# COIS 2300H Lab 4 Recursion

Now we’re going to make a recursive procedure (a procedure that calls itself). This is conceptually very similar to the previous example except the program can have a fairly large stack.

Let’s say we have 3 programs. A calls B, B calls C, C calls D.

A{ B()}; B{C()}; C{D()}; D{something}. The way this actually works is D finishes first, gets popped off the execution stack then C gets the result of D, finishes what it was doing, gets popped off, B gets the result of C, finishes, pops off the stack, then A gets the result of B and finishes.

Recursion is essentially the same, except it’s the same function, just with different parameters at each step. Well, hopefully. If you keep giving it the same parameters, you’ll get an infinite loop.

We will make a simple recursive function, given two integers (s and f) we will have a recursive function that adds all of the integers from s to f

In C

int mySum (int s, int f){

if (s==f)

Return f;

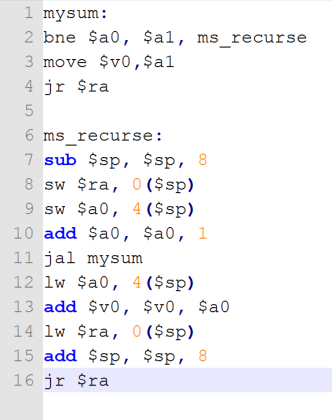
else

Return s+mySum(s+1, f);

}

(note that the else is actually redundant, but I put it there for readability)  
  
the MIPS conversion of that is

You need to make this work (notice that $a0 and $a1 are never set)



## Lab Completion for Today

1. Make a program which reads in two integers (s and f), and then uses a recursive procedure to add up the values from s to f.

(So in C type language that would be

int addnumbers(int smaller, int larger) {

if (smaller == larger)

return smaller;

else

return larger + addnumbers(smaller, larger - 1);

}