**Test Process Documentation**

The 6 test process are used to test 3 test cases. Test processes 1 through 5 are used to test various primitive calls such as send/receive message and request memory block while test process 6 is used to print the results to the janusROM terminal.

Since processes are not allowed to terminate in the OS, a call to receive message is used to block each process after it has completed its necessary functions to simulate process termination.

**Test Case 1: Testing send message, receive message and request memory block**

Process 1

request memory block mem\_block to use as message envelope

place 'a' in message part of message envelope

send message to process 1

block process by calling receive message

loop forever

release processor

end loop

Process 2

receive message

if sender of message is 1 and message contains 'a' then

flag test1 as ok

end if

block process by calling receive message

loop forever

release processor

end loop

**Test Case 2: Testing delayed send, getting and setting process priority and preemption**

Process 3

set process 3's priority to 1

if process 3's priority is 1 then

request memory block to use as message envelope

place 'b' in message part of message envelope

5000 msec delayed send to process 3

block process for 5 seconds by calling receive message

flag process 3 as having come out of block

if message sender is 3 and message contains 'b' then

set process 3's priority back to 3

if process 3's priority is 3 then

flag test2 as ok

end if

end if

end if

block proces by calling receive message

loop forever

release processor

end loop

**Test Case 3: Testing proper stack management through modifications of variables and function calls**

Process 4

initialize i to 0

initialize n to 5

set process 4's priority to 2

if process 4's priority is 2 then

loop i < n

i equals add\_one(i)

end loop

end if

if i equals n

flag test3 as ok

end if

block process by calling receive message

loop forever

release processor

end loop

add\_one(num)

initialize result to num + 1

release processor

return num

Process 5

loop process 3 is not done

release processor

end loop

request memory block to use as message envelope

send message to process 6

loop forever

release processor

end loop

**Print Results:**

Process 6

receive message

if test1 is ok then

print test 1 ok

else

print test 1 fail

if test2 is ok then

print test 2 ok

else

print test 2 fail

if test3 is ok then

print test 3 ok

else

print test 3 fail

print number of tests ok / number of total tests

print number of tests fail / number of total tests

print end