modularity using functions
function

input
arguments

func

output
return
function in c

calling a function: func(arguments)
defining a function:
ty func(arguments){
    //body of func
    return var; //var has type ty
}
calling a function 2: ty x = func(arguments)
x will contain var
int main()
{
    cout << "hello world";
    return 0;
}
findmax - prototype

- returns an integer
- takes two integers as inputs

```c
int findmax(int, int);
```
```c
int findmax(int x, int y){
    int maxnum;
    if (x >= y )
        maxnum = x;
    else
        maxnum = y;
    return maxnum;
}
```
findmax – multiple returns

```c
int findmax(int x, int y){
    if (x >= y )
        return x;
    return y;
}
```
findmax-return expressions

int findmax(int x, int y){
    if (x < y )
        return x + (y-x);
    return x;
}

using findmax

```cpp
int a = 5;
int b = 3;
int max_ab;
max_ab = findmax(a, b);
cout << "Max of " << a << " and " << b;
cout << "is " << max_ab;
cout << " or " << findmax(a, b);
```
int findmax(int, int = 0);

compares 5 to 3
  j = findmax(5, 3)

compares 5 to 0 (the default)
  j = findmax(5)

must be listed after required values
overloading

double findmax(double, double)
double findmax(double x, double y){
    if (x > y)
        return x;
    return y;
}

templates

handles multiple types without overloading

```cpp
template <class T>
T findmax(T x, T y) {
    if (x > y)
        return x;
    return y;
}
```
multiple returns – pass by reference

```cpp
void findmax(int, int, int&)

void findmax(int x, int y, int& max) {
    if (x > y) {
        max = x;
    } else {
        max = y;
    }
}
```

& means the address (box containing) max
int a = 5;
int b = 3;
int max;
findmax(a, b, max);
cout << "Max of " << a << " and " << b;
cout << "is " << max;
Variable Scope

- **local scope**
  - only seen by function it's defined in
  - name can be reused in other functions

- **global scope**
  - entire program can see the variable;
  - defined outside main;
  - bad/don't use unless sure you need it.
scope example

int add3(int x, int y, int z){
    return add2 x +add2(y, z);
}

int add2(int x, int y){
    return x + y;
}

category type varname;

- **auto** (default in c/c++)
  - storage allocated each time declaration statement is seen

- **static**
  - storage allocated only when program first compiled so declaration only executes once

- **extern**
  - extends scope to multiple files
  - uses variables existing storage (no allocation)
Quiz

implement findminmax, which given a and b returns min(a,b) and max(a,b)

hint: findminmax is pass by reference and takes in 4 input arguments
Homework

• Chapter 6 problems:
  • 5-10 (pg. 367)