Subject: Making IDL Code Available: linked list modules
Posted by Doug Larson on Wed, 02 Sep 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Here are some modules for linked lists. They are posted "as-is" to
spur others to write a better linked list library.

Cheers,
Doug, PiledHigher and Deeper

--
Douglas J. Larson e-mail: djl@loki.srl.caltech.edu
Space Radiation Lab
California Institute of Technology
Mail Code: 220-47
Pasadena, CA 91125

```
;  ************************************************************ ******************
;+
; Setup_llist.pro
;
; PURPOSE:
;       Setup a new node of a generic linked list.
;
; CATEGORY:
;       Generic linked list generator.
;
; CALLING SEQUENCE:
; somenewnode = ptr_new(Setup_llist(bugmsg))
;
; KEYWORD PARAMETERS:
; bugmsg        - Debugging message activator (0-No messages, 1-verbose mode)
;
; COMMON BLOCKS:
; None
;
; OUTPUTS:
; Data structure
;
; MODIFICATION HISTORY:
; Created by: Douglas J. Larson, 18 August, 1998.
;-
;  ************************************************************ ******************
FUNCTION Setup_llist, bugmsg

 if ( bugmsg eq 1 ) then message,' Entered', /INFORMATIONAL
```

```
 lls = { nodePtr:PTR_NEW(), $ ; Points to the linked list information
         next:PTR_NEW() } ; Pointer to the next data vector

 if ( bugmsg eq 1 ) then message,' Exited', /INFORMATIONAL

 RETURN, lls

END
;  ************************************************************ ******************
;+
; llist_insert.pro
;
; PURPOSE:
;       Setup a new node of a generic linked list.
;
; CATEGORY:
;       Generic linked list generator.
;
; CALLING SEQUENCE:
;
; KEYWORD PARAMETERS:
; bugmsg        - Debugging message activator (0-No messages, 1-verbose mode)
;
; COMMON BLOCKS:
; None
;
; OUTPUTS:
; Data structure
;
; MODIFICATION HISTORY:
; Created by: Douglas J. Larson, 18 August, 1998.
;-
;  ************************************************************ ******************
FUNCTION llist_insert, bugmsg, startPtr, value

 if ( bugmsg eq 1 ) then message,' Entered', /INFORMATIONAL

  if NOT(PTR_VALID(startPtr)) then begin
    if ( bugmsg eq 1 ) then message,'startPtr is NULL', /INFORMATIONAL
    startPtr = PTR_NEW(Setup_llist(bugmsg))
    currentPtr = startPtr
  endif else if (PTR_VALID(startPtr)) then begin
    if ( bugmsg eq 1 ) then message,'startPtr is Valid', /INFORMATIONAL
    currentPtr=startPtr
    while (PTR_VALID(currentPtr)) do begin
  previousPtr=currentPtr
  currentPtr=(*currentPtr).next
```

```
   endwhile
   (*previousPtr).next = PTR_NEW(Setup_llist(bugmsg))
   currentPtr=(*previousPtr).next
 endif
(*currentPtr).nodePtr = value
(*currentPtr).next = PTR_NEW()

 if(bugmsg eq 1 ) then begin
   if NOT(PTR_VALID(value)) then begin
          message,'value is NULL', /INFORMATIONAL
   endif else begin
          message,'value is Valid', /INFORMATIONAL
   endelse
   if(NOT(PTR_VALID(startPtr)) or NOT(PTR_VALID(currentPtr)))then begin
          print,'PTR_VALID(startPtr)=',PTR_VALID(startPtr)
          print,'PTR_VALID(nextPtr)=',PTR_VALID(nextPtr)
          print,'PTR_VALID(currentPtr)=',PTR_VALID(currentPtr)
   endif
 endif

 if ( bugmsg eq 1 ) then message,'Exited', /INFORMATIONAL

 RETURN, currentPtr

END
;  ************************************************************ ******************
;+
; PURPOSE:
; Return the pointer to the desired record in the linked list.
;
; CALLING SEQUENCE:
;
; EXAMPLE:
;
; REQUIRED ROUTINES:
;
; VARIABLES:
; bugmsg       - Debugging message activator (0-No messages, 1-verbose mode)
;
; MODIFICATION HISTORY:
;       Created: Douglas J. Larson, 05 August, 1998.
;
;-
;  ============================================================
==================
function llist_lookup, bugmsg, startPtr, key, criteria

 if(bugmsg eq 1) then message,' Entered', /INFORMATIONAL
```

```
 if NOT(PTR_VALID(startPtr)) then begin
    if ( bugmsg eq 1 ) then message,'startPtr is NULL', /INFORMATIONAL
    retval = PTR_NEW()
 endif else if (PTR_VALID(startPtr)) then begin
    if ( bugmsg eq 1 ) then message,'startPtr is Valid', /INFORMATIONAL

    ; Create a pointer to the heap variable pointed at by _startPtr_.
    currentPtr = startPtr

          uckey = STRUPCASE(key)

     tag_number = where(tag_names((*(*currentPtr).nodeptr)) eq uckey, count)
    if(bugmsg eq 1) then begin
             print,'counter=',count,' uckey=',uckey,' tag_number=',tag_number
    endif

    done = 0
    while (done eq 0) do begin
      if(bugmsg eq 1) then begin
     text='Tag String Requested='+key
     message, text, /INFORMATIONAL
     help,criteria
     ;help,(*currentPtr).(tag_number[0])
      endif
      if(criteria eq (*(*currentPtr).nodeptr).(tag_number[0])) then begin
        retval = (*currentPtr).nodeptr
        done = 1
      endif else begin
        ; Move to the next link in the list
        currentPtr = (*currentPtr).next
        if(ptr_valid(currentPtr) eq 0) then begin
          retval = ptr_new()
          done = 0
        endif
      endelse
    endwhile
 endif

 if(bugmsg eq 1) then message,' Exited', /INFORMATIONAL

 RETURN, retval
END
; ************************************************************** *****************
;+
; llist_print.pro
;
; PURPOSE:
```

```
;       Print a specified node element of an entire generic linked list.
;
; CATEGORY:
;       Generic singly linked list.
;
; CALLING SEQUENCE:
;       llist_print, bugmsg, startPtr, fieldwanted
;
; KEYWORD PARAMETERS:
;
; COMMON BLOCKS:
;       None
;
; OUTPUTS:
;       Data structure
;
; MODIFICATION HISTORY:
;       Created by: Douglas J. Larson, 22 August, 1998.
;-
;  ============================================================
==================
FUNCTION llist_print, bugmsg, startPtr, fieldwanted

if ( bugmsg eq 1 ) then message,' Entered', /INFORMATIONAL

retval = 1
if NOT(PTR_VALID(startPtr)) then begin
   if ( bugmsg eq 1 ) then message,'startPtr is NULL', /INFORMATIONAL
   retval = 0
endif else if (PTR_VALID(startPtr)) then begin
   ; Create a second pointer to the heap variable pointed at by _startPtr_.
   currentPtr = startPtr

   ; IDL likes all structure tags in uppercase.
   fieldwanted = strupcase(fieldwanted)

   tag_number = where(tag_names((*(*currentPtr).nodeptr)) eq fieldwanted, count)

   if(count eq 0)then begin
      message,'Tag is NOT in structure, check the code!'
      retval = 0
   endif else begin
      ; Traverse the linked list and print the list element information.
      while PTR_VALID(currentPtr) do begin
        print, currentPtr, ',named ', $
              (*(*currentPtr).nodeptr).(tag_number[0]), $
              ', points to: ', (*currentPtr).next
        currentPtr = (*currentPtr).next
```

```
    endwhile
  endelse
  ; PTR_FREE,currentPtr
endif

if ( bugmsg eq 1 ) then message,'Exited', /INFORMATIONAL

return, retval
end
```

File Attachments

1) llist.txt, downloaded 64 times