CS 33

Introduction to C

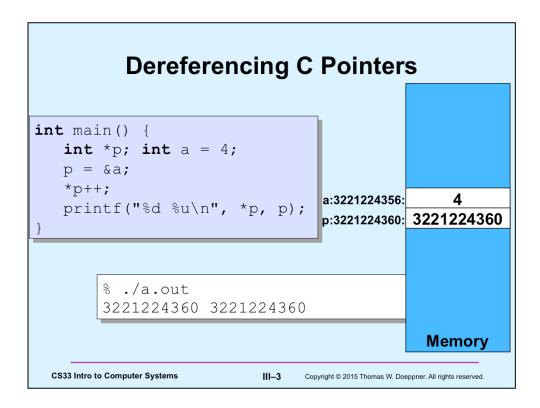
CS33 Intro to Computer Systems

III–1

Arrays and Parameters

CS33 Intro to Computer Systems

III-2



Operator precedence is hard to remember!

Dereferencing C Pointers

```
int main() {
   int *p; int a = 4;
   p = &a;
   (*p)++;
   printf("%d %u\n", *p, p);
}
```

```
% ./a.out
5 3221224356
```

CS33 Intro to Computer Systems

III–4

Dereferencing C Pointers

```
int main() {
   int *p; int a = 4;
   p = &a;
   ++*p;
   printf("%d %u\n", *p, p);
}
```

```
% ./a.out
5 3221224356
```

CS33 Intro to Computer Systems

III–5

Quiz 1

```
int proc(int arg[]) {
    arg++;
    return arg[1];
}
int main() {
    int A[3]={0, 1, 2};
    printf("%d\n",
        proc(A));
}
```

What's printed?

- a) 0
- b) 1
- c) 2
- d) indeterminate

CS33 Intro to Computer Systems

III–6

Strings

- Strings are arrays of characters terminated by '\0' ("null")
 - the '\0' is included at the end of string constants

»"Hello"



CS33 Intro to Computer Systems

III–7

Strings

```
int main() {
   printf("%s\n","Hello");
   return 0;
}
$ ./a.out
Hello
```

CS33 Intro to Computer Systems

III–8

Strings

```
void printString(char s[]) {
   int i;
   for(i=0; s[i]!='\0'; i++)
      printf("%c", s[i]);
}
int main() {
   printString("Hello");
   printf("\n");
   return 0;
}
```

Tells C that this function does not return a value

CS33 Intro to Computer Systems

III–9

- Suppose T is a datatype (such as int)
- T n[6]
 - declares n to be an array of (six) T
 - the type of n is T[6]
- Thus T[6] is effectively a datatype
- Thus we can have an array of T[6]
- T m[7][6]
 - m is an array of (seven) T[6]
 - m[i] is of type T[6]
 - -m[i][j] is of type T

CS33 Intro to Computer Systems

III-10

Copyright © 2015 Thomas W. Doeppner. All rights reserved.

Note that even though we might think of "int [6]" as being a datatype, to declare "n" to be of that type, we must write "int n[6]" — the identifier we are declaring goes in the middle of the name of the datatype.

- How do we declare an array of eight T[7][6]?
 - T p[8][7][6]
 - p is an array of (eight) T[7][6]
 - p[i] is of type T[7][6]
 - p[i][j] is of type T[6]
 - p[i][j][k] is of type T

CS33 Intro to Computer Systems

III–11

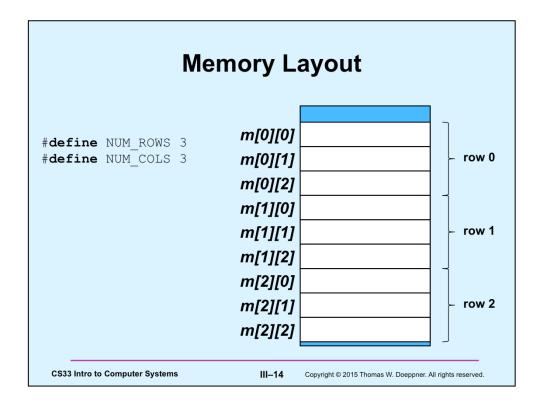
```
% ./a.out
                                  0
                                          1
                                                  2
                                                          3
 #define NUM ROWS 3
                                          5
                                                         7
 #define NUM COLS 4
                                  8
                                          9
                                                10
                                                        11
 int main() {
     int row, col;
     int m[NUM ROWS][NUM COLS];
     for(row=0; row<NUM ROWS; row++)</pre>
        for(col=0; col<NUM COLS; col++)</pre>
            m[row][col] = row*NUM COLS+col;
     printMatrix(NUM_ROWS, NUM_COLS, m);
     return 0;
CS33 Intro to Computer Systems
                           III–12
                                 Copyright © 2015 Thomas W. Doeppner. All rights reserved.
```

It must be told the dimensions

```
void printMatrix(int nr, "int nc,
        int m[nr][nc]) {
    int row, col;
    for(row=0; row<nr; row++) {
        for(col=0; col<nc; col++)
            printf("%6d", m[row][col]);
        printf("\n");
    }
}</pre>
```

CS33 Intro to Computer Systems

III–13



C arrays are stored in *row-major order*, as shown in the slide. The idea is that the left index references the row, the right index references the column. Thus C arrays are stored row-by-row. Thus to index into a 2D array, we need to know how large each row is (i.e., how many columns there are). But it's not necessary, for indexing purposes, to know how many rows there are.

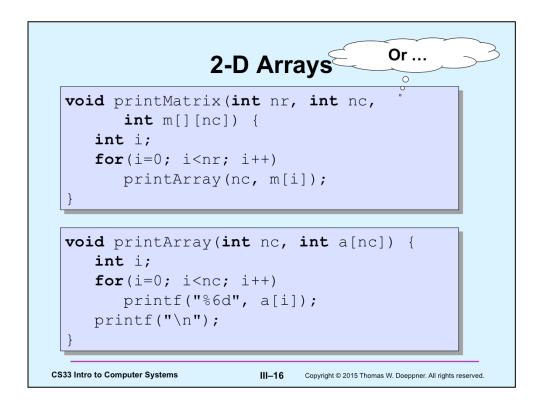
Alternatively ...

0

```
void printMatrix(int nr, int nc,
        int m[][nc]) {
   int row, col;
   for(row=0; row<nr; row++) {
        for(col=0; col<nc; col++)
            printf("%6d", m[row][col]);
        printf("\n");
}</pre>
```

CS33 Intro to Computer Systems

III–15



Note that m is an array of arrays (in particular, an array of 1-D arrays).

Parameters

Quiz 2

1) Consider the array

c) A[3][0]

d) none of the above

CS33 Intro to Computer Systems

III–18

Global Variables

The scope is global; m can be used by all functions

```
#define NUM_ROWS 3
#define NUM_COLS 4
int m[NUM_ROWS][NUM_COLS];

int main() {
   int row, col;
   for(row=0; row<NUM_ROWS; row++)
      for(col=0; col<NUM_COLS; col++)
        m[row][col] = row*NUM_COLS+col;
   return 0;
}</pre>
```

CS33 Intro to Computer Systems

III–19

#define NUM_ROWS 3 #define NUM_COLS 4 int m[NUM_ROWS][NUM_COLS]; int main() { int row, col; printf("%u\n", m); printf("%u\n", &row); return 0; } % ./a.out 8384 3221224352 CS33 Intro to Computer Systems | Copyright © 2015 Thomas W. Doeppner. All rights reserved.

Note that the reference to "m" gives the address of the array in memory.

Global Variables are Initialized!

```
#define NUM_ROWS 3
#define NUM_COLS 4
int m[NUM_ROWS][NUM_COLS];

int main() {
   printf("%d\n", m[0][0]);
   return 0;
}
```

```
% ./a.out
0
```

CS33 Intro to Computer Systems

III–21

Scope

```
int a; // global variable

int main() {
    int a; // local variable
    a = 0;
    proc();
    printf("a = %d\n", a); // what's printed?
    return 0;
}

int proc() {
    a = 1;
    return a;
}

CS33 Intro to Computer Systems

III-22 Copyright © 2015 Thomas W. Doeppner. All rights reserved.
```

Scope (continued)

```
int a;  // global variable

int main() {
    a = 0;
    proc(1);
    return 0;
}

int proc(int a) {
    printf("a = %d\n", a); // what's printed?
    return a;
}
```

CS33 Intro to Computer Systems

III-23

Scope (still continued)

```
int a; // global variable

int main() {
    a = 0;
    proc(1);
    return 0;
} $ gcc prog.c
prog.c:12:8: error: redefinition of 'a'
    int a;
    ^

int proc(int a) {
    int a;
    printf("a = %d\n", a); // what's printed?
    return a;
}

CS33 Intro to Computer Systems

III-24 Copyright © 2015 Thomas W. Doeppner. All rights reserved.
```

Syntax error ...

Scope (more ...)

```
int a; // global variable
int main() {
     // the brackets define a new scope
     int a;
     a = 6;
  printf("a = %d\n", a); // what's printed?
  return 0;
                         $ ./a.out
}
                         0
```

CS33 Intro to Computer Systems

Quiz 3

```
int a;
int proc(int b) {
    {int b=4;}
    a = b;
    return a+2;
}
int main() {
    {int a = proc(6);}
    printf("a = %d\n", a);
    return 0;
}
```

- What's printed?
 - a) 0
 - b) 4
 - c) 6
 - d) 8
 - e) nothing; there's a syntax error

CS33 Intro to Computer Systems

III–26