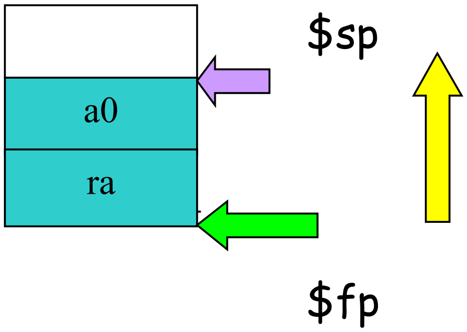
Handling recursive procedure calls

Example. Compute factorial (n)

```
int fact (int n)

{

        if (n < 1) return (1);

                else return (n * fact(n-1))

}
```

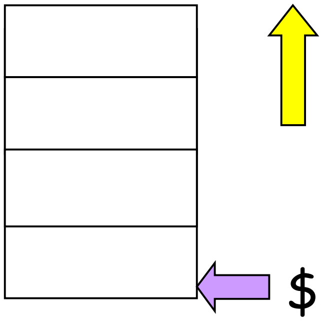
(Plan) Put n in $a0. Result should be available in $v0.

```
fact:       subi $sp, $sp, 8

            sw   $ra, 4($sp)

            sw   $a0, 0($sp)
```

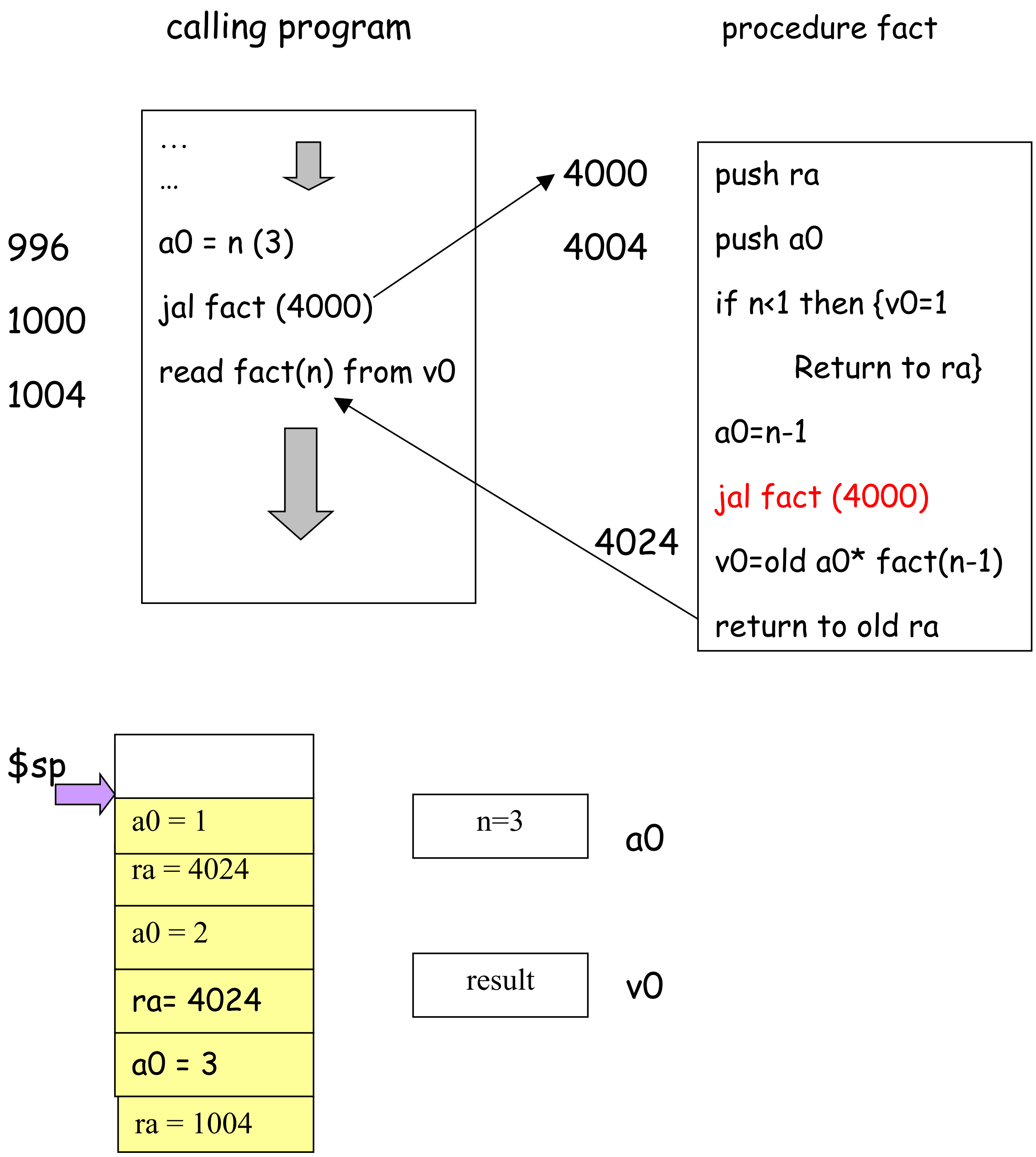


NEW

OLD



$sp (current top of the stack)



calling program

```
        ...
        ...
996     a0 = n (3)
1000    jal fact (4000)
1004    read fact(n) from v0
```

procedure fact

```
4000    push ra
4004    push a0
        if n<1 then {v0=1
                Return to ra}
        a0=n-1
        jal fact (4000)
4024    v0=old a0* fact(n-1)
        return to old ra
```

$sp

- a0 = 1
- ra = 4024
- a0 = 2
- ra= 4024
- a0 = 3
- ra = 1004

n=3 a0

result v0

The growth of the stack as the recursion unfolds

Now test if n < 1 (i.e. n = 0). In that case return 0 to $v0.

```
     slti  $t0, $a0, 1          # if n ≥ 1 then goto L1
     beq   $t0, $zero, L1
     addi  $v0, $zero, 1        # return 1 to $v0
     addi  $sp, $sp, 8          # pop 2 items from stack
     jr    $ra                  # return
L1:  subi  $a0, $a0, 1          # decrement n
     jal   fact                 # call fact with (n – 1)
```

Now, we need to compute n * fact (n-1)

```
     lw    $a0, 0($sp)          # restore argument n
     lw    $ra, 4($sp)          # restore return address
     addi  $sp, $sp, 8          # pop 2 items
     mult  $v0, $a0, $v0        # return n * fact(n-1)
     jr    $ra                  # return to caller
```