

**Submit a report in PDF format. The report should have the following.**

Code in `print_ns()` function, with concise comments. Clearly mark each of the five steps in the function.

```
# function
print_ns:
    addi sp, sp, -128      # move stack pointer down 128 bytes (step 2)
    add a1, x0, sp         # store address of sp in a1

    addi sp, sp, -4        # push address of a0 to stack (step 1)
    sw a0, 0(sp)

    addi sp, sp, -4        # push return address to stack (step 1)
    sw x1, 0(sp)

    jal ra, remove_spaces  # call to function (step 3)

    lw x1, 0(sp)           # pop return address from stack
    addi sp, sp, 4

    # load result into a0 and print result (step 4)
    li a7, 4
    add a0, x0, a1
    ecall

    lw a0, 0(sp)           # pop a0 address from stack (step 5)
    addi sp, sp, 4

    addi sp, sp, 128       # move stack pointer back to original position (step 5)

    jr      ra             # exit function

# TODO
# allocate a byte array of 128 bytes on stack to save result
# call remove_spaces
# print the result string
```

A screenshot of "Run I/O" tab showing that the program works and can process multiple lines from the console. Select some input lines yourself.

```
-- program is finished running (0) --

hey hey hey
heyheyhey
yo 12345 ay ay          ay
yo12345ayayay
      2                  3
23

-- program is finished running (0) --
```

A screenshot of Data Segment window showing the stack after registers are saved on the stack, i.e., after Step 1 in `print_ns()`. Find the saved return address on the stack and write down its value and address in text.

Address	Value (+0)	Value (+4)	Value (+8)	Value (+c)	Value (+10)	Value (+14)	Value (+18)	Value (+1c)
0x7ffffef60	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00400024	0x10010000	0x00000000
0x7ffffef80	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x7ffffefa0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x7ffffefc0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x7ffffefe0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x7fffff000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x7fffff020	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x7fffff040	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x7fffff060	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x7fffff080	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x7fffff0a0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x7fffff0c0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x7fffff0e0	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x7fffff100	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x7fffff120	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000
0x7fffff140	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000	0x00000000

current sp    Hexadecimal Addresses    Hexadecimal Values    ASCII

The saved return address after step 1 is 0x00400024