CS 161
Intro to CS I

Continue 1-d Arrays, C-Strings, and Command-Line Arguments
Odds and Ends...

• No class Friday
• No demo or office hours Friday
• Questions???
```cpp
#include <iostream>

using namespace std;

void create_and_init(int **a, int num_ele) {
    *a=new int[num_ele]; //take me to p in main, what I point to
    for(int i=1; i<=num_ele; i++)
        //need to dereference a 1st then access array, i.e. go to p 1st,
        //then the array p points to
        (*a)[i-1]=i; //will this initialize elements to 1, 2, 3, ...
}

//just pass where the array is to access array
void print_array(int a[], int num_ele) {
    for(int i=0; i<num_ele; i++)
        cout << a[i] << endl;
}

int main() {
    //int &p=new int[10]; //can you make ref point to heap?, NO!!!
    //int *p=new int &; //can you make a ref on the heap?, NO!!!

    int *p=NULL; //make the pointer
    p=new int; //now set its contents to the location of int on heap
```
```cpp
//int &p=new int[10]; //can you make ref point to heap?, NO!!!
//int *p=new int &; //can you make a ref on the heap?, NO!!!

int *p=NULL; //make the pointer
p=new int; //now set its contents to the location of int on heap

int a[10];
//can't change a constant pointer, compiler will error
//a=p; //can you change where a points? who catches this?

//don't try to delete something off stack, compiler will give warning
//delete a; //can you delete mem off stack? who catches this?

//free the memory on heap before making p point to something new
delete p; //delete doesn't need [] because only one int on heap

int n;
cout << "enter the number of elements: ";
cin >> n;
//how do I call create_and_init()?
create_and_init(&p, n); //pass address of p to make p point to array
print_array(p, n); //pass address of array to access array
delete [] p; //delete needs [] because we used [] with new to make array

return 0;
```
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:**
  
  ```c
  int array[2][3] = {{0,0,0},{0,0,0}};
  ```

- **Individual elements:**
  
  ```c
  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:**
  
  ```c
  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**
  ```cpp
  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**
  ```cpp
  for(i = 0; i < 2; i++)
      for(j = 0; j < 3; j++)
          cout << "Array: " << array[i][j] << endl;
  ```
Static 2-D arrays…

```c
const int array[2][3];
```

Row-major