by dab

Local k,i,cc,g_vr,g_d,g_dim,g_ll,ss,varin,vardeb

Define varin(c)=Func
Local cc
ord(c)→cc
when(c≥"a" and c≤"z",true,when(c≥"Ø" and c≤"9",true,when(cc≥128 and cc≤148 and
cc≠14Ø,true,when(c="_",true,false))))
EndFunc

Define vardeb(c)=Func
Local cc
ord(c)→cc
when(c≥"a" and c≤"z",true,when(cc≥128 and cc≤148 and cc≠14Ø,true,false))
EndFunc

dim(g_str)→g_dim
{}→g_ll
Ø→g_d
For k,1,g_dim
 mid(g_str,k,1)→cc
 if cc≥"A" and cc≤"Z":char(ord(cc)+32)→cc
 If vardeb(cc) then
  cc→ss
  k+1→k
  mid(g_str,k,1)→cc
  if cc≥"A" and cc≤"Z":char(ord(cc)+32)→cc
  while k≤g_dim and varin(cc)
   ss&cc→ss
   k+1→k
   mid(g_str,k,1)→cc
   if cc≥"A" and cc≤"Z":char(ord(cc)+32)→cc
  Endwhile
  If cc≠"(" and ss≠"and" and ss≠"or" and ss≠"xor" then
   For i,1,g_d
    if ss=g_ll[i]:exit
   Endfor
   If i>g_d then
    augment(g_ll,{ss})→g_ll
    g_d+1→g_d
   Endif
  Endif
 Endif
Endfor
g_ll
Endfunc
```

Note that the expression is passed as a string, and need not be a valid expression. The variable names are returned as string elements of a list. For example, the call

```
exprvars("a+b/c+sin(d^e)+22.2=f")
```

returns

```
{"a","b","c","d","e","f"}
```

Timité Hassan's original getvars() function

This is Timité's original *getvar()* code. I include it here out of courtesy, and so that if I have made in mistake in my changes, you can use the original code. *getvars()* calls *varin()* and *vardeb()*.

getvars():

```
getvars(g_str)
Func
Local k,i,cc,g_vr,g_d,g_dim,g_ll,ss

dim(g_str)→g_dim
{}→g_ll
Ø→g_d
For k,1,g_dim
mid(g_str,k,1)→cc
if cc≥"A" and cc≤"Z"
char(ord(cc)+32)→cc
If vardeb(cc) then
cc→ss
k+1→k
mid(g_str,k,1)→cc
if cc≥"A" and cc≤"Z"
char(ord(cc)+32)→cc
while k≤g_dim and varin(cc)
ss&cc→ss
k+1→k
mid(g_str,k,1)→cc
if cc≥"A" and cc≤"Z"
char(ord(cc)+32)→cc
Endwhile
If cc≠"(" and ss≠"and" and ss≠"or" and ss≠"xor" then
For i,1,g_d
if ss=g_ll[i]
exit
Endfor
If i>g_d then
augment(g_ll,{ss})→g_ll
g_d+1→g_d
Endif
Endif
Endif
Endfor
g_ll
Endfunc
```

vardeb():

```
vardeb(c)
Func
Local cc
ord(c)→cc
If c≥"a" and c≤"z" then
 Return true
```

```
Elseif cc≥128 and cc≤148 and cc≠140 then
 Return true
Endif
False
EndFunc
```

varin():

```
varin(c)
Func
Local cc
ord(c)→cc
If c≥"a" and c≤"z" then
 Return true
Elseif c≥"0" and c≤"9" then
 Return true
Elseif cc≥128 and cc≤148 and cc≠140 then
 Return true
Elseif c="_" then
 Return true
Endif
False
EndFunc
```

Bhuvanesh' C version: varlist()

Bhuvanesh' version of this function is called *VarList()*, and differs functionally from *exprvars()* in that the input argument can be passed as a string or an expression, and an optional input argument excludes user functions from the variable list.

```
VarList(expr|string,{excludeFunctions})
```

where *expr* | *string* is the expression or string. Set *excludeFunctions* to 1 to exclude user functions from the list, or omit it to include user functions. *VarList()* must be executed in Auto or Exact mode.

Some examples:

`VarList(a+b*x)`	returns	{a, b, x}	
`VarList("a+b")`	returns	{a, b}	
`VarList(f(x,y))`	returns	{y, x, f}	*(function name included*)
`VarList(f(x,y),1)`	returns	{y, x}	*(function name excluded*)