

Re: Array and Pointer Tutorial

Source: http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2006-05/msg01930.html

- *From:* "Rod Pemberton" <do_not_have@xxxxxxxxxxx>
 - *Date:* Fri, 12 May 2006 17:54:50 -0400
-

"Richard Heathfield" <invalid@xxxxxxxxxxxxxxxx> wrote in message
[news:S_2dnfO7YaslYP7ZRVny0A@xxxxxxxxxx](mailto:S_2dnfO7YaslYP7ZRVny0A@xxxxxxxxxx)

Chad said:

I can't really put me pinpoint the exact part, but I know I'm missing some kind of underlying concept when I see the construction:

```
int *q = malloc(sizeof *q);
```

The canonical way to allocate space for n objects of type T is:

```
T *p = malloc(n * sizeof *p);
```

or, if p is already declared, simply this:

```
p = malloc(n * sizeof *p);
```

The reason this is the canonical way is that it doesn't rely on the type

of

p, except that it must be an object type, not an incomplete type or function type. If the type of p changes during maintenance, you don't have to hunt down this line and hack it about. It will automagically work correctly with the new type.

You should point out the negatives too. If p is already declared, and there is no comment telling him what file p is declared in. He'll spend forever trying to answer this question: "What the HELL is p?"

```
p = malloc(n * sizeof *p); /* p originally was of type T and declared in  
file somedecl.h */
```

Re: Array and Pointer Tutorial

Rod Pemberton

.