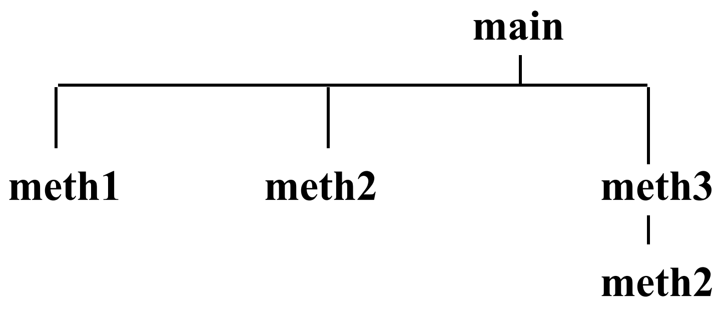
Methods (a.k.a functions)

- organize programs into sections
- allow sections of one program to be re-used in another program
- enhance the readability of a program

methods can be called from anywhere

```
public static void main(String
   args[] )
{
   meth1();
   meth2();
   meth3();
}
private static void meth1()
{
   System.out.println(“in meth1”);
}
```

```
private static void meth2()
{
   System.out.println(“in
   meth2”);
}
private static void meth3()
{
  System.out.println(“in
   meth3”);
  meth2(); // call meth2
}
```



Call Graph of program

Methods can take parameters

- Parameters allow you to pass data to the function
- Parameters must have a name & data type
- What are the parameters and data types of this function?
- Why can’t I print **message** from inside the function body?

```
public static void main(String args[] )
{
   String message = “Hello World”;
   printMsg( message );       // Go DO the method CALL
}

private static void printMsg( String msg ) // method DEFINITION
{
   System.out.println( msg );
}
```

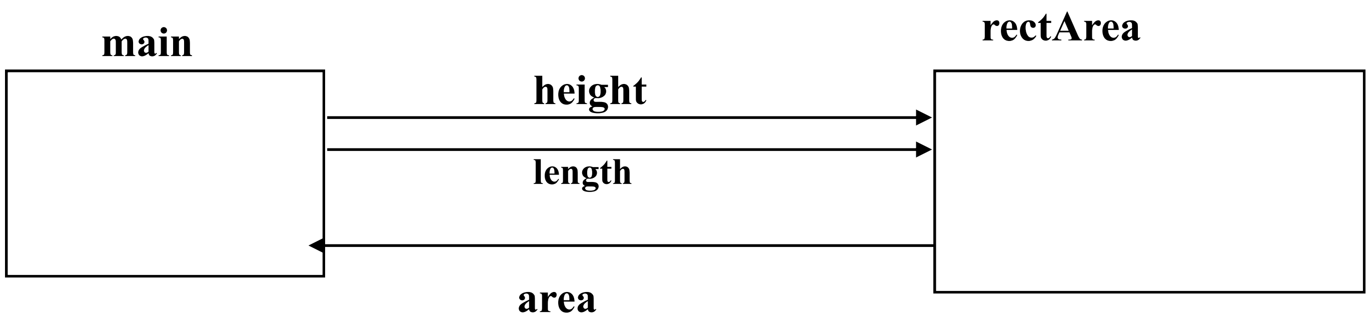
parameters are NOT modified by the method
You are just sending in a COPY of the value

```
public static void main( String args[] )
{
  int x = 3, y = 10;
  add2(x); // prints 5
  System.out.println( “x: “ + x + “,  y: “ + y ); // x: 3  y: 10
  add2(y); // prints 12
  System.out.println( “x: “ + x + “,  y: “ + y ); // x: 3  y: 10
}

private static void add2(int p)
{
  p += 2;
  System.out.println( p );
}
```

Return statement - sends a value back from a method

- *return* statement
  - transfer a value back from a function
  - you can only send back ONE piece of information
  - can only be used with functions that are non-void
  - e.g. private static float distance(int x1, int y1, int x2, int y2);
  - private static int taxcode(float salary);



Example of parameters and return stmt

```
public static void main( String args[] )
{
   int lo=1, hi=100;
   int rangeSum = calcRangeSum( lo, hi );
   System.out.println("sum of 1 thru 100= " + rangeSum );

} // END main

private static int calcRangeSum( int lo, int hi)
{
   int sum=0; // local variable - lives & dies in this method

   for (int i=lo ; i<=hi ; ++i)
      sum+=i;

   return sum; // sum of all the numbers from lo to hi
} // END calcRangeSum
```

Embedded function calls

```
public static void main()
{
   int a = 5, b = 8, g;
   g = meth5(meth6(a), meth5(a, b));
   System.out.println( “g is “ + g );
}

private static int meth5(int v, int w)
{
   return v + w;
}
private static int meth6(int p)
{
   return p - 2;
}
```

Scope

- **You can have variables with the same name in different methods.**

```
public static void main()
{
   int a = 4, b = 6, c;
   int a = 15, c = 9;  // ILLEGAL can’t re-declare in same scope
}

private static void meth1(int x, int y){
   int a = 7, b = 5; // OK: different block i.e. {}
   ...
}

private static void meth2(int a, int b) // OK: different a & b
{
   int c; // OK: different c than main’s
}
```

Scope

- **Actual and Formal parameter names do NOT have to match**
  - **the values map by POSITION**

```
public static void main()
{
   int g = 50, h = 90, k;

   k = meth3(g, h);
   System.out.println( k );
   k = meth3(h, g);
   System.out.println( k );
   k = meth3(h, h);
   System.out.println( k );
}

private static int meth3(int g, int h)
{
   return g - h;
}
```