CS 161
Intro to CS I

Pointers vs. References Exercise

and Intro to Recursion

Odds and Ends…

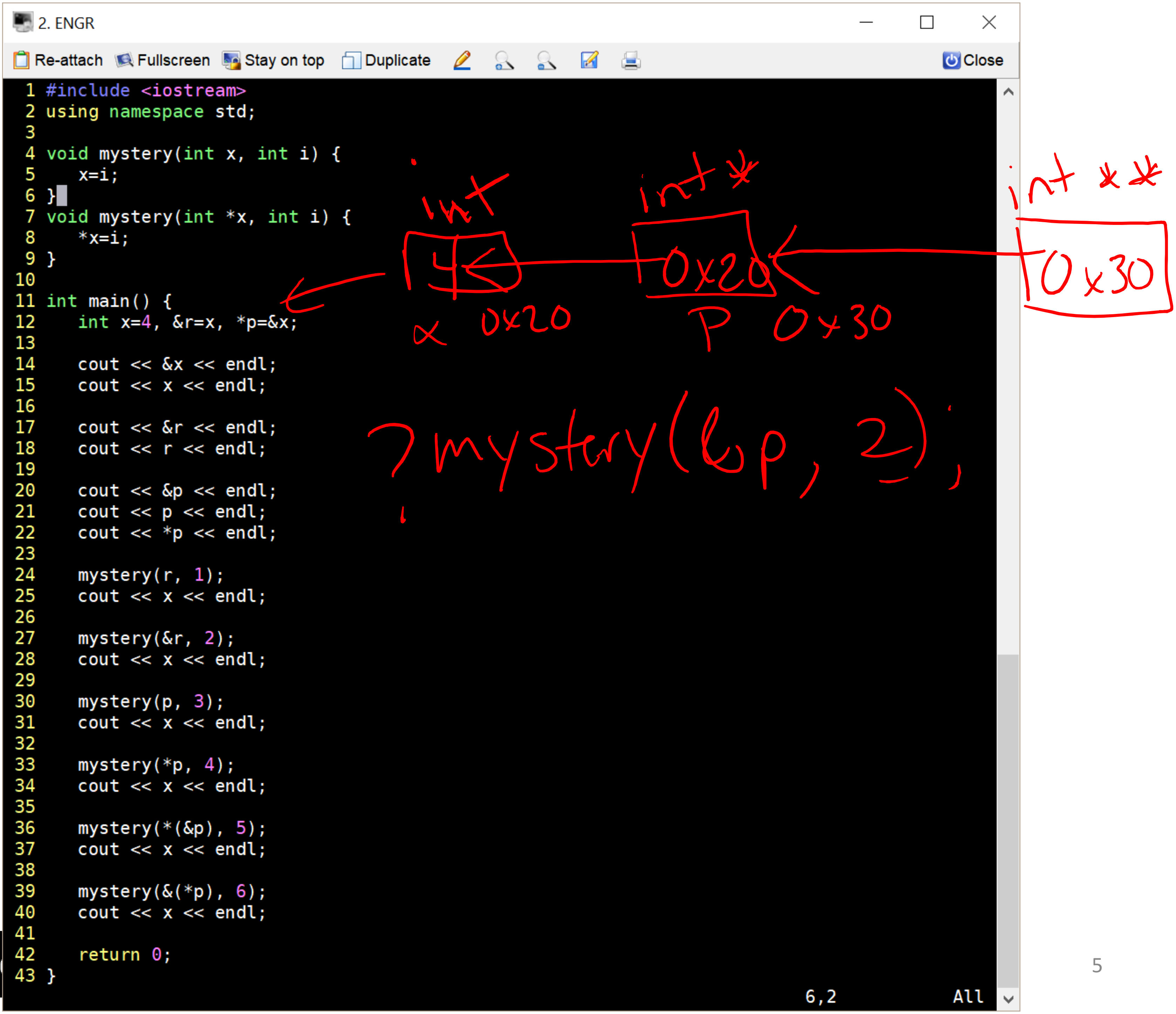
- Assignment #4 (Little Acorns)
- Design due Sunday night on Canvas.
  - Make sure you include postconditions and preconditions for functions.

More About Functions

- Do not use global variables!
- Function Headers
  - Description, Parameters, and Return Value
  - Preconditions
    - What is this?
  - Postconditions (look at Recitation Worksheet!)
    - What is this?

Pointer and References Cheat Sheet

- *
  - If used **in a declaration** (which includes function parameters), it **creates** the pointer.
    - Ex. int *p; //p will hold an address to where an int is stored
  - If used **outside a declaration**, it **dereferences** the pointer
    - Ex. *p = 3; //**goes to the address** stored in p and stores a value
    - Ex. cout << *p; //**goes to the address** stored in p and fetches the value
- &
  - If used **in a declaration** (which includes function parameters), it **creates and initializes** the reference.
    - Ex. void fun(int &p); //p will refer to an argument that is an int by implicitly using *p (dereference) for p
    - Ex. int &p=a; //p will refer to an int, a, by implicitly using *p for p
  - If used **outside a declaration**, it means **“address of”**
    - Ex. p=&a; //**fetches the address of** a (only used as rvalue!!!) and store the address in p.



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

void mystery(int x, int i) {
   x=i;
}
void mystery(int *x, int i) {
   *x=i;
}

int main() {
   int x=4, &r=x, *p=&x;

   cout << &x << endl;
   cout << x << endl;

   cout << &r << endl;
   cout << r << endl;

   cout << &p << endl;
   cout << p << endl;
   cout << *p << endl;

   mystery(r, 1);
   cout << x << endl;

   mystery(&r, 2);
   cout << x << endl;

   mystery(p, 3);
   cout << x << endl;

   mystery(*p, 4);
   cout << x << endl;

   mystery(*(&p), 5);
   cout << x << endl;

   mystery(&(*p), 6);
   cout << x << endl;

   return 0;
}
                                                          6,2           All
```

? mystery(&p, 2);

In-class Exercise
Pointers vs. References

- What if you made a pointer (p2) that points to a pointer (p) to an int (x)?
  - What would the picture look like?
  - Write the code for this picture.

- Can you make this same picture for references?
  - What if you had two references, r and r2?

Recursion

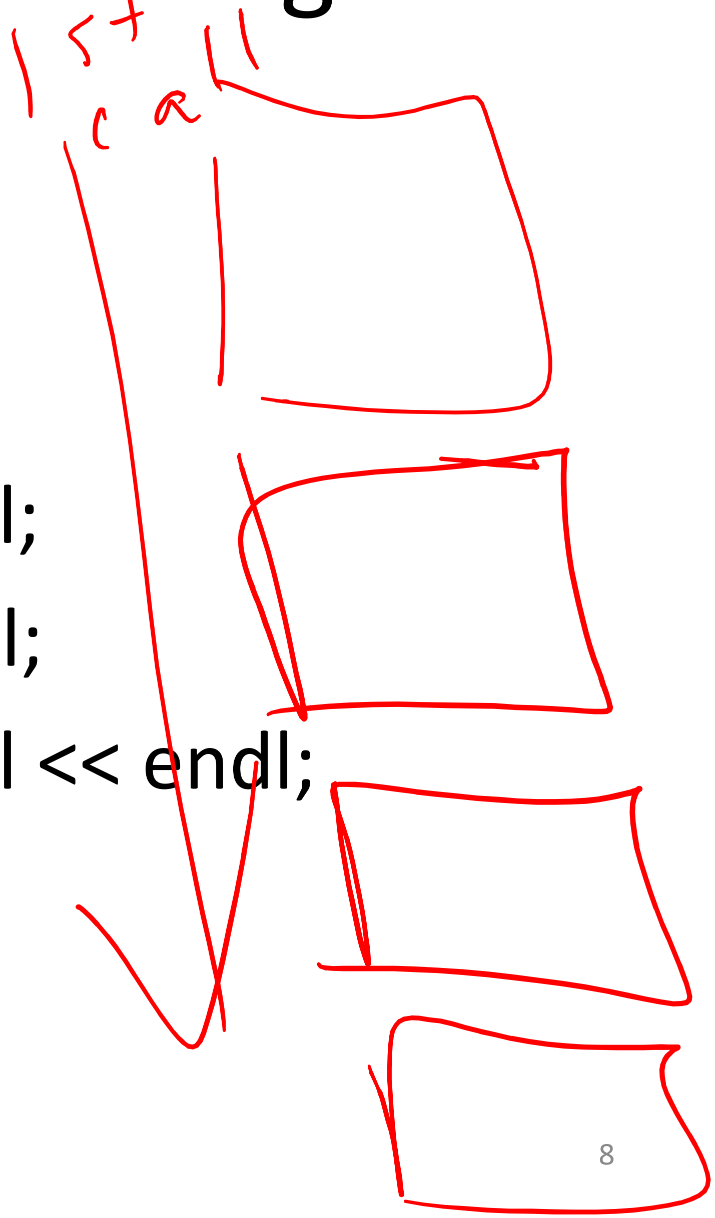
- What is it?
  - Function that calls itself 1 or more times (directly or indirectly)
  - Has 1 or more base case for stopping
  - Inductive reasoning: general case must eventually be reduced to a base case

Example: Drawing Rectangles

- Iterative Solution:

4

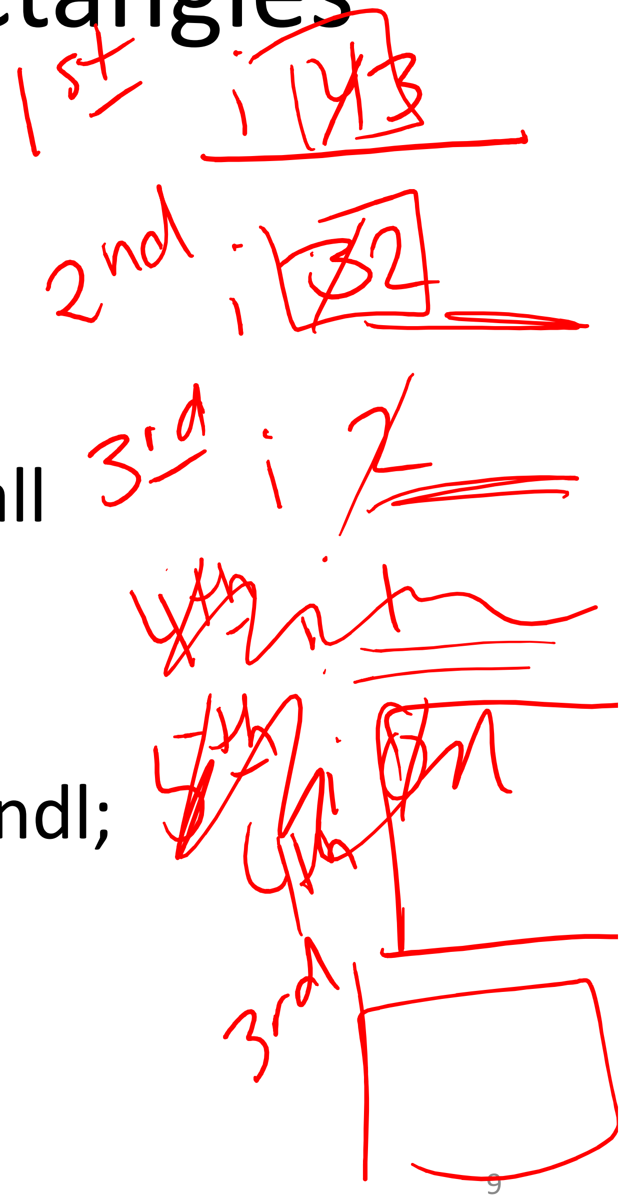
```
void draw_rect(int i) {
    for( ; i > 0; i--){
      cout << "******“ << endl;
      cout << "*    *“ << endl;
      cout << "******“ << endl << endl;
    }
}
```



Example: Drawing Rectangles

- Recursive Solution 4

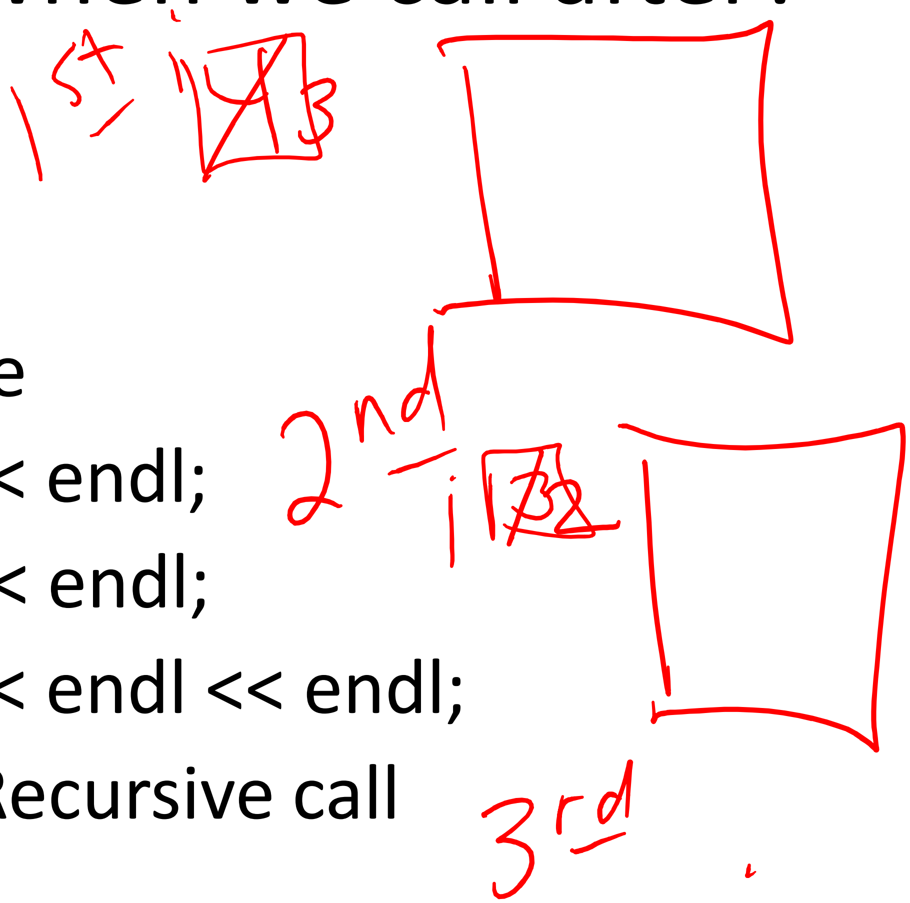
```
void draw_rect(int i) {
        if(i>0){        //Base case
          draw_rect(--i);    //Recursive call
          cout << "******“ << endl;
          cout << "*           *“ << endl;
          cout << "******“ << endl << endl;
        }
  }
```



What is different when we call after?

- Recursive Solution

```
void draw_rect(int i) {
        if(i>0){        //Base case
          cout << "*******“ << endl;
          cout << "*           *“ << endl;
          cout << "*******“ << endl << endl;
          draw_rect(--i);    //Recursive call
        }
 }
```


1st
i ~~4~~ 3
2nd
i ~~3~~ 2
3rd