POINTER TO AN ARRAY IN OBJECTIVE-C

http://www.tutorialspoint.com/objective_c/objective_c_pointer_to_an_array.htm Copyright © tutorialspoint.com

It is most likely that you would not understand this chapter until you are through the chapter related to Pointers in Objective-C.

So assuming you have a bit understanding on pointers in Objective-C programming language, let us start: An array name is a constant pointer to the first element of the array. Therefore, in the declaration:

```
double balance[50];
```

balance is a pointer to &balance[0], which is the address of the first element of the array balance. Thus, the following program fragment assigns **p** the address of the first element of **balance**:

```
double *p;
double balance[10];

p = balance;
```

It is legal to use array names as constant pointers, and vice versa. Therefore, $*balance + 4$ is a legitimate way of accessing the data at balance[4].

Once you store the address of first element in p, you can access array elements using *p, $*p + 1$, $*p + 2$ and so on. Below is the example to show all the concepts discussed above:

```
#import <Foundation/Foundation.h>

int main ()
{
   /* an array with 5 elements */
   double balance[5] = {1000.0, 2.0, 3.4, 17.0, 50.0};
   double *p;
   int i;

   p = balance;

   /* output each array element's value */
   NSLog( @"Array values using pointer\n");
   for ( i = 0; i < 5; i++ )
   {
       NSLog(@"*(p + %d) : %f\n",  i, *(p + i) );
   }

   NSLog(@"Array values using balance as address\n");
   for ( i = 0; i < 5; i++ )
   {
       NSLog(@"*(balance + %d) : %f\n",  i, *(balance + i) );
   }

   return 0;
}
```

When the above code is compiled and executed, it produces the following result:

```
2013-09-14 01:36:57.995 demo[31469] Array values using pointer
2013-09-14 01:36:57.995 demo[31469] *(p + 0) : 1000.000000
2013-09-14 01:36:57.995 demo[31469] *(p + 1) : 2.000000
2013-09-14 01:36:57.995 demo[31469] *(p + 2) : 3.400000
2013-09-14 01:36:57.995 demo[31469] *(p + 3) : 17.000000
2013-09-14 01:36:57.995 demo[31469] *(p + 4) : 50.000000
2013-09-14 01:36:57.995 demo[31469] Array values using balance as address
2013-09-14 01:36:57.995 demo[31469] *(balance + 0) : 1000.000000
2013-09-14 01:36:57.995 demo[31469] *(balance + 1) : 2.000000
```

```
2013-09-14 01:36:57.995 demo[31469] *(balance + 2) : 3.400000
2013-09-14 01:36:57.995 demo[31469] *(balance + 3) : 17.000000
2013-09-14 01:36:57.995 demo[31469] *(balance + 4) : 50.000000
```

In the above example, p is a pointer to double, which means it can store address of a variable of double type. Once we have address in p, then ***p** will give us value available at the address stored in p, as we have shown in the above example.

Loading [MathJax]/jax/output/HTML-CSS/jax.js