MISC: FACTORIAL IN ACTION

Computer Systems and Networks

Dr. Pallipuram
(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
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

PC →

s0	s1
0	0

a0	v0	ra
0	0	0

stack add.	value
256	
252	
248	
244	
240	
236	
232	
228	
224	
220	
216	

$sp →

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
	32	.globl main
PC →	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
0	0

a0	v0	ra
0	0	0

	stack add.	value
$sp →	256	
	252	
	248	
	244	
	240	
	236	
	232	
	228	
	224	
	220	
	216	

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
32	.globl main
36	.text
PC → 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
0	0

a0	v0	ra
0	0	0

stack add.	value
$sp → 256	
252	
248	
244	
240	
236	
232	
228	
224	
220	
216	

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
32	.globl main
36	.text
40 main:	li $s0, 3 #n=3
PC → 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	0

stack add.	value
$sp → 256	
252	
248	
244	
240	
236	
232	
228	
224	
220	
216	

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
32	.globl main
36	.text
40 main:	li $s0, 3 #n=3
44	move $a0, $s0 #arg
PC → 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	0	0

	stack add.	value
$sp →	256	
	252	
	248	
	244	
	240	
	236	
	232	
	228	
	224	
	220	
	216	

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
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
PC → 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	0	52

	stack add.	value
$sp →	256	
	252	
	248	
	244	
	240	
	236	
	232	
	228	
	224	
	220	
	216	

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
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
PC → 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	0	52

	stack add.	value
	256	
$sp →	252	
	248	
	244	
	240	
	236	
	232	
	228	
	224	
	220	
	216	

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
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)
PC → 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	0	52

	stack add.	value
	256	
$sp →	252	3
	248	
	244	
	240	
	236	
	232	
	228	
	224	
	220	
	216	

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
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
PC → 74	sw $ra, 0($sp)
78	li $t1, 1 #constant
82 if1:	blt $a0, $t1, else1

s0	s1
3	0

a0	v0	ra
3	0	52

	stack add.	value
	256	
	252	3
$sp →	248	
	244	
	240	
	236	
	232	
	228	
	224	
	220	
	216	

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
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)
PC → 78	li $t1, 1 #constant
82 if1:	blt $a0, $t1, else1

s0	s1
3	0

a0	v0	ra
3	0	52

	stack add.	value
	256	
	252	3
$sp →	248	52
	244	
	240	
	236	
	232	
	228	
	224	
	220	
	216	

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
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
PC → 82 if1:	blt $a0, $t1, else1

s0	s1
3	0

a0	v0	ra
3	0	52

$t1=1

	stack add.	value
	256	
	252	3
$sp →	248	52
	244	
	240	
	236	
	232	
	228	
	224	
	220	
	216	

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
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	0	52

$t1=1 blt fails

$sp →

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

PC →

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
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	0	52

PC →

$t1=1

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

$sp →

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
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	0	52

PC →

$t1=1

$sp →

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

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
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
2	0	52

$t1=1

$sp →

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

PC →

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
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
PC → 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
2	0	102

$t1=1

stack add.	value
256	
252	3
248	52
$sp → 244	3
240	
236	
232	
228	
224	
220	
216	

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
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
PC → 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
2	0	102

$t1=1

stack add.	value
256	
252	3
248	52
244	3
$sp → 240	
236	
232	
228	
224	
220	
216	

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
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)
PC → 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
2	0	102

$t1=1

stack add.	value
256	
252	3
248	52
244	3
$sp → 240	3
236	
232	
228	
224	
220	
216	

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
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
PC → 74	sw $ra, 0($sp)
78	li $t1, 1 #constant
82 if1:	blt $a0, $t1, else1

s0	s1
3	0

a0	v0	ra
2	0	102

$t1=1

	stack add.	value
	256	
	252	3
	248	52
	244	3
	240	3
$sp →	236	
	232	
	228	
	224	
	220	
	216	

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
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)
PC → 78	li $t1, 1 #constant
82 if1:	blt $a0, $t1, else1

s0	s1
3	0

a0	v0	ra
2	0	102

$t1=1

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

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
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
PC → 82 if1:	blt $a0, $t1, else1

s0	s1
3	0

a0	v0	ra
2	0	102

$t1=1

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

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
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
2	0	102

$t1=1 blt fails

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

	Inst. Addr.	Instruction
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
2	0	102

$t1=1

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

	Inst. Addr.	Instruction
	86	addi $sp, $sp, -4
PC →	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
2	0	102

$t1=1

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

PC →

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
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
1	0	102

$t1=1

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

	Inst. Addr.	Instruction
	86	addi $sp, $sp, -4
	90	sw $a0, 0($sp)
	94	addi $a0, $a0, -1 #new arg. = n-1
PC →	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
PC → 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
1	0	102

$t1=1

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

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
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
PC → 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
1	0	102

$t1=1

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

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
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)
PC → 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
1	0	102

$t1=1

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

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
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
PC → 74	sw $ra, 0($sp)
78	li $t1, 1 #constant
82 if1:	blt $a0, $t1, else1

s0	s1
3	0

a0	v0	ra
1	0	102

$t1=1

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

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
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)
PC → 78	li $t1, 1 #constant
82 if1:	blt $a0, $t1, else1

s0	s1
3	0

a0	v0	ra
1	0	102

$t1=1

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

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
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
PC → 82 if1:	blt $a0, $t1, else1

s0	s1
3	0

a0	v0	ra
1	0	102

$t1=1

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

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
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

PC →

a0	v0	ra
1	0	102

$t1=1 blt fails

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

$sp →

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
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
1	0	102

PC →

$t1=1

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

$sp →

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
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
1	0	102

PC →

$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	

$sp →

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
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

PC →

stack add.	value
256	
252	3
248	52
244	3
240	3
236	102
232	2
228	3
224	102
220	1
216	

$sp →

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
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
PC → 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
$sp → 220	1
216	
212	
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

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
PC → 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
$sp → 216	
212	
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

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)
PC → 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
$sp → 216	3
212	
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

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
PC → 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	3
$sp → 212	
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

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)
PC → 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	3
$sp → 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

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
PC → 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	3
$sp → 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

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 blt passes

stack add.	value
256	
252	3
248	52
244	3
240	3
236	102
232	2
228	3
224	102
220	1
216	3
212	102
208	

$sp →

PC →

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
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	1	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	3
212	102
208	

$sp →

PC →

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
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	1	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	3
212	102
208	

$sp →

PC →

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
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	1	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	3
212	102
208	

$sp →

PC →

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
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	1	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	3
212	102
208	

$sp →

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 →

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	1	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	3
212	102
208	

$sp →

PC →

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
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	1	102

$t1=1

	stack add.	value
	256	
	252	3
	248	52
	244	3
	240	3
	236	102
	232	2
	228	3
	224	102
$sp →	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

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

s0	s1
3	0

a0	v0	ra
0	1	102

$t1=1

	stack add.	value
	256	
	252	3
	248	52
	244	3
	240	3
	236	102
	232	2
	228	3
	224	102
$sp →	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
PC →	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
1	1	102

$t1=1

	stack add.	value
	256	
	252	3
	248	52
	244	3
	240	3
	236	102
	232	2
	228	3
	224	102
$sp →	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)
PC →	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
1	1	102

$t1=1

	stack add.	value
	256	
	252	3
	248	52
	244	3
	240	3
	236	102
	232	2
	228	3
$sp →	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
PC →	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
1	1	102

$t1=1

	stack add.	value
	256	
	252	3
	248	52
	244	3
	240	3
	236	102
	232	2
	228	3
$sp →	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)
PC →	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
1	1	102

$t1=1

stack add.	value
256	
252	3
248	52
244	3
240	3
236	102
232	2
228	3
$sp → 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
PC → 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
1	1	102

$t1=1

	stack add.	value
	256	
	252	3
	248	52
	244	3
	240	3
	236	102
	232	2
	228	3
$sp →	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)
PC →	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
1	1	102

$t1=1

	stack add.	value
	256	
	252	3
	248	52
	244	3
	240	3
	236	102
	232	2
$sp →	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
PC →	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
1	1	102

$t1=1

	stack add.	value
	256	
	252	3
	248	52
	244	3
	240	3
	236	102
	232	2
$sp →	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)
PC → 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
1	1	102

$t1=1

	stack add.	value
	256	
	252	3
	248	52
	244	3
	240	3
	236	102
$sp →	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

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

s0	s1
3	0

a0	v0	ra
1	1	102

$t1=1

$sp →

stack add.	value
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
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
2	1	102

$t1=1

stack add.	value
256	
252	3
248	52
244	3
240	3
236	102
$sp → 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)
PC → 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
2	1	102

$t1=1

	stack add.	value
	256	
	252	3
	248	52
	244	3
	240	3
$sp →	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
PC →	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
2	2	102

$t1=1

stack add.	value
256	
252	3
248	52
244	3
240	3
$sp → 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)
PC → 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
2	2	102

$t1=1

	stack add.	value
	256	
	252	3
	248	52
	244	3
	240	3
$sp →	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
PC →	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
2	2	102

$t1=1

	stack add.	value
	256	
	252	3
	248	52
	244	3
	240	3
$sp →	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)
PC →	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
2	2	102

$t1=1

	stack add.	value
	256	
	252	3
	248	52
	244	3
$sp →	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
PC → 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
2	2	102

$t1=1

	stack add.	value
	256	
	252	3
	248	52
	244	3
$sp →	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)
PC → 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
2	2	102

$t1=1

	stack add.	value
	256	
	252	3
	248	52
$sp →	244	3
	240	3
	236	102
	232	2
	228	3
	224	102
	220	1
	216	3
	212	102
	208	

PC →

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
2	2	102

$t1=1

	stack add.	value
	256	
	252	3
	248	52
$sp →	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
PC →	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	2	102

$t1=1

	stack add.	value
	256	
	252	3
	248	52
$sp →	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)
PC →	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	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
PC →	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)
PC →	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

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

s0	s1
3	0

a0	v0	ra
3	6	52

$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)
PC → 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	52

$t1=1

	stack add.	value
	256	
$sp →	252	3
	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

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

s0	s1
3	0

a0	v0	ra
3	6	52

$t1=1

	stack add.	value
	256	
$sp →	252	3
	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

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

s0	s1
3	0

a0	v0	ra
3	6	52

$t1=1

$sp returns where it started

	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
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
PC → 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	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	

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.
PC → 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	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	

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
PC → 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	6

a0	v0	ra
3	10	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	

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
PC → 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	6

a0	v0	ra
3	10	52

DONE!

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	

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