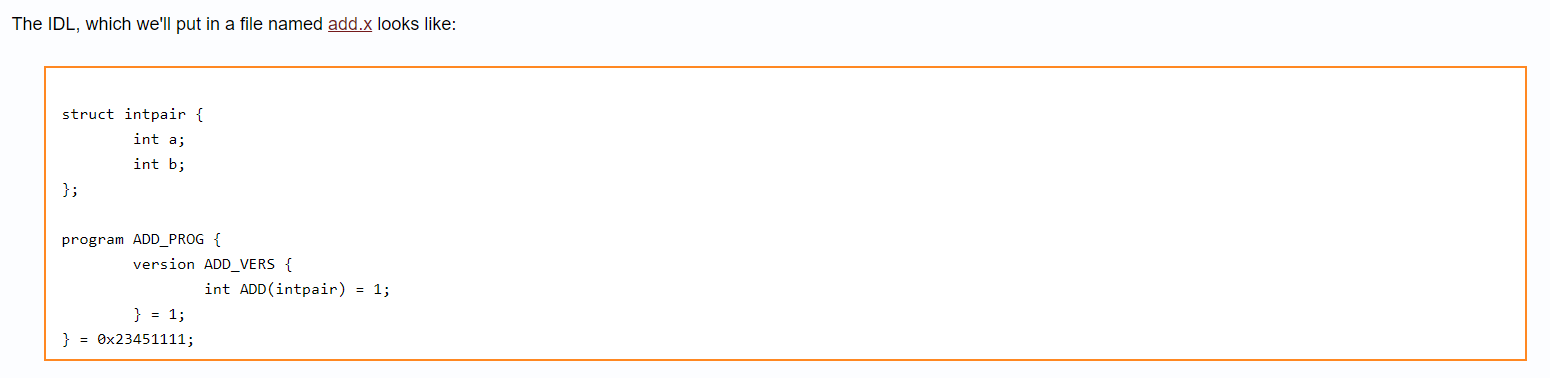
**Interface Definition Language**

Sun has a format for its IDL, an IDL is a file which optionally begins with a bunch of type definitions and then defines the remote procedures. A set of remote procedures are grouped into a version, one or more versions are grouped into a program.

In this example we have one type definition to define a structure that holds two integers: this will be our input parameter for the *add* function. Our interface will also have one version and one program. Programs are stored in memory between 0 to 0x1fffffff.

  
  
  
  
We can compile:  
  
rpcgen –C add.x

This will generate files:

1. Add.h
2. Add\_svc.c
3. Add\_clnt.c
4. Add\_xdr.c

We can generate each file bit by bit by calling other commands; but instead, we can use this to generate everything.  
  
rpcgen -a -C add.x

This will create all the files explained above, plus the client, the server and the makefile.   
  
  
  
We change these in the makefile:

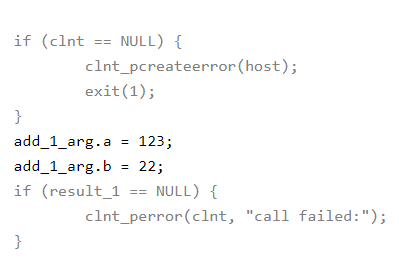
CFLAGS += -g to CFLAGS += -g DRPC\_SVC\_FG

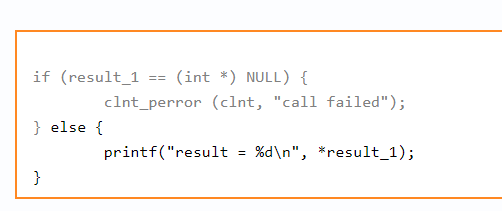
RPGCENFLAGS = to RPCGENFLAGS = -C  
  
We then call:

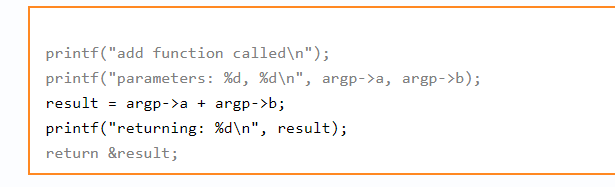
1. Make –f Makefile.add
2. ./add\_server [to run server]
3. ./add\_client [to run client once]

In the server window, you should see the following text appear:

“add function called”  
  
**Getting the server to do some work**add\_client:

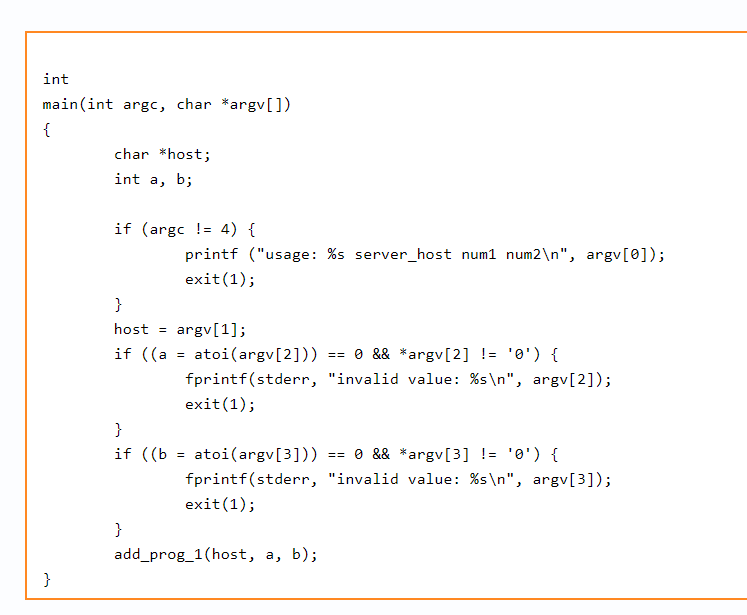
*add\_prog\_1 [function definition]*

  
  
  
  
add\_server:



Note that the variable result is declared static. This is crucial because local (automatic) variables live on the stack. As soon as the server function returns the pointer to the result back to the server stub, the memory used by local variables can be reclaimed for use by the server stub. Failure to declare the return type static can result in nasty bugs where the code may seem to work a lot of the time but not always.

**Final Changes**Client:

  
  
  
  
Function call:  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
