

# MISC: FACTORIAL IN ACTION

## Computer Systems and Networks

---

Dr. Pallipuram

([vpallipuramkrishnamani@pacific.edu](mailto:vpallipuramkrishnamani@pacific.edu))

# C Factorial

---

```
int main()
{
    n=3;
    x=factorial(n);
}
int factorial(int n)
{
    if(n>=1)
        return n*factorial(n-1);
    else
        return 1;
}
```

**Register Map**  
\$s0: n  
\$v0: return val.  
\$s1: x  
\$a0: argument

MIPS Code:

For simplicity, main does not back up \$t0-\$t9; \$a0-\$a3

factorial function:

- must take the back-up both as a caller and callee
- back-up only \$s registers used by previous callers
- must back-up \$ra

NOTE: Aggressive back-up is guaranteed to work!

Inst. Addr.	Instruction			Inst. Addr.	Instruction																																				
32	.globl main		<table border="1"><tr><td>s0</td><td>s1</td></tr><tr><td>0</td><td>0</td></tr></table>	s0	s1	0	0	86	addi \$sp, \$sp, -4																																
s0	s1																																								
0	0																																								
36	.text		<table border="1"><tr><td>a0</td><td>v0</td><td>ra</td></tr><tr><td>0</td><td>0</td><td>0</td></tr></table>	a0	v0	ra	0	0	0	90	sw \$a0, 0(\$sp)																														
a0	v0	ra																																							
0	0	0																																							
40 main:	li \$s0, 3 #n=3			94	addi \$a0, \$a0, -1 #new arg. = n-1																																				
44	move \$a0, \$s0 #arg			98	jal factorial																																				
48	jal factorial			102	lw \$a0, 0(\$sp)																																				
52	move \$s1, \$v0 #ret.		<table border="1"><tr><td>\$sp</td><td>stack add.</td><td>value</td></tr><tr><td>256</td><td></td><td></td></tr><tr><td>252</td><td></td><td></td></tr><tr><td>248</td><td></td><td></td></tr><tr><td>244</td><td></td><td></td></tr><tr><td>240</td><td></td><td></td></tr><tr><td>236</td><td></td><td></td></tr><tr><td>232</td><td></td><td></td></tr><tr><td>228</td><td></td><td></td></tr><tr><td>224</td><td></td><td></td></tr><tr><td>220</td><td></td><td></td></tr><tr><td>216</td><td></td><td></td></tr></table>	\$sp	stack add.	value	256			252			248			244			240			236			232			228			224			220			216			106	addi \$sp, \$sp, 4
\$sp	stack add.	value																																							
256																																									
252																																									
248																																									
244																																									
240																																									
236																																									
232																																									
228																																									
224																																									
220																																									
216																																									
56	li \$v0, 10			110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)																																				
60	syscall			114	j return																																				
64 factorial:	addi \$sp, \$sp, -4			118	li \$v0, 1																																				
68	sw \$s0, 0(\$sp)			else1:																																					
72	addi \$sp, \$sp, -4			122	j return																																				
74	sw \$ra, 0(\$sp)			126 return:	lw \$ra, 0(\$sp)																																				
78	li \$t1, 1 #constant			130	addi \$sp, \$sp, 4																																				
82 if1:	blt \$a0, \$t1, else1			134	lw \$s0, 0(\$sp)																																				
				138	addi \$sp, \$sp, 4																																				
				142	jr \$ra																																				

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		<b>s0</b>	<b>s1</b>	
36	.text		0	0	86 addi \$sp, \$sp, -4
40 main:	li \$s0, 3 #n=3		<b>a0</b>	<b>v0</b>	90 sw \$a0, 0(\$sp)
44	move \$a0, \$s0 #arg		0	ra	94 addi \$a0, \$a0, -1 #new arg. = n-1
48	jal factorial				98 jal factorial
52	move \$s1, \$v0 #ret.				102 lw \$a0, 0(\$sp)
56	li \$v0, 10				106 addi \$sp, \$sp, 4
60	syscall				110 mul \$v0, \$a0, \$v0 #n*factorial(n-1)
64 factorial:	addi \$sp, \$sp, -4				114 j return
68	sw \$s0, 0(\$sp)				118 li \$v0, 1
72	addi \$sp, \$sp, -4				else1:
74	sw \$ra, 0(\$sp)				122 j return
78	li \$t1, 1 #constant				126 lw \$ra, 0(\$sp)
82 if1:	blt \$a0, \$t1, else1				return:
					130 addi \$sp, \$sp, 4
					134 lw \$s0, 0(\$sp)
					138 addi \$sp, \$sp, 4
					142 jr \$ra

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main			0	s0
36	.text			0	s1
PC →	40 main:	li \$s0, 3 #n=3		0	a0
	44	move \$a0, \$s0 #arg		0	v0
	48	jal factorial		0	ra
	52	move \$s1, \$v0 #ret.			stack add.
	56	li \$v0, 10		256	value
	60	syscall		252	
	64	addi \$sp, \$sp, -4		248	
	factorial:			244	
				240	
	68	sw \$s0, 0(\$sp)		236	
	72	addi \$sp, \$sp, -4		232	
	74	sw \$ra, 0(\$sp)		228	
	78	li \$t1, 1 #constant		224	
	82 if1:	blt \$a0, \$t1, else1		220	
				216	
				86	addi \$sp, \$sp, -4
				90	sw \$a0, 0(\$sp)
				94	addi \$a0, \$a0, -1 #new arg. = n-1
				98	jal factorial
				102	lw \$a0, 0(\$sp)
				106	addi \$sp, \$sp, 4
				110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
				114	j return
				118	li \$v0, 1
				else1:	
				122	j return
				126	lw \$ra, 0(\$sp)
				return:	
				130	addi \$sp, \$sp, 4
				134	lw \$s0, 0(\$sp)
				138	addi \$sp, \$sp, 4
				142	jr \$ra

Inst. Addr.	Instruction		s0	s1		Inst. Addr.	Instruction
32	.globl main		3	0		86	addi \$sp, \$sp, -4
36	.text		a0	v0	ra	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		0	0	0	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg					98	jal factorial
48	jal factorial					102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.					106	addi \$sp, \$sp, 4
56	li \$v0, 10					110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall					114	j return
64 factorial:	addi \$sp, \$sp, -4					118 else1:	li \$v0, 1
68	sw \$s0, 0(\$sp)					122	j return
72	addi \$sp, \$sp, -4					126 return:	lw \$ra, 0(\$sp)
74	sw \$ra, 0(\$sp)					130	addi \$sp, \$sp, 4
78	li \$t1, 1 #constant					134	lw \$s0, 0(\$sp)
82 if1:	blt \$a0, \$t1, else1					138	addi \$sp, \$sp, 4
						142	jr \$ra

Inst. Addr.	Instruction		s0	s1		Inst. Addr.	Instruction
32	.globl main		3	0		86	addi \$sp, \$sp, -4
36	.text		a0	v0	ra	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		3	0	0	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg					98	jal factorial
48	jal factorial					102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.					106	addi \$sp, \$sp, 4
56	li \$v0, 10					110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall					114	j return
64 factorial:	addi \$sp, \$sp, -4					118 else1:	li \$v0, 1
68	sw \$s0, 0(\$sp)					122	j return
72	addi \$sp, \$sp, -4					126 return:	lw \$ra, 0(\$sp)
74	sw \$ra, 0(\$sp)					130	addi \$sp, \$sp, 4
78	li \$t1, 1 #constant					134	lw \$s0, 0(\$sp)
82 if1:	blt \$a0, \$t1, else1					138	addi \$sp, \$sp, 4
						142	jr \$ra

Inst. Addr.	Instruction	s0	s1	Inst. Addr.	Instruction
32	.globl main	3	0	\$sp →	86 addi \$sp, \$sp, -4
36	.text	a0	v0		90 sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3	3	0		94 addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg	52			98 jal factorial
48	jal factorial				102 lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.	stack	value		106 addi \$sp, \$sp, 4
56	li \$v0, 10	add.			110 mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	256			114 j return
64 factorial:	addi \$sp, \$sp, -4	252			118 li \$v0, 1 else1:
68	sw \$s0, 0(\$sp)	248			122 j return
72	addi \$sp, \$sp, -4	244			126 lw \$ra, 0(\$sp) return:
74	sw \$ra, 0(\$sp)	240			130 addi \$sp, \$sp, 4
78	li \$t1, 1 #constant	236			134 lw \$s0, 0(\$sp)
82 if1:	blt \$a0, \$t1, else1	232			138 addi \$sp, \$sp, 4
		228			142 jr \$ra
		224			
		220			
		216			

Inst. Addr.	Instruction	s0	s1	Inst. Addr.	Instruction
32	.globl main	3	0	86	addi \$sp, \$sp, -4
36	.text	a0	v0	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3	3	0	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			98	jal factorial
48	jal factorial			102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.			106	addi \$sp, \$sp, 4
56	li \$v0, 10			110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall			114	j return
64 factorial:	addi \$sp, \$sp, -4			118 else1:	li \$v0, 1
68	sw \$s0, 0(\$sp)			122	j return
72	addi \$sp, \$sp, -4			126 return:	lw \$ra, 0(\$sp)
74	sw \$ra, 0(\$sp)			130	addi \$sp, \$sp, 4
78	li \$t1, 1 #constant			134	lw \$s0, 0(\$sp)
82 if1:	blt \$a0, \$t1, else1			138	addi \$sp, \$sp, 4
				142	jr \$ra

Inst. Addr.	Instruction			Inst. Addr.	Instruction																								
32	.globl main		<table border="1"><tr><td>s0</td><td>s1</td></tr><tr><td>3</td><td>0</td></tr></table>	s0	s1	3	0	86	addi \$sp, \$sp, -4																				
s0	s1																												
3	0																												
36	.text		<table border="1"><tr><td>a0</td><td>v0</td><td>ra</td></tr><tr><td>3</td><td>0</td><td>52</td></tr></table>	a0	v0	ra	3	0	52	90	sw \$a0, 0(\$sp)																		
a0	v0	ra																											
3	0	52																											
40 main:	li \$s0, 3 #n=3			94	addi \$a0, \$a0, -1 #new arg. = n-1																								
44	move \$a0, \$s0 #arg			98	jal factorial																								
48	jal factorial			102	lw \$a0, 0(\$sp)																								
52	move \$s1, \$v0 #ret.		<table border="1"><tr><td>stack add.</td><td>value</td></tr><tr><td>256</td><td></td></tr><tr><td>252</td><td>3</td></tr><tr><td>248</td><td></td></tr><tr><td>244</td><td></td></tr><tr><td>240</td><td></td></tr><tr><td>236</td><td></td></tr><tr><td>232</td><td></td></tr><tr><td>228</td><td></td></tr><tr><td>224</td><td></td></tr><tr><td>220</td><td></td></tr><tr><td>216</td><td></td></tr></table>	stack add.	value	256		252	3	248		244		240		236		232		228		224		220		216		106	addi \$sp, \$sp, 4
stack add.	value																												
256																													
252	3																												
248																													
244																													
240																													
236																													
232																													
228																													
224																													
220																													
216																													
56	li \$v0, 10		\$sp →	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)																								
60	syscall			114	j return																								
64 factorial:	addi \$sp, \$sp, -4			118 else1:	li \$v0, 1																								
68	sw \$s0, 0(\$sp)			122	j return																								
72	addi \$sp, \$sp, -4			126 return:	lw \$ra, 0(\$sp)																								
74	sw \$ra, 0(\$sp)			130	addi \$sp, \$sp, 4																								
78	li \$t1, 1 #constant			134	lw \$s0, 0(\$sp)																								
82 if1:	blt \$a0, \$t1, else1			138	addi \$sp, \$sp, 4																								
				142	jr \$ra																								

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 0	
36	.text		a0 3	v0 0	
40 main:	li \$s0, 3 #n=3		ra 52		
44	move \$a0, \$s0 #arg				
48	jal factorial				
52	move \$s1, \$v0 #ret.				
56	li \$v0, 10				
60	syscall				
64 factorial:	addi \$sp, \$sp, -4				
68	sw \$s0, 0(\$sp)				
72	addi \$sp, \$sp, -4				
74	sw \$ra, 0(\$sp)				
78	li \$t1, 1 #constant				
82 if1:	blt \$a0, \$t1, else1				
			stack add. 256	value	
			252	3	
		\$sp →	248		
			244		
			240		
			236		
			232		
			228		
			224		
			220		
			216		
PC					

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 0	
36	.text		a0 3	v0 0	
40 main:	li \$s0, 3 #n=3		ra 52		
44	move \$a0, \$s0 #arg				
48	jal factorial				
52	move \$s1, \$v0 #ret.				
56	li \$v0, 10				
60	syscall				
64 factorial:	addi \$sp, \$sp, -4				
68	sw \$s0, 0(\$sp)				
72	addi \$sp, \$sp, -4				
74	sw \$ra, 0(\$sp)				
78	li \$t1, 1 #constant				
82 if1:	blt \$a0, \$t1, else1				
			stack add. 256	value	
			252	3	
		\$sp →	248	52	
			244		
			240		
			236		
			232		
			228		
			224		
			220		
			216		
86	addi \$sp, \$sp, -4				
90	sw \$a0, 0(\$sp)				
94	addi \$a0, \$a0, -1 #new arg. = n-1				
98	jal factorial				
102	lw \$a0, 0(\$sp)				
106	addi \$sp, \$sp, 4				
110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)				
114	j return				
118 else1:	li \$v0, 1				
122	j return				
126 return:	lw \$ra, 0(\$sp)				
130	addi \$sp, \$sp, 4				
134	lw \$s0, 0(\$sp)				
138	addi \$sp, \$sp, 4				
142	jr \$ra				

PC  
→

Inst. Addr.	Instruction
32	.globl main
36	.text
40 main:	li \$s0, 3 #n=3
44	move \$a0, \$s0 #arg
48	jal factorial
52	move \$s1, \$v0 #ret.
56	li \$v0, 10
60	syscall
64 factorial:	addi \$sp, \$sp, -4
68	sw \$s0, 0(\$sp)
72	addi \$sp, \$sp, -4
74	sw \$ra, 0(\$sp)
78	li \$t1, 1 #constant
82 if1:	blt \$a0, \$t1, else1



Inst. Addr.	Instruction
86	addi \$sp, \$sp, -4
90	sw \$a0, 0(\$sp)
94	addi \$a0, \$a0, -1 #new arg. = n-1
98	jal factorial
102	lw \$a0, 0(\$sp)
106	addi \$sp, \$sp, 4
110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
114	j return
118 else1:	li \$v0, 1
122	j return
126 return:	lw \$ra, 0(\$sp)
130	addi \$sp, \$sp, 4
134	lw \$s0, 0(\$sp)
138	addi \$sp, \$sp, 4
142	jr \$ra

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 0	
36	.text		a0 3	v0 0	
40 main:	li \$s0, 3 #n=3			ra 52	
44	move \$a0, \$s0 #arg				\$t1=1 blt fails
48	jal factorial				stack add.
52	move \$s1, \$v0 #ret.				value
56	li \$v0, 10			256	
60	syscall			252	3
64 factorial:	addi \$sp, \$sp, -4		\$sp 248	52	
68	sw \$s0, 0(\$sp)			244	
72	addi \$sp, \$sp, -4			240	
74	sw \$ra, 0(\$sp)			236	
78	li \$t1, 1 #constant			232	
82 if1:	blt \$a0, \$t1, else1			228	
				224	
				220	
				216	
			PC →	86	addi \$sp, \$sp, -4
				90	sw \$a0, 0(\$sp)
				94	addi \$a0, \$a0, -1 #new arg. = n-1
				98	jal factorial
				102	lw \$a0, 0(\$sp)
				106	addi \$sp, \$sp, 4
				110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
				114	j return
				118 else1:	li \$v0, 1
				122	j return
				126 return:	lw \$ra, 0(\$sp)
				130	addi \$sp, \$sp, 4
				134	lw \$s0, 0(\$sp)
				138	addi \$sp, \$sp, 4
				142	jr \$ra

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 0	
36	.text		a0 3	v0 0	
40 main:	li \$s0, 3 #n=3		ra 52		
44	move \$a0, \$s0 #arg			\$t1=1	
48	jal factorial			stack add.	value
52	move \$s1, \$v0 #ret.			256	
56	li \$v0, 10			252	3
60	syscall			248	52
64 factorial:	addi \$sp, \$sp, -4		\$sp 244		
68	sw \$s0, 0(\$sp)			240	
72	addi \$sp, \$sp, -4			236	
74	sw \$ra, 0(\$sp)			232	
78	li \$t1, 1 #constant			228	
82 if1:	blt \$a0, \$t1, else1			224	
				220	
				216	
			PC →	86	addi \$sp, \$sp, -4
				90	sw \$a0, 0(\$sp)
				94	addi \$a0, \$a0, -1 #new arg. = n-1
				98	jal factorial
				102	lw \$a0, 0(\$sp)
				106	addi \$sp, \$sp, 4
				110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
				114	j return
				118 else1:	li \$v0, 1
				122	j return
				126 return:	lw \$ra, 0(\$sp)
				130	addi \$sp, \$sp, 4
				134	lw \$s0, 0(\$sp)
				138	addi \$sp, \$sp, 4
				142	jr \$ra

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main	s0 3	s1 0	86	addi \$sp, \$sp, -4
36	.text	a0 3	v0 0	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3	ra 52		94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg		\$t1=1	98	jal factorial
48	jal factorial		stack add.	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.		value	106	addi \$sp, \$sp, 4
56	li \$v0, 10	256		110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	252	3	114	j return
64 factorial:	addi \$sp, \$sp, -4	248	52	118 else1:	li \$v0, 1
68	sw \$s0, 0(\$sp)	244	3	122	j return
72	addi \$sp, \$sp, -4	240		126 return:	lw \$ra, 0(\$sp)
74	sw \$ra, 0(\$sp)	236		130	addi \$sp, \$sp, 4
78	li \$t1, 1 #constant	232		134	lw \$s0, 0(\$sp)
82 if1:	blt \$a0, \$t1, else1	228		138	addi \$sp, \$sp, 4
		224		142	jr \$ra
		220			
		216			

Inst. Addr.	Instruction			Inst. Addr.	Instruction
		s0	s1		
32	.globl main	3	0	86	addi \$sp, \$sp, -4
36	.text	a0	v0	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3	2	0	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg		\$t1=1	98	jal factorial
48	jal factorial		stack add.	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.		value	106	addi \$sp, \$sp, 4
56	li \$v0, 10	256		110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	252	3	114	j return
64 factorial:	addi \$sp, \$sp, -4	248	52	118 else1:	li \$v0, 1
68	sw \$s0, 0(\$sp)	244	3	122	j return
72	addi \$sp, \$sp, -4	240		126 return:	lw \$ra, 0(\$sp)
74	sw \$ra, 0(\$sp)	236		130	addi \$sp, \$sp, 4
78	li \$t1, 1 #constant	232		134	lw \$s0, 0(\$sp)
82 if1:	blt \$a0, \$t1, else1	228		138	addi \$sp, \$sp, 4
		224		142	jr \$ra
		220			
		216			

Inst. Addr.	Instruction	s0	s1	Inst. Addr.	Instruction
32	.globl main	3	0	86	addi \$sp, \$sp, -4
36	.text	a0	v0		sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3	2	ra		addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg				jal factorial
48	jal factorial				lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.				addi \$sp, \$sp, 4
56	li \$v0, 10				mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall				j return
64 factorial:	addi \$sp, \$sp, -4				li \$v0, 1
68	sw \$s0, 0(\$sp)				else1:
72	addi \$sp, \$sp, -4				j return
74	sw \$ra, 0(\$sp)				lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant				addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1				lw \$s0, 0(\$sp)
					addi \$sp, \$sp, 4
					jr \$ra

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 0	
36	.text		a0 2	v0 0	86 addi \$sp, \$sp, -4
40 main:	li \$s0, 3 #n=3			ra 102	90 sw \$a0, 0(\$sp)
44	move \$a0, \$s0 #arg				94 addi \$a0, \$a0, -1 #new arg. = n-1
48	jal factorial				98 jal factorial
52	move \$s1, \$v0 #ret.				102 lw \$a0, 0(\$sp)
56	li \$v0, 10				106 addi \$sp, \$sp, 4
60	syscall				110 mul \$v0, \$a0, \$v0 #n*factorial(n-1)
64 factorial:	addi \$sp, \$sp, -4				114 j return
68	sw \$s0, 0(\$sp)	\$sp →	stack add. 256		118 li \$v0, 1
72	addi \$sp, \$sp, -4		252	3	else1:
74	sw \$ra, 0(\$sp)		248	52	
78	li \$t1, 1 #constant		244	3	
82 if1:	blt \$a0, \$t1, else1		240		
			236		122 j return
			232		126 lw \$ra, 0(\$sp)
			228		return:
			224		
			220		130 addi \$sp, \$sp, 4
			216		134 lw \$s0, 0(\$sp)
					138 addi \$sp, \$sp, 4
					142 jr \$ra

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main	s0 3	s1 0	86	addi \$sp, \$sp, -4
36	.text	a0 2	v0 0	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		ra 102	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			98	jal factorial
48	jal factorial			102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.		\$t1=1	106	addi \$sp, \$sp, 4
56	li \$v0, 10	stack add. 256	value	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	252	3	114	j return
64 factorial:	addi \$sp, \$sp, -4	248	52	118 else1:	li \$v0, 1
68	sw \$s0, 0(\$sp)	244	3	122	j return
72	addi \$sp, \$sp, -4	240	3	126 return:	lw \$ra, 0(\$sp)
74	sw \$ra, 0(\$sp)	236		130	addi \$sp, \$sp, 4
78	li \$t1, 1 #constant	232		134	lw \$s0, 0(\$sp)
82 if1:	blt \$a0, \$t1, else1	228		138	addi \$sp, \$sp, 4
		224		142	jr \$ra
		220			
		216			

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 0	
36	.text		a0 2	v0 0	
40 main:	li \$s0, 3 #n=3		ra 102		
44	move \$a0, \$s0 #arg			\$t1=1	
48	jal factorial			stack add.	value
52	move \$s1, \$v0 #ret.			256	
56	li \$v0, 10			252	3
60	syscall			248	52
64 factorial:	addi \$sp, \$sp, -4			244	3
68	sw \$s0, 0(\$sp)			240	3
72	addi \$sp, \$sp, -4		\$sp 236		
74	sw \$ra, 0(\$sp)			232	
78	li \$t1, 1 #constant			228	
82 if1:	blt \$a0, \$t1, else1			224	
				220	
				216	
PC →				142	jr \$ra
				138	addi \$sp, \$sp, 4
				134	lw \$s0, 0(\$sp)
				130	addi \$sp, \$sp, 4
				126 return:	lw \$ra, 0(\$sp)
				122	j return
				118 else1:	li \$v0, 1
				114	j return
				110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
				106	addi \$sp, \$sp, 4
				102	lw \$a0, 0(\$sp)
				98	jal factorial
				94	addi \$a0, \$a0, -1 #new arg. = n-1
				90	sw \$a0, 0(\$sp)
				86	addi \$sp, \$sp, -4

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 0	
36	.text		a0 2	v0 0	
40 main:	li \$s0, 3 #n=3		ra 102		
44	move \$a0, \$s0 #arg			\$t1=1	
48	jal factorial			stack add.	value
52	move \$s1, \$v0 #ret.			256	
56	li \$v0, 10			252	3
60	syscall			248	52
64 factorial:	addi \$sp, \$sp, -4			244	3
68	sw \$s0, 0(\$sp)			240	3
72	addi \$sp, \$sp, -4		\$sp 236	102	
74	sw \$ra, 0(\$sp)			232	
78	li \$t1, 1 #constant			228	
82 if1:	blt \$a0, \$t1, else1			224	
				220	
				216	
86	addi \$sp, \$sp, -4				
90	sw \$a0, 0(\$sp)				
94	addi \$a0, \$a0, -1 #new arg. = n-1				
98	jal factorial				
102	lw \$a0, 0(\$sp)				
106	addi \$sp, \$sp, 4				
110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)				
114	j return				
118 else1:	li \$v0, 1				
122	j return				
126 return:	lw \$ra, 0(\$sp)				
130	addi \$sp, \$sp, 4				
134	lw \$s0, 0(\$sp)				
138	addi \$sp, \$sp, 4				
142	jr \$ra				

PC  
→

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main	s0 3	s1 0	86	addi \$sp, \$sp, -4
36	.text	a0 2	v0 0	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		ra 102	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			98	jal factorial
48	jal factorial			102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.		\$t1=1	106	addi \$sp, \$sp, 4
56	li \$v0, 10	stack add. 256	value	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	252	3	114	j return
64 factorial:	addi \$sp, \$sp, -4	248	52	118 else1:	li \$v0, 1
68	sw \$s0, 0(\$sp)	244	3	122	j return
72	addi \$sp, \$sp, -4	240	3	126 return:	lw \$ra, 0(\$sp)
74	sw \$ra, 0(\$sp)	236	102	130	addi \$sp, \$sp, 4
78	li \$t1, 1 #constant	232		134	lw \$s0, 0(\$sp)
82 if1:	blt \$a0, \$t1, else1	228		138	addi \$sp, \$sp, 4
		224		142	jr \$ra
		220			
		216			

PC  
→

Inst. Addr.	Instruction		s0	s1		Inst. Addr.	Instruction
32	.globl main		3	0		86	addi \$sp, \$sp, -4
36	.text		a0	v0	ra	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		2	0	102	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			\$t1=1	blt fails	98	jal factorial
48	jal factorial			stack add.	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.			256		106	addi \$sp, \$sp, 4
56	li \$v0, 10			252	3	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall			248	52	114	j return
64 factorial:	addi \$sp, \$sp, -4			244	3	118 else1:	li \$v0, 1
68	sw \$s0, 0(\$sp)		\$sp	240	3	122	j return
72	addi \$sp, \$sp, -4		236	102		126 return:	lw \$ra, 0(\$sp)
74	sw \$ra, 0(\$sp)		232			130	addi \$sp, \$sp, 4
78	li \$t1, 1 #constant		228			134	lw \$s0, 0(\$sp)
82 if1:	blt \$a0, \$t1, else1		224			138	addi \$sp, \$sp, 4
			220			142	jr \$ra
			216				

Inst. Addr.	Instruction		s0	s1		Inst. Addr.	Instruction
32	.globl main		3	0		86	addi \$sp, \$sp, -4
36	.text		a0	v0	ra	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		2	0	102	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			\$t1=1		98	jal factorial
48	jal factorial			stack	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.			add.		106	addi \$sp, \$sp, 4
56	li \$v0, 10			256		110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall			252	3	114	j return
64 factorial:	addi \$sp, \$sp, -4			248	52	118 else1:	li \$v0, 1
68	sw \$s0, 0(\$sp)			244	3	122	j return
72	addi \$sp, \$sp, -4			240	3	126 return:	lw \$ra, 0(\$sp)
74	sw \$ra, 0(\$sp)			236	102	130	addi \$sp, \$sp, 4
78	li \$t1, 1 #constant			232		134	lw \$s0, 0(\$sp)
82 if1:	blt \$a0, \$t1, else1			228		138	addi \$sp, \$sp, 4
				224		142	jr \$ra
				220			
				216			

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 0	
36	.text		a0 2	v0 0	
40 main:	li \$s0, 3 #n=3		ra 102		
44	move \$a0, \$s0 #arg			\$t1=1	
48	jal factorial			stack add.	value
52	move \$s1, \$v0 #ret.			256	
56	li \$v0, 10			252	3
60	syscall			248	52
64 factorial:	addi \$sp, \$sp, -4			244	3
68	sw \$s0, 0(\$sp)			240	3
72	addi \$sp, \$sp, -4		\$sp 236	102	
74	sw \$ra, 0(\$sp)			232	2
78	li \$t1, 1 #constant			228	
82 if1:	blt \$a0, \$t1, else1			224	
				220	
				216	
					addi \$sp, \$sp, -4
					sw \$a0, 0(\$sp)
					addi \$a0, \$a0, -1
					#new arg. = n-1
					jal factorial
					lw \$a0, 0(\$sp)
					addi \$sp, \$sp, 4
					mul \$v0, \$a0, \$v0
					#n*factorial(n-1)
					j return
					li \$v0, 1
					else1:
					j return
					lw \$ra, 0(\$sp)
					return:
					addi \$sp, \$sp, 4
					lw \$s0, 0(\$sp)
					addi \$sp, \$sp, 4
					jr \$ra

Inst. Addr.	Instruction			Inst. Addr.	Instruction
		s0	s1		
32	.globl main	3	0	86	addi \$sp, \$sp, -4
36	.text	a0	v0	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3	1	0	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg		\$t1=1	PC → 98	jal factorial
48	jal factorial		stack add.	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.	256		106	addi \$sp, \$sp, 4
56	li \$v0, 10	252	3	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	248	52	114	j return
64 factorial:	addi \$sp, \$sp, -4	244	3	118 else1:	li \$v0, 1
68	sw \$s0, 0(\$sp)	240	3	122	j return
72	addi \$sp, \$sp, -4	236	102	126 return:	lw \$ra, 0(\$sp)
74	sw \$ra, 0(\$sp)	232	2	130	addi \$sp, \$sp, 4
78	li \$t1, 1 #constant	228		134	lw \$s0, 0(\$sp)
82 if1:	blt \$a0, \$t1, else1	224		138	addi \$sp, \$sp, 4
		220		142	jr \$ra
		216			

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 0	
36	.text		a0 1	v0 0	86 addi \$sp, \$sp, -4
40 main:	li \$s0, 3 #n=3		ra 102		90 sw \$a0, 0(\$sp)
44	move \$a0, \$s0 #arg				94 addi \$a0, \$a0, -1 #new arg. = n-1
48	jal factorial				98 jal factorial
52	move \$s1, \$v0 #ret.				102 lw \$a0, 0(\$sp)
56	li \$v0, 10				106 addi \$sp, \$sp, 4
60	syscall				110 mul \$v0, \$a0, \$v0 #n*factorial(n-1)
PC 64 factorial:	addi \$sp, \$sp, -4				114 j return
	sw \$s0, 0(\$sp)				118 else1: li \$v0, 1
	addi \$sp, \$sp, -4				122 j return
	sw \$ra, 0(\$sp)				126 return: lw \$ra, 0(\$sp)
	li \$t1, 1 #constant				130 addi \$sp, \$sp, 4
	blt \$a0, \$t1, else1				134 lw \$s0, 0(\$sp)
					138 addi \$sp, \$sp, 4
					142 jr \$ra

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 0	
36	.text		a0 1	v0 0	86 addi \$sp, \$sp, -4
40 main:	li \$s0, 3 #n=3		ra 102		90 sw \$a0, 0(\$sp)
44	move \$a0, \$s0 #arg			\$t1=1	94 addi \$a0, \$a0, -1 #new arg. = n-1
48	jal factorial				98 jal factorial
52	move \$s1, \$v0 #ret.				102 lw \$a0, 0(\$sp)
56	li \$v0, 10				106 addi \$sp, \$sp, 4
60	syscall				110 mul \$v0, \$a0, \$v0 #n*factorial(n-1)
64 factorial:	addi \$sp, \$sp, -4				114 j return
68	sw \$s0, 0(\$sp)				118 else1:
72	addi \$sp, \$sp, -4				122 j return
74	sw \$ra, 0(\$sp)				126 return:
78	li \$t1, 1 #constant				130 addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1				134 lw \$s0, 0(\$sp)
					138 addi \$sp, \$sp, 4
					142 jr \$ra

PC →

s0	s1
3	0

a0	v0	ra
1	0	102

stack add.	value
256	
252	3
248	52
244	3
240	3
236	102
232	2
228	
224	
220	
216	

\$sp →

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 0	
36	.text		a0 1	v0 0	86 addi \$sp, \$sp, -4
40 main:	li \$s0, 3 #n=3			ra 102	90 sw \$a0, 0(\$sp)
44	move \$a0, \$s0 #arg				94 addi \$a0, \$a0, -1 #new arg. = n-1
48	jal factorial				98 jal factorial
52	move \$s1, \$v0 #ret.				102 lw \$a0, 0(\$sp)
56	li \$v0, 10				106 addi \$sp, \$sp, 4
60	syscall				110 mul \$v0, \$a0, \$v0 #n*factorial(n-1)
64 factorial:	addi \$sp, \$sp, -4				114 j return
68	sw \$s0, 0(\$sp)				118 else1:
72	addi \$sp, \$sp, -4	\$sp →	stack add. 256	value 3	122 j return
74	sw \$ra, 0(\$sp)		252	52	126 return:
78	li \$t1, 1 #constant		248	3	130 addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1		244	3	134 lw \$s0, 0(\$sp)
			240	3	138 addi \$sp, \$sp, 4
			236	102	142 jr \$ra
			232	2	
			228	3	
			224		
			220		
			216		

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 0	
36	.text		a0 1	v0 0	86 addi \$sp, \$sp, -4
40 main:	li \$s0, 3 #n=3		ra 102		90 sw \$a0, 0(\$sp)
44	move \$a0, \$s0 #arg			\$t1=1	94 addi \$a0, \$a0, -1 #new arg. = n-1
48	jal factorial				98 jal factorial
52	move \$s1, \$v0 #ret.				102 lw \$a0, 0(\$sp)
56	li \$v0, 10				106 addi \$sp, \$sp, 4
60	syscall				110 mul \$v0, \$a0, \$v0 #n*factorial(n-1)
64 factorial:	addi \$sp, \$sp, -4				114 j return
68	sw \$s0, 0(\$sp)				118 else1:
72	addi \$sp, \$sp, -4				122 j return
74	sw \$ra, 0(\$sp)				126 return:
78	li \$t1, 1 #constant				130 addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1				134 lw \$s0, 0(\$sp)
					138 addi \$sp, \$sp, 4
					142 jr \$ra

PC  
→

s0	s1
3	0

a0	v0	ra
1	0	102

stack add.	value
256	
252	3
248	52
244	3
240	3
236	102
232	2
228	3
224	
220	
216	

→ \$sp

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 0	
36	.text		a0 1	v0 0	86 addi \$sp, \$sp, -4
40 main:	li \$s0, 3 #n=3		ra 102		90 sw \$a0, 0(\$sp)
44	move \$a0, \$s0 #arg			\$t1=1	94 addi \$a0, \$a0, -1 #new arg. = n-1
48	jal factorial				98 jal factorial
52	move \$s1, \$v0 #ret.				102 lw \$a0, 0(\$sp)
56	li \$v0, 10				106 addi \$sp, \$sp, 4
60	syscall				110 mul \$v0, \$a0, \$v0 #n*factorial(n-1)
64 factorial:	addi \$sp, \$sp, -4				114 j return
68	sw \$s0, 0(\$sp)				118 else1:
72	addi \$sp, \$sp, -4				122 j return
74	sw \$ra, 0(\$sp)				126 return:
78	li \$t1, 1 #constant				130 addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1				134 lw \$s0, 0(\$sp)
			stack add. 256 252 248 244 240 236 232 228 224 220 216	value 3 52 3 3 3 102 2 3 102	138 addi \$sp, \$sp, 4 142 jr \$ra
		PC →	\$sp		

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 0	
36	.text		a0 1	v0 0	86 addi \$sp, \$sp, -4
40 main:	li \$s0, 3 #n=3		ra 102		90 sw \$a0, 0(\$sp)
44	move \$a0, \$s0 #arg			\$t1=1	94 addi \$a0, \$a0, -1 #new arg. = n-1
48	jal factorial			stack add.	98 jal factorial
52	move \$s1, \$v0 #ret.			value	102 lw \$a0, 0(\$sp)
56	li \$v0, 10		256		106 addi \$sp, \$sp, 4
60	syscall		252	3	110 mul \$v0, \$a0, \$v0 #n*factorial(n-1)
64 factorial:	addi \$sp, \$sp, -4		248	52	114 j return
68	sw \$s0, 0(\$sp)		244	3	118 else1:
72	addi \$sp, \$sp, -4		240	3	122 j return
74	sw \$ra, 0(\$sp)		236	102	126 return:
78	li \$t1, 1 #constant		232	2	130 addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1	PC →	228	3	134 lw \$s0, 0(\$sp)
			224	102	138 addi \$sp, \$sp, 4
			220		142 jr \$ra
			216		

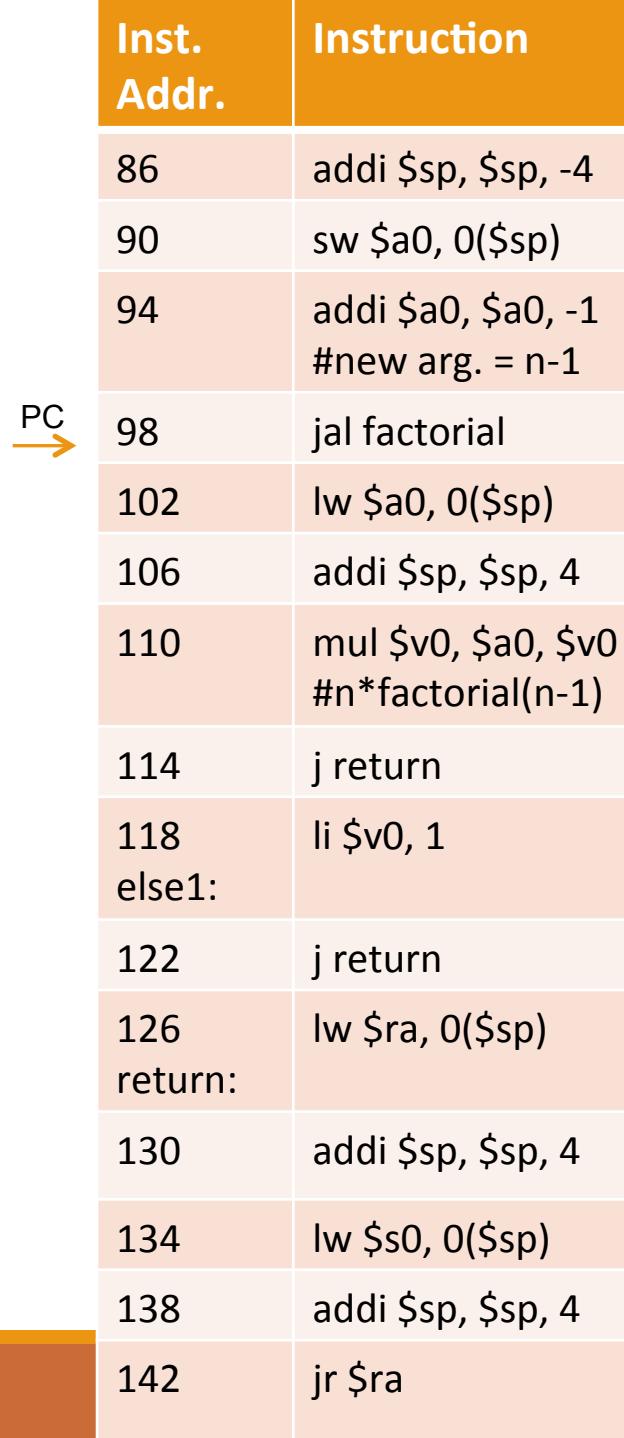
Inst. Addr.	Instruction		s0	s1		Inst. Addr.	Instruction
32	.globl main		3	0		86	addi \$sp, \$sp, -4
36	.text		a0	v0	ra	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		1	0	102	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			\$t1=1	blt fails	98	jal factorial
48	jal factorial			stack add.	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.			256		106	addi \$sp, \$sp, 4
56	li \$v0, 10			252	3	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall			248	52	114	j return
64 factorial:	addi \$sp, \$sp, -4			244	3	118 else1:	li \$v0, 1
68	sw \$s0, 0(\$sp)			240	3	122	j return
72	addi \$sp, \$sp, -4			236	102	126 return:	lw \$ra, 0(\$sp)
74	sw \$ra, 0(\$sp)			232	2	130	addi \$sp, \$sp, 4
78	li \$t1, 1 #constant			228	3	134	lw \$s0, 0(\$sp)
82 if1:	blt \$a0, \$t1, else1			224	102	138	addi \$sp, \$sp, 4
				220		142	jr \$ra
				216			

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 0	
36	.text		a0 1	v0 0	
40 main:	li \$s0, 3 #n=3		ra 102		
44	move \$a0, \$s0 #arg			\$t1=1	
48	jal factorial			stack add.	value
52	move \$s1, \$v0 #ret.			256	
56	li \$v0, 10			252	3
60	syscall			248	52
64 factorial:	addi \$sp, \$sp, -4			244	3
68	sw \$s0, 0(\$sp)			240	3
72	addi \$sp, \$sp, -4			236	102
74	sw \$ra, 0(\$sp)			232	2
78	li \$t1, 1 #constant			228	3
82 if1:	blt \$a0, \$t1, else1			224	102
		\$sp		220	
				216	
			PC		
				86	addi \$sp, \$sp, -4
				90	sw \$a0, 0(\$sp)
				94	addi \$a0, \$a0, -1 #new arg. = n-1
				98	jal factorial
				102	lw \$a0, 0(\$sp)
				106	addi \$sp, \$sp, 4
				110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
				114	j return
				118 else1:	li \$v0, 1
				122	j return
				126 return:	lw \$ra, 0(\$sp)
				130	addi \$sp, \$sp, 4
				134	lw \$s0, 0(\$sp)
				138	addi \$sp, \$sp, 4
				142	jr \$ra

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 0	
36	.text		a0 1	v0 0	
40 main:	li \$s0, 3 #n=3		ra 102		
44	move \$a0, \$s0 #arg			\$t1=1	
48	jal factorial			stack add.	value
52	move \$s1, \$v0 #ret.			256	
56	li \$v0, 10			252	3
60	syscall			248	52
64 factorial:	addi \$sp, \$sp, -4			244	3
68	sw \$s0, 0(\$sp)			240	3
72	addi \$sp, \$sp, -4			236	102
74	sw \$ra, 0(\$sp)			232	2
78	li \$t1, 1 #constant			228	3
82 if1:	blt \$a0, \$t1, else1			224	102
		\$sp		220	1
				216	
			PC		
				86	addi \$sp, \$sp, -4
				90	sw \$a0, 0(\$sp)
				94	addi \$a0, \$a0, -1 #new arg. = n-1
				98	jal factorial
				102	lw \$a0, 0(\$sp)
				106	addi \$sp, \$sp, 4
				110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
				114	j return
				118 else1:	li \$v0, 1
				122	j return
				126 return:	lw \$ra, 0(\$sp)
				130	addi \$sp, \$sp, 4
				134	lw \$s0, 0(\$sp)
				138	addi \$sp, \$sp, 4
				142	jr \$ra

Inst. Addr.	Instruction
32	.globl main
36	.text
40 main:	li \$s0, 3 #n=3
44	move \$a0, \$s0 #arg
48	jal factorial
52	move \$s1, \$v0 #ret.
56	li \$v0, 10
60	syscall
64 factorial:	addi \$sp, \$sp, -4
68	sw \$s0, 0(\$sp)
72	addi \$sp, \$sp, -4
74	sw \$ra, 0(\$sp)
78	li \$t1, 1 #constant
82 if1:	blt \$a0, \$t1, else1

s0	s1	
3	0	
a0	v0	ra
0	0	102
$\$t1=1$		
stack add.	value	
256		
252	3	
248	52	
244	3	
240	3	
236	102	
232	2	
228	3	
224	102	
220	1	
216		



Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main			86	addi \$sp, \$sp, -4
36	.text	a0	v0	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3	0	0	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			98	jal factorial
48	jal factorial			102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.	\$t1=1		106	addi \$sp, \$sp, 4
56	li \$v0, 10	stack add.	value	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	256		114	j return
PC →	64 factorial:	252	3	118	li \$v0, 1
	addi \$sp, \$sp, -4	248	52	else1:	
	68 sw \$s0, 0(\$sp)	244	3	122	j return
	72 addi \$sp, \$sp, -4	240	3	126	lw \$ra, 0(\$sp)
	74 sw \$ra, 0(\$sp)	236	102	return:	
	78 li \$t1, 1 #constant	232	2	130	addi \$sp, \$sp, 4
	82 if1: blt \$a0, \$t1, else1	228	3	134	lw \$s0, 0(\$sp)
		224	102	138	addi \$sp, \$sp, 4
		220	1	142	jr \$ra
		216			
		212			
		208			

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main			86	addi \$sp, \$sp, -4
36	.text			90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3	a0 0	v0 0	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			98	jal factorial
48	jal factorial			102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.		\$t1=1	106	addi \$sp, \$sp, 4
56	li \$v0, 10		stack add.	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall		value	114	j return
64 factorial:	addi \$sp, \$sp, -4		256		118 else1:
68	sw \$s0, 0(\$sp)		252	3	
72	addi \$sp, \$sp, -4		248	52	
74	sw \$ra, 0(\$sp)		244	3	
78	li \$t1, 1 #constant		240	3	
82 if1:	blt \$a0, \$t1, else1		236	102	
		\$sp →	232	2	
			228	3	
			224	102	
			220	1	
			216		
			212		
			208		

Inst. Addr.	Instruction	s0	s1	Inst. Addr.	Instruction	
32	.globl main	3	0	86 90 94 98 102 106 110 114 118 else1: 122 126 return: 130 134 138 142	addi \$sp, \$sp, -4	
36	.text	a0	v0		sw \$a0, 0(\$sp)	
40 main:	li \$s0, 3 #n=3	0	ra		addi \$a0, \$a0, -1 #new arg. = n-1	
44	move \$a0, \$s0 #arg	\$t1=1			jal factorial	
48	jal factorial	stack add.	value		lw \$a0, 0(\$sp)	
52	move \$s1, \$v0 #ret.	256			addi \$sp, \$sp, 4	
56	li \$v0, 10	252	3		mul \$v0, \$a0, \$v0 #n*factorial(n-1)	
60	syscall	248	52		j return	
64 factorial:	addi \$sp, \$sp, -4	244	3		li \$v0, 1	
68	sw \$s0, 0(\$sp)	240	3		else1:	
72	addi \$sp, \$sp, -4	236	102		j return	
74	sw \$ra, 0(\$sp)	232	2		lw \$ra, 0(\$sp)	
78	li \$t1, 1 #constant	228	3		addi \$sp, \$sp, 4	
82 if1:	blt \$a0, \$t1, else1	224	102		lw \$s0, 0(\$sp)	
		220	1		addi \$sp, \$sp, 4	
		216	3		jr \$ra	
		212				
		208				

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 0	
36	.text		a0 0	v0 0	
40 main:	li \$s0, 3 #n=3		ra 102		
44	move \$a0, \$s0 #arg			\$t1=1	
48	jal factorial			stack add.	value
52	move \$s1, \$v0 #ret.			256	
56	li \$v0, 10			252	3
60	syscall			248	52
64 factorial:	addi \$sp, \$sp, -4			244	3
68	sw \$s0, 0(\$sp)			240	3
72	addi \$sp, \$sp, -4			236	102
74	sw \$ra, 0(\$sp)			232	2
78	li \$t1, 1 #constant			228	3
82 if1:	blt \$a0, \$t1, else1			224	102
PC →				220	1
\$sp →				216	3
212				212	
208				208	
86	addi \$sp, \$sp, -4			86	
90	sw \$a0, 0(\$sp)			90	
94	addi \$a0, \$a0, -1 #new arg. = n-1			94	
98	jal factorial			98	
102	lw \$a0, 0(\$sp)			102	
106	addi \$sp, \$sp, 4			106	
110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)			110	
114	j return			114	
118	li \$v0, 1			118	
else1:				else1:	
122	j return			122	
126 return:	lw \$ra, 0(\$sp)			126 return:	
130	addi \$sp, \$sp, 4			130	
134	lw \$s0, 0(\$sp)			134	
138	addi \$sp, \$sp, 4			138	
142	jr \$ra			142	

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 0	
36	.text		a0 0	v0 0	
40 main:	li \$s0, 3 #n=3		ra 102		
44	move \$a0, \$s0 #arg	\$t1=1			
48	jal factorial	stack add.	value		
52	move \$s1, \$v0 #ret.	256			
56	li \$v0, 10	252	3		
60	syscall	248	52		
64 factorial:	addi \$sp, \$sp, -4	244	3		
68	sw \$s0, 0(\$sp)	240	3		
72	addi \$sp, \$sp, -4	236	102		
74	sw \$ra, 0(\$sp)	232	2		
78	li \$t1, 1 #constant	228	3		
82 if1:	blt \$a0, \$t1, else1	224	102		
		220	1		
		216	3		
		212	102		
		208			
PC →				86	addi \$sp, \$sp, -4
				90	sw \$a0, 0(\$sp)
				94	addi \$a0, \$a0, -1 #new arg. = n-1
				98	jal factorial
				102	lw \$a0, 0(\$sp)
				106	addi \$sp, \$sp, 4
				110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
				114	j return
				118	li \$v0, 1
				else1:	
				122	j return
				126 return:	lw \$ra, 0(\$sp)
				130	addi \$sp, \$sp, 4
				134	lw \$s0, 0(\$sp)
				138	addi \$sp, \$sp, 4
				142	jr \$ra

Inst. Addr.	Instruction			Inst. Addr.	Instruction
		s0	s1		
32	.globl main	3	0	86	addi \$sp, \$sp, -4
36	.text	a0	v0	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3	0	0	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg		\$t1=1	98	jal factorial
48	jal factorial		stack add.	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.	256		106	addi \$sp, \$sp, 4
56	li \$v0, 10	252	3	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	248	52	114	j return
64 factorial:	addi \$sp, \$sp, -4	244	3	118	li \$v0, 1
68	sw \$s0, 0(\$sp)	240	3	else1:	
72	addi \$sp, \$sp, -4	236	102	122	j return
74	sw \$ra, 0(\$sp)	232	2	126 return:	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant	228	3	130	addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1	224	102	134	lw \$s0, 0(\$sp)
		220	1	138	addi \$sp, \$sp, 4
		216	3	142	jr \$ra
		212	102		
		208			

Inst. Addr.	Instruction			Inst. Addr.	Instruction
		s0	s1		
		3	0		
32	.globl main	a0	v0	86	addi \$sp, \$sp, -4
36	.text	0	0	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3			94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg		\$t1=1	blt passes	
48	jal factorial	stack add.	value	98	jal factorial
52	move \$s1, \$v0 #ret.	256		102	lw \$a0, 0(\$sp)
56	li \$v0, 10	252	3	106	addi \$sp, \$sp, 4
60	syscall	248	52	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
64 factorial:	addi \$sp, \$sp, -4	244	3	114	j return
68	sw \$s0, 0(\$sp)	240	3	118 else1:	li \$v0, 1
72	addi \$sp, \$sp, -4	236	102	122	j return
74	sw \$ra, 0(\$sp)	232	2	126 return:	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant	228	3	130	addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1	224	102	134	lw \$s0, 0(\$sp)
		220	1	138	addi \$sp, \$sp, 4
		216	3	142	jr \$ra
		212	102		
		208			

Inst. Addr.	Instruction			Inst. Addr.	Instruction
		s0	s1		
		3	0		
32	.globl main	a0	v0	86	addi \$sp, \$sp, -4
36	.text	0	1	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3			94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg		\$t1=1	98	jal factorial
48	jal factorial	stack add.	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.	256		106	addi \$sp, \$sp, 4
56	li \$v0, 10	252	3	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	248	52	114	j return
64 factorial:	addi \$sp, \$sp, -4	244	3	118	li \$v0, 1
68	sw \$s0, 0(\$sp)	240	3	else1:	
72	addi \$sp, \$sp, -4	236	102	122	j return
74	sw \$ra, 0(\$sp)	232	2	126	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant	228	3	return:	
82 if1:	blt \$a0, \$t1, else1	224	102	130	addi \$sp, \$sp, 4
		220	1	134	lw \$s0, 0(\$sp)
		216	3	138	addi \$sp, \$sp, 4
		212	102	142	jr \$ra
		208			

Inst. Addr.	Instruction			Inst. Addr.	Instruction
		s0	s1		
		3	0		
32	.globl main	a0	v0	86	addi \$sp, \$sp, -4
36	.text	0	1	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3			94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg		\$t1=1	98	jal factorial
48	jal factorial	stack add.	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.	256		106	addi \$sp, \$sp, 4
56	li \$v0, 10	252	3	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	248	52	114	j return
64 factorial:	addi \$sp, \$sp, -4	244	3	118	li \$v0, 1
68	sw \$s0, 0(\$sp)	240	3	else1:	
72	addi \$sp, \$sp, -4	236	102	122	j return
74	sw \$ra, 0(\$sp)	232	2	126	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant	228	3	return:	
82 if1:	blt \$a0, \$t1, else1	224	102	130	addi \$sp, \$sp, 4
		220	1	134	lw \$s0, 0(\$sp)
		216	3	138	addi \$sp, \$sp, 4
		212	102	142	jr \$ra
		208			

Inst. Addr.	Instruction			Inst. Addr.	Instruction
		s0	s1		
		3	0		
32	.globl main	a0	v0	86	addi \$sp, \$sp, -4
36	.text	0	1	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3			94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg		\$t1=1	98	jal factorial
48	jal factorial	stack add.	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.	256		106	addi \$sp, \$sp, 4
56	li \$v0, 10	252	3	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	248	52	114	j return
64 factorial:	addi \$sp, \$sp, -4	244	3	118	li \$v0, 1
68	sw \$s0, 0(\$sp)	240	3	else1:	
72	addi \$sp, \$sp, -4	236	102	122	j return
74	sw \$ra, 0(\$sp)	232	2	126 return:	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant	228	3	130	addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1	224	102	134	lw \$s0, 0(\$sp)
		220	1	138	addi \$sp, \$sp, 4
		216	3	142	jr \$ra
		212	102		
		208			

Inst. Addr.	Instruction			Inst. Addr.	Instruction
		s0	s1		
32	.globl main	3	0	86	addi \$sp, \$sp, -4
36	.text	a0	v0	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3	0	1	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg		\$t1=1	98	jal factorial
48	jal factorial		stack add.	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.	256		106	addi \$sp, \$sp, 4
56	li \$v0, 10	252	3	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	248	52	114	j return
64 factorial:	addi \$sp, \$sp, -4	244	3	118	li \$v0, 1
68	sw \$s0, 0(\$sp)	240	3	else1:	
72	addi \$sp, \$sp, -4	236	102	122	j return
74	sw \$ra, 0(\$sp)	232	2	126 return:	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant	228	3	130	addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1	224	102	134	lw \$s0, 0(\$sp)
		220	1	138	addi \$sp, \$sp, 4
		216	3	142	jr \$ra
		212	102		
		208			

Inst. Addr.	Instruction			Inst. Addr.	Instruction
		s0	s1		
		3	0		
32	.globl main	a0	v0	86	addi \$sp, \$sp, -4
36	.text	0	1	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3			94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg		\$t1=1	98	jal factorial
48	jal factorial	stack add.	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.	256		106	addi \$sp, \$sp, 4
56	li \$v0, 10	252	3	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	248	52	114	j return
64 factorial:	addi \$sp, \$sp, -4	244	3	118	li \$v0, 1
68	sw \$s0, 0(\$sp)	240	3	else1:	
72	addi \$sp, \$sp, -4	236	102	122	j return
74	sw \$ra, 0(\$sp)	232	2	126 return:	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant	228	3	130	addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1	224	102	134	lw \$s0, 0(\$sp)
		220	1	138	addi \$sp, \$sp, 4
		216	3	142	jr \$ra
		212	102		
		208			

Inst. Addr.	Instruction		s0	s1		Inst. Addr.	Instruction
32	.globl main		3	0		86	addi \$sp, \$sp, -4
36	.text		a0	v0	ra	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		0	1	102	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			\$t1=1		98	jal factorial
48	jal factorial			stack	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.			add.		106	addi \$sp, \$sp, 4
56	li \$v0, 10					110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall					114	j return
64 factorial:	addi \$sp, \$sp, -4					118	li \$v0, 1
68	sw \$s0, 0(\$sp)					else1:	
72	addi \$sp, \$sp, -4					122	j return
74	sw \$ra, 0(\$sp)					126 return:	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant					130	addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1					134	lw \$s0, 0(\$sp)
						138	addi \$sp, \$sp, 4
						142	jr \$ra #PC=\$ra
			\$sp	220	1		
				216	3		
				212	102		
				208			
					PC		

Inst. Addr.	Instruction		s0	s1		Inst. Addr.	Instruction
32	.globl main		3	0		86	addi \$sp, \$sp, -4
36	.text		a0	v0	ra	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		0	1	102	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			\$t1=1		98	jal factorial
48	jal factorial			stack	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.			add.		106	addi \$sp, \$sp, 4
56	li \$v0, 10					110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall					114	j return
64 factorial:	addi \$sp, \$sp, -4					118	li \$v0, 1
68	sw \$s0, 0(\$sp)					else1:	
72	addi \$sp, \$sp, -4					122	j return
74	sw \$ra, 0(\$sp)					126 return:	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant					130	addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1					134	lw \$s0, 0(\$sp)
						138	addi \$sp, \$sp, 4
						142	jr \$ra #PC=\$ra

Inst. Addr.	Instruction			Inst. Addr.	Instruction
		s0	s1		
		3	0		
32	.globl main	a0	v0	86	addi \$sp, \$sp, -4
36	.text	1	1	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3			94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg		\$t1=1	98	jal factorial
48	jal factorial	stack add.	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.	256		106	addi \$sp, \$sp, 4
56	li \$v0, 10	252	3	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	248	52	114	j return
64 factorial:	addi \$sp, \$sp, -4	244	3	118	li \$v0, 1
68	sw \$s0, 0(\$sp)	240	3	else1:	
72	addi \$sp, \$sp, -4	236	102	122	j return
74	sw \$ra, 0(\$sp)	232	2	126 return:	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant	228	3	130	addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1	224	102	134	lw \$s0, 0(\$sp)
		\$sp	220	138	addi \$sp, \$sp, 4
			216	142	jr \$ra #PC=\$ra
			212		
			208		

Inst. Addr.	Instruction			Inst. Addr.	Instruction
		s0	s1		
32	.globl main	3	0	86	addi \$sp, \$sp, -4
36	.text	a0	v0	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3	1	1	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg		\$t1=1	98	jal factorial
48	jal factorial		stack add.	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.	256		106	addi \$sp, \$sp, 4
56	li \$v0, 10	252	3	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	248	52	114	j return
64 factorial:	addi \$sp, \$sp, -4	244	3	118	li \$v0, 1
68	sw \$s0, 0(\$sp)	240	3	else1:	
72	addi \$sp, \$sp, -4	236	102	122	j return
74	sw \$ra, 0(\$sp)	232	2	126	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant	228	3	return:	
82 if1:	blt \$a0, \$t1, else1	224	102	130	addi \$sp, \$sp, 4
		220	1	134	lw \$s0, 0(\$sp)
		216	3	138	addi \$sp, \$sp, 4
		212	102	142	jr \$ra #PC=\$ra
		208			

Inst. Addr.	Instruction			Inst. Addr.	Instruction
		s0	s1		
32	.globl main	3	0	86	addi \$sp, \$sp, -4
36	.text	a0	v0	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3	1	1	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg		\$t1=1	98	jal factorial
48	jal factorial		stack add.	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.		value	106	addi \$sp, \$sp, 4
56	li \$v0, 10	256		110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	252	3	114	j return
64 factorial:	addi \$sp, \$sp, -4	248	52	118	li \$v0, 1
68	sw \$s0, 0(\$sp)	244	3	else1:	
72	addi \$sp, \$sp, -4	240	3	122	j return
74	sw \$ra, 0(\$sp)	236	102	126 return:	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant	232	2	130	addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1	228	3	134	lw \$s0, 0(\$sp)
		224	102	138	addi \$sp, \$sp, 4
		220	1	142	jr \$ra #PC=\$ra
		216	3		
		212	102		
		208			

Inst. Addr.	Instruction		s0	s1		Inst. Addr.	Instruction
32	.globl main		3	0		86	addi \$sp, \$sp, -4
36	.text		a0	v0	ra	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		1	1	102	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			\$t1=1		98	jal factorial
48	jal factorial			stack	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.			add.		106	addi \$sp, \$sp, 4
56	li \$v0, 10					110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall					114	j return
64 factorial:	addi \$sp, \$sp, -4					118	li \$v0, 1
68	sw \$s0, 0(\$sp)					else1:	
72	addi \$sp, \$sp, -4					122	j return
74	sw \$ra, 0(\$sp)					126 return:	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant					130	addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1					134	lw \$s0, 0(\$sp)
						138	addi \$sp, \$sp, 4
						142	jr \$ra #PC=\$ra

Inst. Addr.	Instruction		s0	s1		Inst. Addr.	Instruction
32	.globl main		3	0		86	addi \$sp, \$sp, -4
36	.text		a0	v0	ra	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		1	1	102	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			\$t1=1		98	jal factorial
48	jal factorial			stack	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.			add.		106	addi \$sp, \$sp, 4
56	li \$v0, 10			256		110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall			252	3	114	j return
64 factorial:	addi \$sp, \$sp, -4			248	52	118	li \$v0, 1
68	sw \$s0, 0(\$sp)			244	3	else1:	
72	addi \$sp, \$sp, -4			240	3	122	j return
74	sw \$ra, 0(\$sp)			236	102	126	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant			232	2	return:	
82 if1:	blt \$a0, \$t1, else1			228	3	130	addi \$sp, \$sp, 4
				224	102	134	lw \$s0, 0(\$sp)
				220	1	138	addi \$sp, \$sp, 4
				216	3	142	jr \$ra #PC=\$ra
				212	102		
				208			

Inst. Addr.	Instruction		s0	s1		Inst. Addr.	Instruction
32	.globl main		3	0		86	addi \$sp, \$sp, -4
36	.text		a0	v0	ra	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		1	1	102	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			\$t1=1		98	jal factorial
48	jal factorial			stack	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.			add.		106	addi \$sp, \$sp, 4
56	li \$v0, 10					110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall					114	j return
64 factorial:	addi \$sp, \$sp, -4					118	li \$v0, 1
68	sw \$s0, 0(\$sp)					else1:	
72	addi \$sp, \$sp, -4					122	j return
74	sw \$ra, 0(\$sp)					126 return:	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant					130	addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1					134	lw \$s0, 0(\$sp)
						138	addi \$sp, \$sp, 4
						142	jr \$ra #PC=\$ra

Inst. Addr.	Instruction		s0	s1		Inst. Addr.	Instruction
32	.globl main		3	0		86	addi \$sp, \$sp, -4
36	.text		a0	v0	ra	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		1	1	102	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			\$t1=1		98	jal factorial
48	jal factorial			stack	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.			add.		106	addi \$sp, \$sp, 4
56	li \$v0, 10					110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall					114	j return
64 factorial:	addi \$sp, \$sp, -4					118	li \$v0, 1
68	sw \$s0, 0(\$sp)					else1:	
72	addi \$sp, \$sp, -4					122	j return
74	sw \$ra, 0(\$sp)					126 return:	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant					130	addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1					134	lw \$s0, 0(\$sp)
						138	addi \$sp, \$sp, 4
						142	jr \$ra #PC=\$ra

Inst. Addr.	Instruction		s0	s1		Inst. Addr.	Instruction
32	.globl main		3	0		86	addi \$sp, \$sp, -4
36	.text		a0	v0	ra	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		1	1	102	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			\$t1=1		98	jal factorial
48	jal factorial			stack	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.			add.		106	addi \$sp, \$sp, 4
56	li \$v0, 10					110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall					114	j return
64 factorial:	addi \$sp, \$sp, -4					118	li \$v0, 1
68	sw \$s0, 0(\$sp)		\$sp	256		else1:	
72	addi \$sp, \$sp, -4		→	252	3		
74	sw \$ra, 0(\$sp)			248	52		
78	li \$t1, 1 #constant			244	3		
82 if1:	blt \$a0, \$t1, else1			240	3		
				236	102		
				232	2		
				228	3		
				224	102		
				220	1		
				216	3		
				212	102		
				208		PC	→
						142	jr \$ra #PC=\$ra

Inst. Addr.	Instruction		s0	s1		Inst. Addr.	Instruction
32	.globl main		3	0		86	addi \$sp, \$sp, -4
36	.text		a0	v0	ra	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		1	1	102	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			\$t1=1		98	jal factorial
48	jal factorial			stack add.	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.			256		106	addi \$sp, \$sp, 4
56	li \$v0, 10			252	3	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall			248	52	114	j return
64 factorial:	addi \$sp, \$sp, -4			244	3	118	li \$v0, 1
68	sw \$s0, 0(\$sp)		\$sp	240	3	else1:	
72	addi \$sp, \$sp, -4		→	236	102	122	j return
74	sw \$ra, 0(\$sp)			232	2	126	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant			228	3	return:	
82 if1:	blt \$a0, \$t1, else1			224	102	130	addi \$sp, \$sp, 4
				220	1	134	lw \$s0, 0(\$sp)
				216	3	138	addi \$sp, \$sp, 4
				212	102	142	jr \$ra #PC=\$ra
				208			

Inst. Addr.	Instruction		s0	s1		Inst. Addr.	Instruction
32	.globl main		3	0		86	addi \$sp, \$sp, -4
36	.text		a0	v0	ra	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		2	1	102	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			\$t1=1		98	jal factorial
48	jal factorial			stack	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.			add.		106	addi \$sp, \$sp, 4
56	li \$v0, 10					110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall					114	j return
64 factorial:	addi \$sp, \$sp, -4					118	li \$v0, 1
68	sw \$s0, 0(\$sp)		\$sp	256		else1:	
72	addi \$sp, \$sp, -4		→	252	3	122	j return
74	sw \$ra, 0(\$sp)			248	52	126	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant			244	3	return:	
82 if1:	blt \$a0, \$t1, else1			240	3	130	addi \$sp, \$sp, 4
				236	102	134	lw \$s0, 0(\$sp)
				232	2	138	addi \$sp, \$sp, 4
				228	3	142	jr \$ra #PC=\$ra
				224	102		
				220	1		
				216	3		
				212	102		
				208			

Inst. Addr.	Instruction		s0	s1		Inst. Addr.	Instruction
32	.globl main		3	0		86	addi \$sp, \$sp, -4
36	.text		a0	v0	ra	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		2	1	102	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			\$t1=1		98	jal factorial
48	jal factorial			stack	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.			add.		106	addi \$sp, \$sp, 4
56	li \$v0, 10			256		110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall			252	3	114	j return
64 factorial:	addi \$sp, \$sp, -4			248	52	118	li \$v0, 1
68	sw \$s0, 0(\$sp)		\$sp	244	3	else1:	
72	addi \$sp, \$sp, -4		→	240	3	122	j return
74	sw \$ra, 0(\$sp)			236	102	126	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant			232	2	return:	
82 if1:	blt \$a0, \$t1, else1			228	3	130	addi \$sp, \$sp, 4
				224	102	134	lw \$s0, 0(\$sp)
				220	1	138	addi \$sp, \$sp, 4
				216	3	142	jr \$ra #PC=\$ra
				212	102		
				208			

Inst. Addr.	Instruction			Inst. Addr.	Instruction
		s0	s1		
32	.globl main	3	0	86	addi \$sp, \$sp, -4
36	.text	a0	v0	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3	2	2	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg		\$t1=1	98	jal factorial
48	jal factorial		stack add.	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.		value	106	addi \$sp, \$sp, 4
56	li \$v0, 10	256		110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	252	3	114	j return
64 factorial:	addi \$sp, \$sp, -4	248	52	118	li \$v0, 1
68	sw \$s0, 0(\$sp)	244	3	else1:	
72	addi \$sp, \$sp, -4	240	3	122	j return
74	sw \$ra, 0(\$sp)	236	102	126	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant	232	2	return:	
82 if1:	blt \$a0, \$t1, else1	228	3	130	addi \$sp, \$sp, 4
		224	102	134	lw \$s0, 0(\$sp)
		220	1	138	addi \$sp, \$sp, 4
		216	3	142	jr \$ra #PC=\$ra
		212	102		
		208			

Inst. Addr.	Instruction		s0	s1		Inst. Addr.	Instruction
32	.globl main		3	0		86	addi \$sp, \$sp, -4
36	.text		a0	v0	ra	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		2	2	102	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			\$t1=1		98	jal factorial
48	jal factorial			stack add.	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.			256		106	addi \$sp, \$sp, 4
56	li \$v0, 10			252	3	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall			248	52	114	j return
64 factorial:	addi \$sp, \$sp, -4			244	3	118	li \$v0, 1
68	sw \$s0, 0(\$sp)		\$sp	240	3	else1:	
72	addi \$sp, \$sp, -4		→	236	102	122	j return
74	sw \$ra, 0(\$sp)			232	2	126	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant			228	3	return:	
82 if1:	blt \$a0, \$t1, else1			224	102	130	addi \$sp, \$sp, 4
				220	1	134	lw \$s0, 0(\$sp)
				216	3	138	addi \$sp, \$sp, 4
				212	102	142	jr \$ra #PC=\$ra
				208			

Inst. Addr.	Instruction			Inst. Addr.	Instruction
		s0	s1		
		3	0		
32	.globl main	a0	v0	86	addi \$sp, \$sp, -4
36	.text	2	2	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3			94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg		\$t1=1	98	jal factorial
48	jal factorial		stack add.	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.		value	106	addi \$sp, \$sp, 4
56	li \$v0, 10	256		110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	252	3	114	j return
64 factorial:	addi \$sp, \$sp, -4	248	52	118	li \$v0, 1
68	sw \$s0, 0(\$sp)	244	3	else1:	
72	addi \$sp, \$sp, -4	240	3	122	j return
74	sw \$ra, 0(\$sp)	236	102	126	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant	232	2	return:	
82 if1:	blt \$a0, \$t1, else1	228	3	130	addi \$sp, \$sp, 4
		224	102	134	lw \$s0, 0(\$sp)
		220	1	138	addi \$sp, \$sp, 4
		216	3	142	jr \$ra #PC=\$ra
		212	102		
		208			

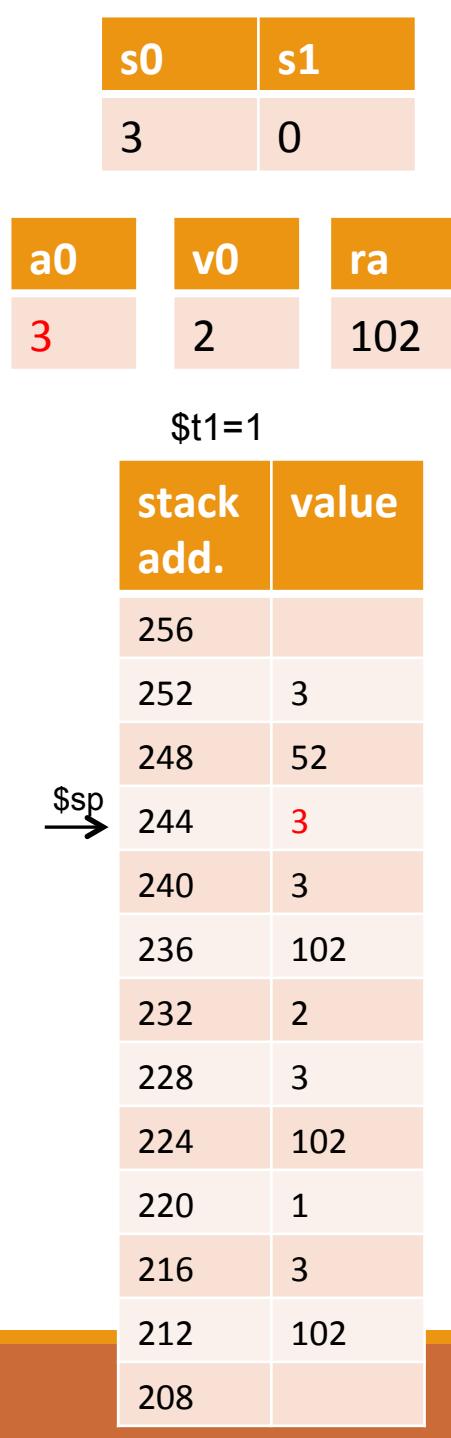
Inst. Addr.	Instruction			Inst. Addr.	Instruction
		s0	s1		
		3	0		
32	.globl main			86	addi \$sp, \$sp, -4
36	.text	a0	v0	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3	2	2	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg		\$t1=1	98	jal factorial
48	jal factorial		stack add.	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.		value	106	addi \$sp, \$sp, 4
56	li \$v0, 10	256		110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	252	3	114	j return
64 factorial:	addi \$sp, \$sp, -4	248	52	118	li \$v0, 1
68	sw \$s0, 0(\$sp)	244	3	else1:	
72	addi \$sp, \$sp, -4	240	3	122	j return
74	sw \$ra, 0(\$sp)	236	102	126	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant	232	2	return:	
82 if1:	blt \$a0, \$t1, else1	228	3	130	addi \$sp, \$sp, 4
		224	102	134	lw \$s0, 0(\$sp)
		220	1	138	addi \$sp, \$sp, 4
		216	3	142	jr \$ra #PC=\$ra
		212	102		
		208			

Inst. Addr.	Instruction			Inst. Addr.	Instruction
		s0	s1		
		3	0		
32	.globl main			86	addi \$sp, \$sp, -4
36	.text	a0	v0	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3	2	2	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			98	jal factorial
48	jal factorial			102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.		\$t1=1	106	addi \$sp, \$sp, 4
56	li \$v0, 10		stack add.	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall		value	114	j return
64 factorial:	addi \$sp, \$sp, -4			118	li \$v0, 1
68	sw \$s0, 0(\$sp)	→ \$sp		else1:	
72	addi \$sp, \$sp, -4	240	3	122	j return
74	sw \$ra, 0(\$sp)	236	102	126	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant	232	2	return:	
82 if1:	blt \$a0, \$t1, else1	228	3	130	addi \$sp, \$sp, 4
		224	102	134	lw \$s0, 0(\$sp)
		220	1	138	addi \$sp, \$sp, 4
		216	3	142	jr \$ra #PC=\$ra
		212	102		
		208			

Inst. Addr.	Instruction		s0	s1		Inst. Addr.	Instruction
32	.globl main		3	0		86	addi \$sp, \$sp, -4
36	.text		a0	v0	ra	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3		2	2	102	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			\$t1=1		98	jal factorial
48	jal factorial			stack add.	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.			256		106	addi \$sp, \$sp, 4
56	li \$v0, 10			252	3	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall			248	52	114	j return
64 factorial:	addi \$sp, \$sp, -4		\$sp →	244	3	118	li \$v0, 1
68	sw \$s0, 0(\$sp)			240	3	else1:	
72	addi \$sp, \$sp, -4			236	102	122	j return
74	sw \$ra, 0(\$sp)			232	2	126	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant			228	3	return:	
82 if1:	blt \$a0, \$t1, else1			224	102	130	addi \$sp, \$sp, 4
				220	1	134	lw \$s0, 0(\$sp)
				216	3	138	addi \$sp, \$sp, 4
				212	102	142	jr \$ra #PC=\$ra
				208		PC →	

Inst. Addr.	Instruction			Inst. Addr.	Instruction
		s0	s1		
		3	0		
32	.globl main	a0	v0	86	addi \$sp, \$sp, -4
36	.text	2	2	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3			94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg		\$t1=1	98	jal factorial
48	jal factorial		stack add.	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.		value	106	addi \$sp, \$sp, 4
56	li \$v0, 10	256		110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	252	3	114	j return
64 factorial:	addi \$sp, \$sp, -4	248	52	118	li \$v0, 1
68	sw \$s0, 0(\$sp)	244	3	else1:	
72	addi \$sp, \$sp, -4	240	3	122	j return
74	sw \$ra, 0(\$sp)	236	102	126 return:	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant	232	2	130	addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1	228	3	134	lw \$s0, 0(\$sp)
		224	102	138	addi \$sp, \$sp, 4
		220	1	142	jr \$ra #PC=\$ra
		216	3		
		212	102		
		208			

Inst. Addr.	Instruction
32	.globl main
36	.text
40 main:	li \$s0, 3 #n=3
44	move \$a0, \$s0 #arg
48	jal factorial
52	move \$s1, \$v0 #ret.
56	li \$v0, 10
60	syscall
64 factorial:	addi \$sp, \$sp, -4
68	sw \$s0, 0(\$sp)
72	addi \$sp, \$sp, -4
74	sw \$ra, 0(\$sp)
78	li \$t1, 1 #constant
82 if1:	blt \$a0, \$t1, else1



Inst. Addr.	Instruction
86	addi \$sp, \$sp, -4
90	sw \$a0, 0(\$sp)
94	addi \$a0, \$a0, -1 #new arg. = n-1
98	jal factorial
102	lw \$a0, 0(\$sp)
106	addi \$sp, \$sp, 4
110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
114	j return
118	li \$v0, 1
else1:	
122	j return
126	lw \$ra, 0(\$sp)
return:	
130	addi \$sp, \$sp, 4
134	lw \$s0, 0(\$sp)
138	addi \$sp, \$sp, 4
142	jr \$ra #PC=\$ra

Inst. Addr.	Instruction
32	.globl main
36	.text
40 main:	li \$s0, 3 #n=3
44	move \$a0, \$s0 #arg
48	jal factorial
52	move \$s1, \$v0 #ret.
56	li \$v0, 10
60	syscall
64 factorial:	addi \$sp, \$sp, -4
68	sw \$s0, 0(\$sp)
72	addi \$sp, \$sp, -4
74	sw \$ra, 0(\$sp)
78	li \$t1, 1 #constant
82 if1:	blt \$a0, \$t1, else1

s0	s1	
3	0	
a0	v0	ra
3	2	102
$\$t1=1$		
stack add.	value	
256		
252	3	
\$sp → 248	52	
244	3	
240	3	
236	102	
232	2	
228	3	
224	102	
220	1	
216	3	
212	102	
208		

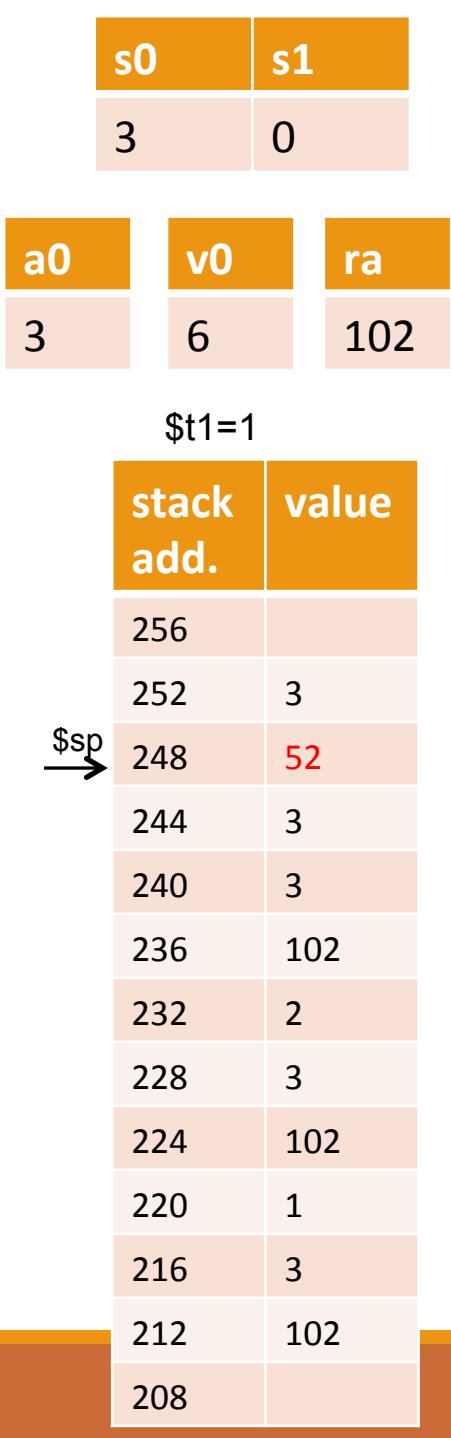
Inst. Addr.	Instruction
86	addi \$sp, \$sp, -4
90	sw \$a0, 0(\$sp)
94	addi \$a0, \$a0, -1 #new arg. = n-1
98	jal factorial
102	lw \$a0, 0(\$sp)
106	addi \$sp, \$sp, 4
110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
114	j return
118 else1:	li \$v0, 1
122	j return
126 return:	lw \$ra, 0(\$sp)
130	addi \$sp, \$sp, 4
134	lw \$s0, 0(\$sp)
138	addi \$sp, \$sp, 4
142	jr \$ra #PC=\$ra

Inst. Addr.	Instruction
32	.globl main
36	.text
40 main:	li \$s0, 3 #n=3
44	move \$a0, \$s0 #arg
48	jal factorial
52	move \$s1, \$v0 #ret.
56	li \$v0, 10
60	syscall
64 factorial:	addi \$sp, \$sp, -4
68	sw \$s0, 0(\$sp)
72	addi \$sp, \$sp, -4
74	sw \$ra, 0(\$sp)
78	li \$t1, 1 #constant
82 if1:	blt \$a0, \$t1, else1

s0	s1	
3	0	
a0	v0	ra
3	6	102
$\$t1=1$		
stack add.	value	
256		
252	3	
\$sp → 248	52	
244	3	
240	3	
236	102	
232	2	
228	3	
224	102	
220	1	
216	3	
212	102	
208		

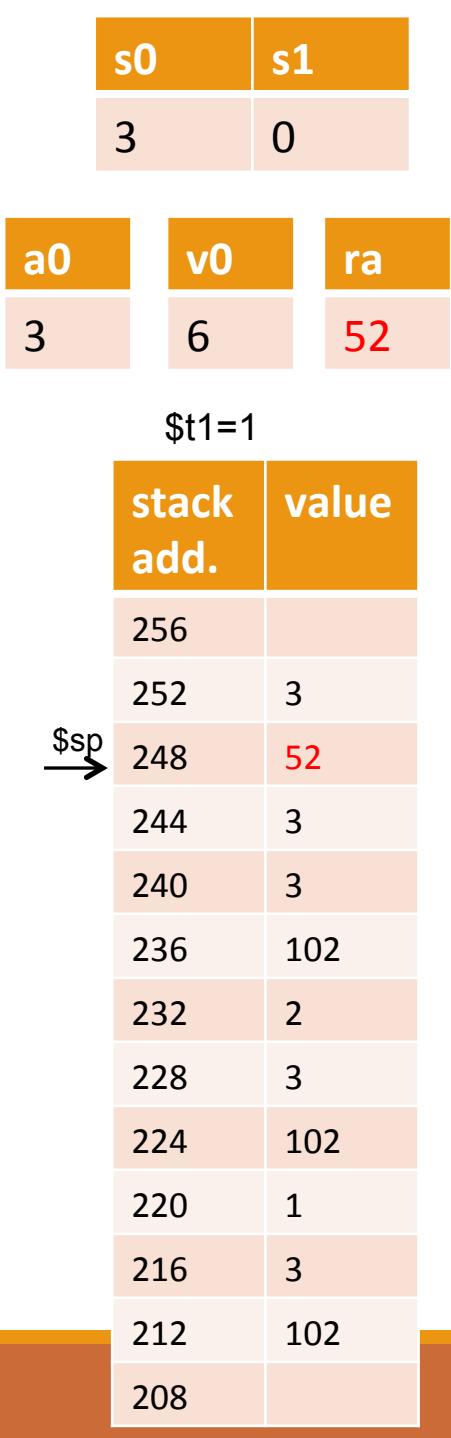
Inst. Addr.	Instruction
86	addi \$sp, \$sp, -4
90	sw \$a0, 0(\$sp)
94	addi \$a0, \$a0, -1 #new arg. = n-1
98	jal factorial
102	lw \$a0, 0(\$sp)
106	addi \$sp, \$sp, 4
110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
114	j return
118 else1:	li \$v0, 1
122	j return
126 return:	lw \$ra, 0(\$sp)
130	addi \$sp, \$sp, 4
134	lw \$s0, 0(\$sp)
138	addi \$sp, \$sp, 4
142	jr \$ra #PC=\$ra

Inst. Addr.	Instruction
32	.globl main
36	.text
40 main:	li \$s0, 3 #n=3
44	move \$a0, \$s0 #arg
48	jal factorial
52	move \$s1, \$v0 #ret.
56	li \$v0, 10
60	syscall
64 factorial:	addi \$sp, \$sp, -4
68	sw \$s0, 0(\$sp)
72	addi \$sp, \$sp, -4
74	sw \$ra, 0(\$sp)
78	li \$t1, 1 #constant
82 if1:	blt \$a0, \$t1, else1



Inst. Addr.	Instruction
86	addi \$sp, \$sp, -4
90	sw \$a0, 0(\$sp)
94	addi \$a0, \$a0, -1 #new arg. = n-1
98	jal factorial
102	lw \$a0, 0(\$sp)
106	addi \$sp, \$sp, 4
110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
114	j return
118 else1:	li \$v0, 1
122	j return
126 return:	lw \$ra, 0(\$sp)
130	addi \$sp, \$sp, 4
134	lw \$s0, 0(\$sp)
138	addi \$sp, \$sp, 4
142	jr \$ra #PC=\$ra

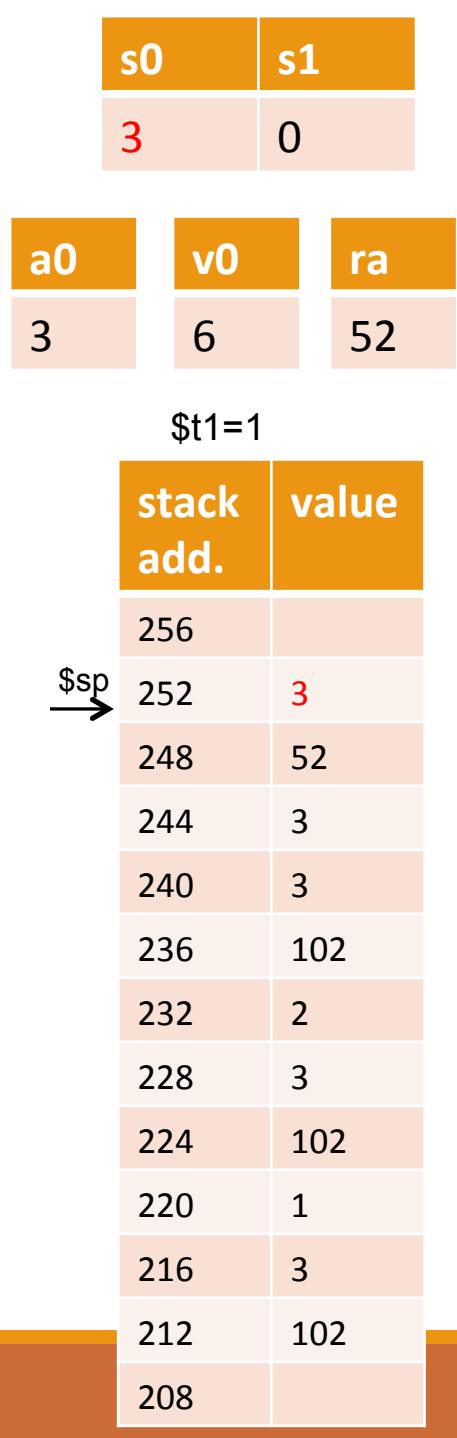
Inst. Addr.	Instruction
32	.globl main
36	.text
40 main:	li \$s0, 3 #n=3
44	move \$a0, \$s0 #arg
48	jal factorial
52	move \$s1, \$v0 #ret.
56	li \$v0, 10
60	syscall
64 factorial:	addi \$sp, \$sp, -4
68	sw \$s0, 0(\$sp)
72	addi \$sp, \$sp, -4
74	sw \$ra, 0(\$sp)
78	li \$t1, 1 #constant
82 if1:	blt \$a0, \$t1, else1



Inst. Addr.	Instruction
86	addi \$sp, \$sp, -4
90	sw \$a0, 0(\$sp)
94	addi \$a0, \$a0, -1 #new arg. = n-1
98	jal factorial
102	lw \$a0, 0(\$sp)
106	addi \$sp, \$sp, 4
110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
114	j return
118 else1:	li \$v0, 1
122	j return
126 return:	lw \$ra, 0(\$sp)
130	addi \$sp, \$sp, 4
134	lw \$s0, 0(\$sp)
138	addi \$sp, \$sp, 4
142	jr \$ra #PC=\$ra

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main	s0 3	s1 0	86	addi \$sp, \$sp, -4
36	.text	a0 3	v0 6	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3	ra 52		94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg		\$t1=1	98	jal factorial
48	jal factorial		stack add.	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.		value	106	addi \$sp, \$sp, 4
56	li \$v0, 10	256		110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	252 →	3	114	j return
64 factorial:	addi \$sp, \$sp, -4	248	52	118	li \$v0, 1
68	sw \$s0, 0(\$sp)	244	3	else1:	
72	addi \$sp, \$sp, -4	240	3	122	j return
74	sw \$ra, 0(\$sp)	236	102	126	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant	232	2	return:	
82 if1:	blt \$a0, \$t1, else1	228	3	130	addi \$sp, \$sp, 4
		224	102	134	lw \$s0, 0(\$sp)
		220	1	138	addi \$sp, \$sp, 4
		216	3	142	jr \$ra #PC=\$ra
		212	102		
		208			

Inst. Addr.	Instruction
32	.globl main
36	.text
40 main:	li \$s0, 3 #n=3
44	move \$a0, \$s0 #arg
48	jal factorial
52	move \$s1, \$v0 #ret.
56	li \$v0, 10
60	syscall
64 factorial:	addi \$sp, \$sp, -4
68	sw \$s0, 0(\$sp)
72	addi \$sp, \$sp, -4
74	sw \$ra, 0(\$sp)
78	li \$t1, 1 #constant
82 if1:	blt \$a0, \$t1, else1



Inst. Addr.	Instruction
86	addi \$sp, \$sp, -4
90	sw \$a0, 0(\$sp)
94	addi \$a0, \$a0, -1 #new arg. = n-1
98	jal factorial
102	lw \$a0, 0(\$sp)
106	addi \$sp, \$sp, 4
110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
114	j return
118 else1:	li \$v0, 1
122	j return
126 return:	lw \$ra, 0(\$sp)
130	addi \$sp, \$sp, 4
134	lw \$s0, 0(\$sp)
138	addi \$sp, \$sp, 4
142	jr \$ra #PC=\$ra

Inst. Addr.	Instruction		s0	s1		Inst. Addr.	Instruction	
32	.globl main		3	0		86	addi \$sp, \$sp, -4	
36	.text		a0	v0	ra	90	sw \$a0, 0(\$sp)	
40 main:	li \$s0, 3 #n=3		3	6	52	94	addi \$a0, \$a0, -1 #new arg. = n-1	
44	move \$a0, \$s0 #arg			\$t1=1	\$sp returns where it started	98	jal factorial	
48	jal factorial				stack add.	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.		\$sp →	256			106	addi \$sp, \$sp, 4
56	li \$v0, 10			252	3		110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall			248	52		114	j return
64 factorial:	addi \$sp, \$sp, -4			244	3		118	li \$v0, 1
68	sw \$s0, 0(\$sp)			240	3	else1:		
72	addi \$sp, \$sp, -4			236	102	122	j return	
74	sw \$ra, 0(\$sp)			232	2	126	lw \$ra, 0(\$sp)	
78	li \$t1, 1 #constant			228	3	return:		
82 if1:	blt \$a0, \$t1, else1			224	102	130	addi \$sp, \$sp, 4	
				220	1	134	lw \$s0, 0(\$sp)	
				216	3	138	addi \$sp, \$sp, 4	
				212	102	142	jr \$ra #PC=\$ra	
				208		PC →		

Inst. Addr.	Instruction	s0	s1	Inst. Addr.	Instruction
32	.globl main	3	0	86	addi \$sp, \$sp, -4
36	.text	a0	v0	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3	3	ra	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg	\$t1=1			98 jal factorial
48	jal factorial	stack add.	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.	\$sp → 256		106	addi \$sp, \$sp, 4
56	li \$v0, 10	252	3	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	248	52	114	j return
64 factorial:	addi \$sp, \$sp, -4	244	3	118 else1:	li \$v0, 1
68	sw \$s0, 0(\$sp)	240	3	122	j return
72	addi \$sp, \$sp, -4	236	102	126 return:	lw \$ra, 0(\$sp)
74	sw \$ra, 0(\$sp)	232	2	130	addi \$sp, \$sp, 4
78	li \$t1, 1 #constant	228	3	134	lw \$s0, 0(\$sp)
82 if1:	blt \$a0, \$t1, else1	224	102	138	addi \$sp, \$sp, 4
		220	1	142	jr \$ra #PC=\$ra
		216	3		
		212	102		
		208			

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main			86	addi \$sp, \$sp, -4
36	.text			90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3			94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg			98	jal factorial
48	jal factorial			102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.			106	addi \$sp, \$sp, 4
56	li \$v0, 10			110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall			114	j return
64 factorial:	addi \$sp, \$sp, -4			118	li \$v0, 1
68	sw \$s0, 0(\$sp)			else1:	
72	addi \$sp, \$sp, -4			122	j return
74	sw \$ra, 0(\$sp)			126 return:	lw \$ra, 0(\$sp)
78	li \$t1, 1 #constant			130	addi \$sp, \$sp, 4
82 if1:	blt \$a0, \$t1, else1			134	lw \$s0, 0(\$sp)
				138	addi \$sp, \$sp, 4
				142	jr \$ra #PC=\$ra

s0	s1	
3	6	
a0	v0	ra
3	6	52
\$t1=1		
stack add.	value	
\$sp → 256		
252	3	
248	52	
244	3	
240	3	
236	102	
232	2	
228	3	
224	102	
220	1	
216	3	
212	102	
208		

PC →

Inst. Addr.	Instruction	s0	s1	Inst. Addr.	Instruction
32	.globl main	3	6	86	addi \$sp, \$sp, -4
36	.text	a0	v0	90	sw \$a0, 0(\$sp)
40 main:	li \$s0, 3 #n=3	3	10	94	addi \$a0, \$a0, -1 #new arg. = n-1
44	move \$a0, \$s0 #arg	\$t1=1		98	jal factorial
48	jal factorial	stack add.	value	102	lw \$a0, 0(\$sp)
52	move \$s1, \$v0 #ret.	\$sp → 256		106	addi \$sp, \$sp, 4
56	li \$v0, 10	252	3	110	mul \$v0, \$a0, \$v0 #n*factorial(n-1)
60	syscall	248	52	114	j return
64 factorial:	addi \$sp, \$sp, -4	244	3	118 else1:	li \$v0, 1
68	sw \$s0, 0(\$sp)	240	3	122	j return
72	addi \$sp, \$sp, -4	236	102	126 return:	lw \$ra, 0(\$sp)
74	sw \$ra, 0(\$sp)	232	2	130	addi \$sp, \$sp, 4
78	li \$t1, 1 #constant	228	3	134	lw \$s0, 0(\$sp)
82 if1:	blt \$a0, \$t1, else1	224	102	138	addi \$sp, \$sp, 4
		220	1	142	jr \$ra #PC=\$ra
		216	3		
		212	102		
		208			

Inst. Addr.	Instruction			Inst. Addr.	Instruction
32	.globl main		s0 3	s1 6	
36	.text		a0 3	v0 10	
40 main:	li \$s0, 3 #n=3			ra 52	
44	move \$a0, \$s0 #arg				DONE!
48	jal factorial				
52	move \$s1, \$v0 #ret.				
56	li \$v0, 10				
60	syscall				
64 factorial:	addi \$sp, \$sp, -4				
68	sw \$s0, 0(\$sp)				
72	addi \$sp, \$sp, -4				
74	sw \$ra, 0(\$sp)				
78	li \$t1, 1 #constant				
82 if1:	blt \$a0, \$t1, else1				
			stack add. \$sp → 256	value	
			252	3	
			248	52	
			244	3	
			240	3	
			236	102	
			232	2	
			228	3	
			224	102	
			220	1	
			216	3	
			212	102	
			208		