

Today

C operators and their precedence

Midterm Review

Homework 2 Questions

Operator Preference in C (16 levels)

Operators

() [] -> . (postfix versions of ++ --)

(prefix versions of ++ --) sizeof

! ~ (unary versions of + - & *)

(type)

* / %

+ -

<< >>

< <= > >=

== !=

&

^

|

&&

||

?:

= += -= *= /= %= &= ^= != <<= >>=

,

Associativity

left to right 16

right to left 15

right to left 15

right to left 14

left to right 13

left to right 12

left to right 11

left to right 10

left to right 9

left to right 8

left to right 7

left to right 6

left to right 5

left to right 4

right to left 3

right to left 2

left to right 1

++ and --

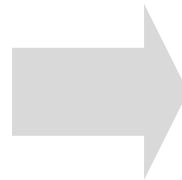
Unary increment(++)/decrement(--) operators

Prefix (to left, before): --x decrement first, then use

Postfix (to right, after): x++ use first, then increment

```
x = 3;
y = x++; // y gets 3, then x incremented to 4
z = --x; // x decremented to 3, then z gets 3
        // x, y, and z all are 3 at end
```

```
int j;
int ni = n*i;
double *rowp = a+ni;
for (j = 0; j < n; j++)
  {*rowp = b[j]; rowp++;}
```



```
int j;
int ni = n*i;
double *rowp = a+ni;
for (j = 0; j < n; j++)
  *rowp++ = b[j];
```

Precedence Examples

`a*b+c`

`a-b+c`

`sizeof(int)*p`

`*p->q`

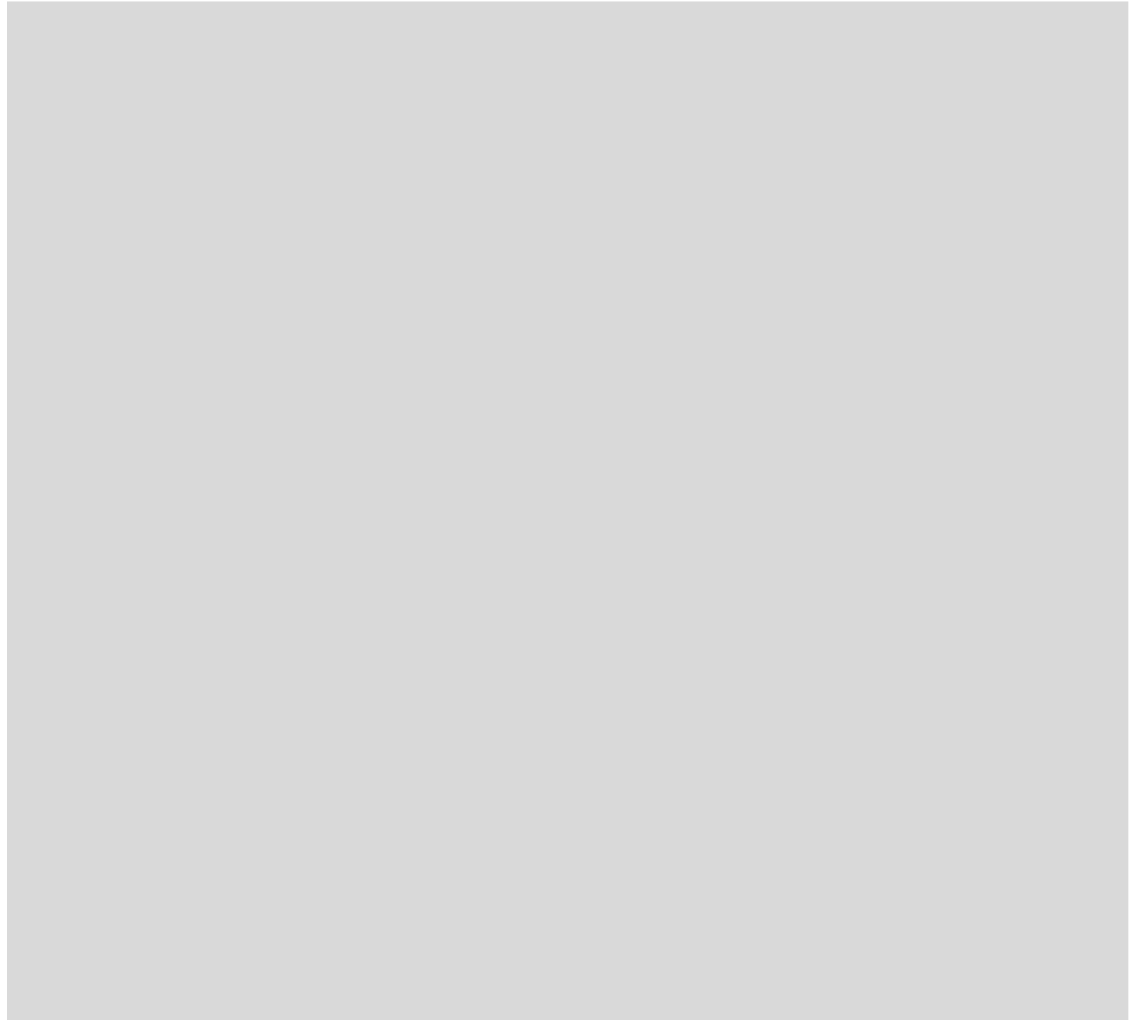
`*x++`

`a+=b++`

`a++b`

`a+++b`

`a++++b`



Precedence Examples

`a*b+c`

`(a*b)+c`

`a-b+c`

`(a-b)+c`

`sizeof(int)*p`

`(sizeof(int))*p`

`*p->q`

`*(p->q)`

`*x++`

`*(x++)` **not** `(*x)++` **incr after use**

`a+=b++`

`a+=(b++)` **incr after use**

`a++b`

`a+(+b)`

`a+++b`

`(a++)+b` **not** `a+(++b)` **incr after use**

`a++++b`

`(a++)+(+b)` **incr after use**

Midterm Review - Topics

- Memory + Data
- Numerical Representations
 - Integers, Floats, Doubles
- Introductory C
 - Operators, Pointers
 - C => Machine Code
- x86 Assembly
 - Basics, Conditionals, Iteratives
- Stack & Procedures
- Data Structures
 - Arrays (of var. dimension), structs, unions
- Buffer Overflows

Midterm Review - Assignments

- HW 0: Intro Performance
- Lab 1: Bitwise Operations / Basic C
- HW 1: Numerical Representations + x86 ISA
- Lab 2: x86 + Debugging
- HW 2: More x86

- HWs cover concepts
- Labs reinforce concepts + provide experience
- Content from both fair game!