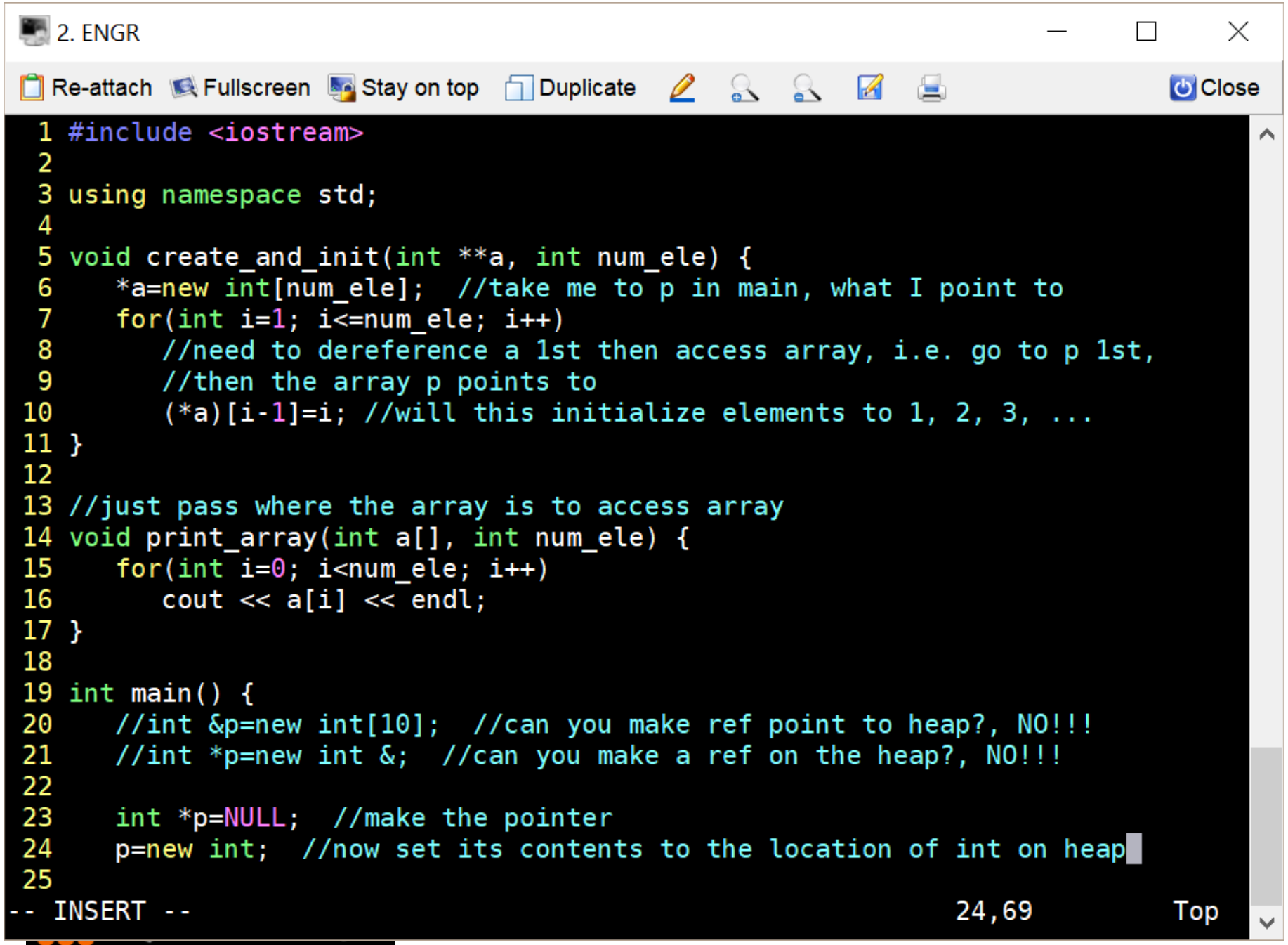
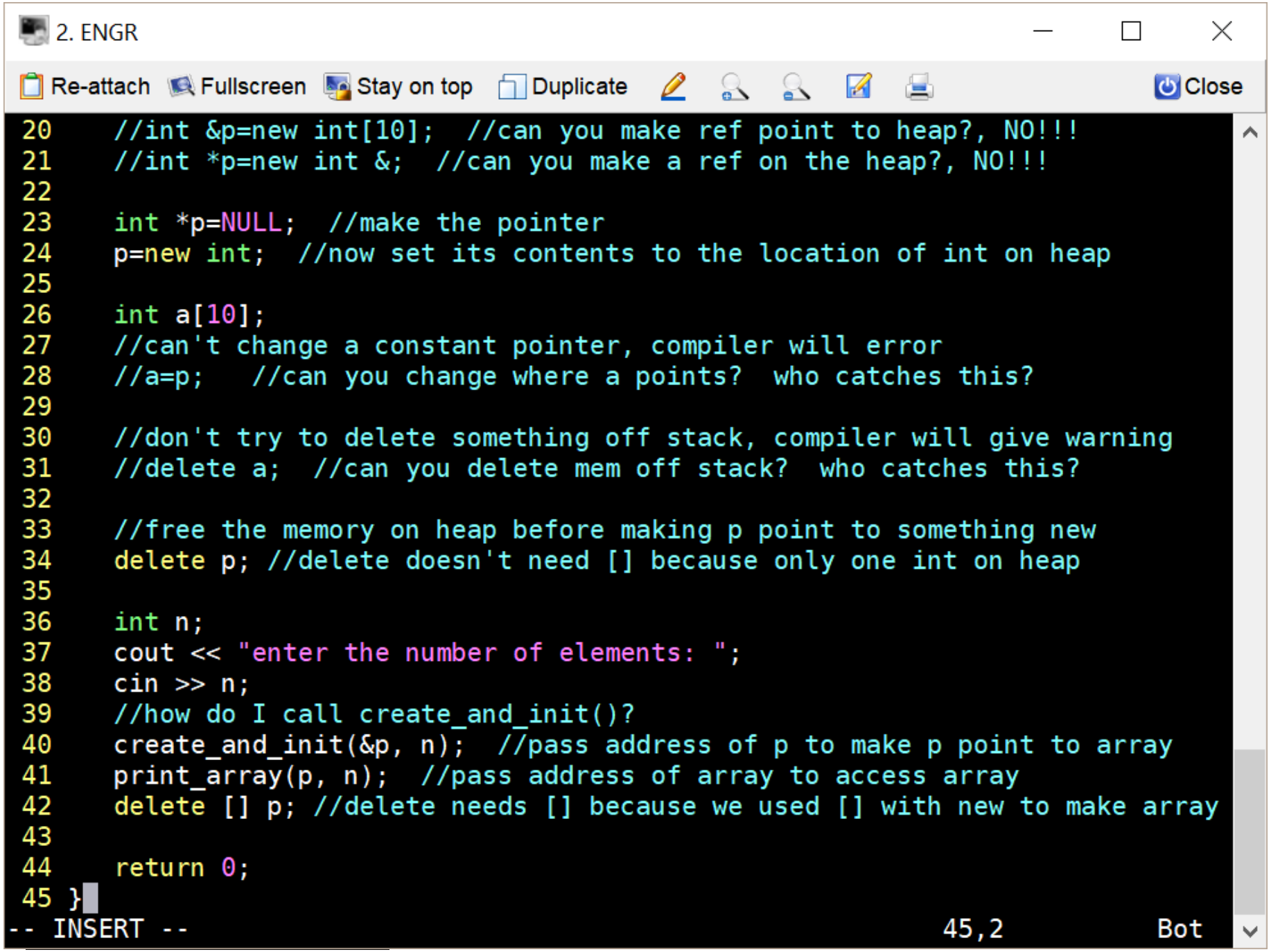
CS 161
Intro to CS I

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

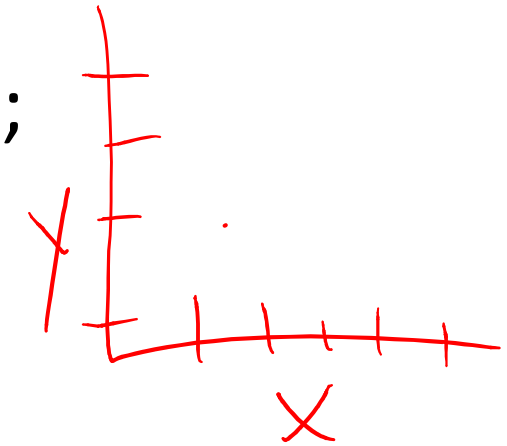
Odds and Ends…

- No class Friday
- No demo or office hours Friday
- Questions???


2. ENGR
Re-attach
Fullscreen
Stay on top
Duplicate
Close
1 #include <iostream>
2
3 using namespace std;
4
5 void create_and_init(int **a, int num_ele) {
6 *a=new int[num_ele]; //take me to p in main, what I point to
7 for(int i=1; i<=num_ele; i++)
8 //need to dereference a 1st then access array, i.e. go to p 1st,
9 //then the array p points to
10 (*a)[i-1]=i; //will this initialize elements to 1, 2, 3, ...
11 }
12
13 //just pass where the array is to access array
14 void print_array(int a[], int num_ele) {
15 for(int i=0; i<num_ele; i++)
16 cout << a[i] << endl;
17 }
18
19 int main() {
20 //int &p=new int[10]; //can you make ref point to heap?, NO!!!
21 //int *p=new int &; //can you make a ref on the heap?, NO!!!
22
23 int *p=NULL; //make the pointer
24 p=new int; //now set its contents to the location of int on heap
25
-- INSERT -- 24,69 Top


2. ENGR
Re-attach
Fullscreen
Stay on top
Duplicate
Close
20 //int &p=new int[10]; //can you make ref point to heap?, NO!!!
21 //int *p=new int &; //can you make a ref on the heap?, NO!!!
22
23 int *p=NULL; //make the pointer
24 p=new int; //now set its contents to the location of int on heap
25
26 int a[10];
27 //can't change a constant pointer, compiler will error
28 //a=p; //can you change where a points? who catches this?
29
30 //don't try to delete something off stack, compiler will give warning
31 //delete a; //can you delete mem off stack? who catches this?
32
33 //free the memory on heap before making p point to something new
34 delete p; //delete doesn't need [] because only one int on heap
35
36 int n;
37 cout << "enter the number of elements: ";
38 cin >> n;
39 //how do I call create_and_init()?
40 create_and_init(&p, n); //pass address of p to make p point to array
41 print_array(p, n); //pass address of array to access array
42 delete [] p; //delete needs [] because we used [] with new to make array
43
44 return 0;
45 }
-- INSERT --
45,2
Bot

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…