Subject: Re: Push value onto an array?
Posted by Rick Towler on Fri, 17 Jan 2003 01:56:40 GMT
View Forum Message <> Reply to Message

"Ed Wright" wrote >
> Question,
>
> How can I push a value onto an array?
>
> Given some array A of size m, how to add an entry at A[m]?
>
> Something like
>
> push @A, $value;
>
> This needs to be done in a loop. Yes, really.

I think this was covered in depth a month or two ago. I just googled one entitled "How to use pointers instead of arrays" which covers some of the details (it quickly diverges from pointers and addresses your question directly).

-Rick

Subject: Re: Push value onto an array?
Posted by Craig Markwardt on Fri, 17 Jan 2003 04:10:49 GMT
View Forum Message <> Reply to Message

Ed Wright <ed.wright@jpl.nasa.gov> writes:

> Question,
>
> How can I push a value onto an array?
>
> Given some array A of size m, how to add an entry at A[m]?
>
> Something like
>
> push @A, $value;

Greetings!

The easiest, or rather fastest, way I've found is to preallocate a bigger array, and then place the values in the array rather than grow it each time.

Here is some example code. It's not super documented, but you
basically use it like this:

```
 ;; Start with STACK and NSTACK undefined
 for i = 0, nvals-1 do pds_push, stack, vals(i), nstack=nstack, template=0L
 ;; ... but you could push all VALS at once if you want

 ;; And later to pop values off,
 for i = 0, nvals-1 do vali = pds_pop(stack, nstack=nstack)
 ;; ... and again, you can pop more if you want.
```

Good luck,
Craig

```
;; Push a node onto the tree
;;
;;  STACK - (INPUT) original stack plus pre-allocated entries
;;          (OUTPUT) modified stack plus pre-allocated entries
;;          user need not modify or access this variable
;;  VAL - values to be pushed onto stack.  Must be same data type as STACK
;;  NSTACK - (INPUT) number of values in original stack
;;           (OUTPUT) number of values in modified stack
;;           user need not modify this variable except to set to zero
;;           DEFAULT: n_elements(stack)
;;  TEMPLATE - value to be used to expand pre-allocated entries in STACK
;;             DEFAULT: VAL(0)
;;
pro pds_push, stack, val, nstack=nstack, template=node0
  nvals = n_elements(val)
  if n_elements(nstack) EQ 0 then nstack = n_elements(stack)
  if n_elements(val) EQ 0 then return

  n2 = nvals + nstack
  if n_elements(stack) LT n2 then begin
     if n_elements(node0) EQ 0 then node0 = val(0)
     if n_elements(stack) EQ 0 then begin
        stack = replicate(node0(0), (n_elements(val)*2) > 10)
     endif else begin
        stack = [stack, replicate(node0(0),((n2-nstack)*5) > 10) ]
     endelse
  endif

  stack(nstack) = val
  nstack = n2
  return
end
```

```
;; Pop a node off of the node tree
;;
;;  STACK - (INPUT) original stack plus preallocated entries
;;          (OUTPUT) modified stack plus preallocated entries
;;          user need not modify or access this variable
;;  NVALS - number of values to be popped from stack
;;  NSTACK - (INPUT) number of values in original stack
;;           (OUTPUT) number of values in modified stack
;;           user need not modify this variable except to set to zero
;;           DEFAULT: n_elements(stack)
;;  CLIP - if set, then pre-allocated entries in STACK will be removed
;;  PEEK - if set, return values, but do not pop from stack
;;
function pds_pop, stack, nvals, nstack=nstack, clip=clip, peek=peek
  if n_elements(nstack) EQ 0 then begin
     nstack = n_elements(stack)
     clip = 1
  endif
  if n_elements(nvals) EQ 0 then nvals = 1L

  n2 = nvals < nstack
  if n2 LE 0 then goto, DO_CLIP

  if n2 EQ 1 then val = stack(nstack-1) $
  else            val = stack(nstack-n2:nstack-1)

  DO_CLIP:
  if keyword_set(peek) then goto, FINISH
  nstack = nstack-n2
  if keyword_set(clip) then begin
     if n_elements(stack) EQ n2 then dummy = temporary(stack) $
     else stack = stack(0:nstack-1)
  endif

  FINISH:
  if n_elements(val) GT 0 then return, val else return, 0L
end



--
 ------------------------------------------------------------ -------------
Craig B. Markwardt, Ph.D.        EMAIL:    craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
 ------------------------------------------------------------ -------------
```

Subject: Re: Push value onto an array?
Posted by Ed Wright on Sun, 19 Jan 2003 02:12:53 GMT
View Forum Message <> Reply to Message

in article b07o36$24do$1@nntp6.u.washington.edu, Rick Towler at rtowler@u.washington.edu wrote on 1/16/03 5:56 PM:

>
> "Ed Wright" wrote >
>> Question,
>>
>> How can I push a value onto an array?
>>
>> Given some array A of size m, how to add an entry at A[m]?
>>
>> Something like
>>
>> push @A, $value;
>>
>> This needs to be done in a loop. Yes, really.
>
> I think this was covered in depth a month or two ago. I just googled one
> entitled "How to use pointers instead of arrays" which covers some of the
> details (it quickly diverges from pointers and addresses your question
> directly).

Thanks,

I used Dr. Fanning's solution.

> In practice, this means that you have some kind of
> counter to tell you where you are in your array. If the
> counter gets above the "chunk" size, you allocate more
> memory to the array:
>
> array[counter] = value
> counter = counter + 1
> IF counter MOD 100 EQ 0 THEN array = [Temporary(array), Findgen(100)]
>
> When you finish adding things to your array, you trim
> it to the correct size:
>
> array = array[0:counter-1]
>
> This is both efficient and fast.

As always,

Ed Wright