600.120 Intermediate Programming, Spring 2017*

Misha Kazhdan

*Much of the code in these examples is not commented because it would otherwise not fit on the slides. This is bad coding practice in general and you should not follow my lead on this.
Why?

*ptr = *(ptr++);
In terms of precedence:

*ptr = *(ptr++);

"ptr++" increments the pointer but returns the old value, so the RHS side becomes

*(ptr++) → ra[0];
ptr → ra+1

This gives

ra[1]=ra[0];

```
#include <stdio.h>
int main( void )
{
    int ra[5] = { 1, 2, 3, 4, 5 };  
    int *ptr = ra;
    for( int i=0 ; i<5 ; i++ ) *ptr = *(ptr++);
    printf( "array is: [ ");
    for( int i=0 ; i<5 ; i++ ) printf( "%d ", ra[i] );
    printf( "]\n" );
    return 0;
}
```

>> ./a.out
array is: [ 1 1 1 1 1 ]
>>
Why?

In addition to giving the wrong result, this is bad code because in the last iteration we are setting
\[ ra[5] = ra[4]; \]
which is an out-of-bounds access!

RHS side becomes
\[ *(ptr++) \rightarrow ra[0]; \]
\[ ptr \rightarrow ra+1 \]
This gives
\[ ra[1]=ra[0]; \]

```c
#include <stdio.h>
int main( void )
{
    int ra[5] = { 1 , 2 , 3 , 4 , 5 };
    int *ptr = ra;
    for( int i=0 ; i<5 ; i++ ) *ptr = *ptr++;
    printf( "array is: [ " );
    for( int i=0 ; i<5 ; i++ ) printf( "%d ", ra[i] );
    printf( " ]\n" );
    return 0;
}
```

```
>> ./a.out
array is: [ 1 1 1 1 1 ]
>>
```
Piazza → Resources section → Resources tab → Exercise 5-1