CS 161
Intro to CS I
Begin Multidimensional Arrays
Odds and Ends...

• Last week to demo Assignment #4
• Exam next Wednesday.
• Assignment #5 questions?
Multidimensional Arrays

• data_type array_name[rows][cols];
  – int array[2][3];
  – int array[4][2][3];
  – int array[2][4][2][3];

• What are examples of these?
  – 2-D – Matrices, Spreadsheet, Minesweeper, Battleship, etc.
  – 3-D – Multiple Spreadsheets, (x, y, z) system
  – 4-D – (x, y, z, time) system
Initializing 2-D Arrays

- **Declaration:** int array[2][3] = {{0,0,0},{0,0,0}};
- **Individual elements:**
  - array[0][0]=0; array[0][1]=0; array[0][2]=0;
  - array[1][0]=0; array[1][1]=0; array[1][2]=0;
- **Loop:**
  - for(i = 0; i < 2; i++)
    - for(j = 0; j < 3; j++)
      - array[i][j]=0;
- **Why do we need multiple brackets?**
Reading/Printing 2-D Arrays

• Reading Array Values
  for(i = 0; i < 2; i++)
    for(j = 0; j < 3; j++) {
      cout << “Enter a value for ” << i << “, ” << j << “: ”;
      cin >> array[i][j];
    }

• Printing Array Values
  for(i = 0; i < 2; i++)
    for(j = 0; j < 3; j++)
      cout << “Array: ” << array[i][j] << endl;
Static 2-D arrays...

Constant self-ref pointer

Row pointers

Constant self-ref cols in one array always contiguous

Pointer to row pointers

Fast
```cpp
#include <iostream>
using namespace std;

#define ROWS 3
#define COLS 3

int main() {
    int array[ROWS][COLS];

    for(int i=0; i<ROWS; i++)
        for(int j=0; j<COLS; j++)
            array[i][j]=10;

    cout << "Address of pointer to row pointers " << &array << endl;
    cout << "Contents of pointer to row pointers " << array << endl;
    cout << "Address of 1st row pointer " << &(array[0]) << endl;
    cout << "Contents of 1st row pointer " << array[0] << endl;
    cout << "Address of 1st element " << &(array[0][0]) << endl;
    cout << "Contents of 1st element " << array[0][0] << endl;
    cout << "Address of 2nd element " << &array[0][1] << endl;
    cout << "Address of 3rd element " << &array[0][2] << endl;
    cout << "Address of 1st element, 2nd row " << &(array[1][0]) << endl;
    cout << "Address of 1st element, 2nd row " << &array[0][3] << endl;

    return 0;
}
```
Passing a 2-D Array (Static)

```c
int main() {
    int array[5][5];
    ...
    pass_2darray(array);
    ...
}

void pass_2darray(int a[5][5]) {
    cout << "Array at zero: " << a[0][0] << endl;
}

OR

void pass_2darray(int a[][5]) {
    cout << "Array at zero: " << a[0][0] << endl;
}
```

Oregon State University
#include <iostream>
using namespace std;

#define ROWS 3
#define COLS 3

// have to have the stride, columns, for static 2-d array
void init_array(int array[][COLS]){
    for(int i=0; i<ROWS; i++)
        for(int j=0; j<COLS; j++)
            array[i][j]=10;
}

int main() {
    int array[ROWS][COLS];

    init_array(array);

cout << "address of pointer to row pointers " << &array << endl;
cout << "address of row pointers " << array << endl;
cout << "address of first row pointer " << &(array[0]) << endl;
cout << "address of first element " << array[0] << endl;
cout << "address of first element " << &(array[0][0]) << endl;
cout << "contents of first element " << array[0][0] << endl;
cout << "address of second element " << &(array[0][1]) << endl;
cout << "address of third element " << &(array[0][2]) << endl;
cout << "address of 1st element 2nd row " << &(array[1][0]) << endl;
cout << "address of 1st element 2nd row " << &(array[0][3]) << endl;
cout << "address of 1st element 2nd row in 2nd row pointer " << array[1] << endl;
cout << "address of 2nd row pointer is constant, self-ref " << &(array[1]) << endl;

    return 0;
}