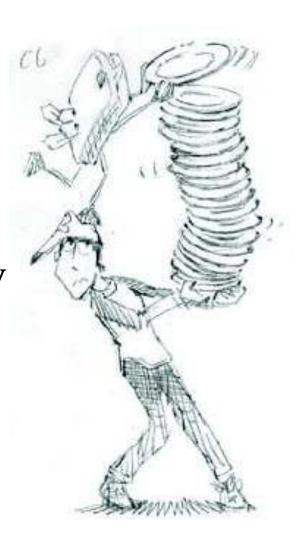
# The Stack

- Memory used to temporarily save variables
- Like stack of dishes, last-in-first-out (LIFO) queue
- *Expands*: uses more memory when more space needed
- *Contracts*: uses less memory when the space is no longer needed



### The Stack

- Grows down (from higher to lower memory addresses)
- Stack pointer: \$sp points to top of the stack

Address	Data		Address	Data	
7FFFFFC	12345678	<b>←</b> \$sp	7FFFFFC	12345678	_
7FFFFF8	12010070	-	7FFFFF8	AABBCCDD	_
7FFFFFF4			7FFFFFF4	11223344	<b>←</b> \$sp
7FFFFF0			7FFFFF0		
•	•		•	•	
•	•		•	•	

#### How Functions use the Stack

- Called functions must have no unintended side effects
- But diffofsums overwrites 3 registers: \$t0, \$t1, \$s0

```
# MIPS assembly
# $s0 = result
diffofsums:
   add $t0, $a0, $a1  # $t0 = f + g
   add $t1, $a2, $a3  # $t1 = h + i
   sub $s0, $t0, $t1  # result = (f + g) - (h + i)
   add $v0, $s0, $0  # put return value in $v0
   jr $ra  # return to caller
```

### Storing Register Values on the Stack

```
# $s0 = result
diffofsums:
 addi $sp, $sp, -12 # make space on stack
                    # to store 3 registers
 sw $s0, 8($sp) # save $s0 on stack
 sw $t0, 4($sp) # save $t0 on stack
 sw $t1, 0($sp) # save $t1 on stack
 add $t0, $a0, $a1 # $t0 = f + g
     $t1, $a2, $a3 # $t1 = h + i
 add
 sub $s0, $t0, $t1 # result = (f + g) - (h + i)
 add $v0, $s0, $0 # put return value in $v0
 lw $t1, 0($sp) # restore $t1 from stack
 lw $t0, 4($sp) # restore $t0 from stack
 lw $s0, 8($sp) # restore $s0 from stack
 addi $sp, $sp, 12  # deallocate stack space
 jr $ra
                    # return to caller
```

## The stack during diffofsums Call

