Last week to get demoed
Assign #3 to do critiques by end of week.

CS 161
Intro to CS I

Pointers and Static vs. Dynamic Memory
Iterative Factorial

factorial(0) = 1;
factorial(n) = n*n-1*n-2*…*n-(n-1)*1;

long factorial(int n) {
    long fact;
    if(n==0)
        fact=1;
    else
        for(fact=n; n > 1; n--)
            fact=fact*(n-1);
    return fact;
}
Recursive Factorial

factorial(0) = 1;
factorial(n) = n * factorial(n - 1);

long factorial(int n) {
    if (n == 0) // Base case
        return 1;
    else
        return n * factorial(n - 1); // Recursive call
In-class Exercise

• Get into groups of 4 – 5.
• Write your own recursive int pwr() function that takes two integers as arguments and returns the integer result.
  – What does the function prototype look like?
  – Now, write the function definition...
Recursion Demo...

```cpp
#include <iostream>

using namespace std;

int pwr_i(int b, int e) {
    int result=1;
    for(int i=1; i<=e; i++)
        result=result*b; // same as result*=b;
    return result;
}

int pwr(int b, int e) {
    if(e==0) // better have a base case to stop!!
        return 1;
    else
        return b*pwr(b, e-1);
}

int main() {
    cout << pwr_i(2, 0) << endl;
    cout << pwr(2, 0) << endl;
    return 0;
}
```

-- INSERT --
Variables vs. Pointers

- **Value Semantics**
  - Values stored directly
  - Copy of value is passed
    ```c
    int i, j=2;
    i=j;
    ```

- **Pointer Semantics**
  - Address to variable is stored
  - Copy of address is passed
    ```c
    int *i, j=2;
    i=&j;
    ```
What if we don’t have the j?

• We need to create the address space.
• How do we do this?
  – new type;
• For example:
  int *i = NULL;
  i = new int; //new returns an address
  *i = 10;
• [http://cslibrary.stanford.edu/104/]
Stack vs. Heap

- Static vs. Dynamic
Static vs. Dynamic

- **Static Semantics**
  - Assign address of variable
    int *i, j=2;
    i=&j;

- **Dynamic Semantics**
  - Create memory
    int *i=NULL;
    i=new int;
  - Assign memory to pointer
    *i=2;