What are member functions, really?

Pointers to structs and accessing data members a.k.a. fields.

CSI 310: Lecture 11
Selection using field name:

```c
STL *ps = new STL;
```

Dynamic allocation by:

```c
STL g;
```

Automatic storage allocation by

```c
{ struct STL int ddp; STL *tink;
  } struct definition;
```

Selected by field (member) name ddp

```
allocated near each other,
```

nonuniform types defined by YOU

```
Collection of variables of
collection (class type object) G?
```

What's a structure (class type object) G?

[[?]
```
assert (? > 2003);
```

[[?]
```
assert (0 <= 2003);
```

Selection using array notation:

```c
ettype *pa = new ettype[?];
```

Dynamic allocation by:

```c
e ettype A[2003];
```

Automatic storage allocation by

Selected by subscript value i

allocated contiguously

uniform type

Collection of variables of

What's an array A?
struct STT
{
    int ddF;
    STT* link;
};

STT G;

STT* pS;
pS = new STT;

STT c, STT* pS;

STT* Tink;

set c = new eleType[5];

set A = new eleType[3];

set pA = new eleType[3];

ddF = new eleType[3];

pA = new eleType[5];
Calling the "Fake function member":
{
  count>>pd<<dtt; //ERROR:dtt not in scope
count>>print the same thing!
  count>>pd<<dtt;
  count>>(*pd<<dtt);
}

What does calling a function member like fun really mean??

... fun init ddr;
Suppose struct stt { void fun ( ) { } } stt d;
stt *ps; ps = new stt;

(Also, it is in the class's scope and can access private members.)

called, (also, it is in the class's scope and can access private members).

Value parameter (which you can access via the keyword within
the body) holding the address of the object for which it was
called (also, it is in the class's scope and can access private members).

A fun. member is like an ordinary fun. except it has AN EXTRAA (IMPLICIT)
first.

you must 'break main' or equivel and run
by setting a breakpoint on it.

You can also debug a function
debugger out of it with the finish command.

Tips: (1) If you step into a MESS (library function, etc.)
you can pull the

FUNCTION WAS CALLED.

WILL PRINT THE OBJECT FOR WHICH THE
PRINT * THIS

ACTIVATION:

WHEN DEBUGGING INSIDE A C++ FUNCTION

Command step steps into functions when they are called, too.

Function calls are performed and return without stopping.

next command performs line-by-line controlled execution of the current

'this' is handy for debugging.
slope's, so try anything you're curious about.

Computer Programming, unlike skiing, etc., is unlikely to break legs, etc., on too difficult

4. Overloading prefix ++ operator with a member function (+X returns X,'.

    ) (Coded by C++ Library designers, not
    cin, etc.) They were called for. (Coded by C++ Library designers, not
    these member functions return a reference to the ostream object (cont,'.
    ostream classes (instead of just friends of data type to output or input)::
    ostream >> operator>(>> operator>> (>> operator>>)
    and (>> operator>> (>> operator>> (>> operator>>

3. Defining of ostream ostream::operator<<(ostream::operator <<
   >>) (>> operator<< (>> operator<< (>> operator<<
   and (>> operator<< (>> operator<< (>> operator<<

RECOMMENDED FOR BEGINNERS" (i.e., this course),
advanced object-oriented styles of data structure programming; NOT
advanced object-oriented styles of data structure programming; NOT

2. Some member functions will store the address of the current object into

   assignment operators: See page 174 of DSO, for details.

1. Detection of self-assignment for proper implementation of overloaded

Some uses of C++ "this" pointer value in programming:

University at Albany Computer Science Dept.