Module: R2: Intro to RISC-V Assembly

Section: RISC-V ISA **Task:** RISC-V Instruction Formats

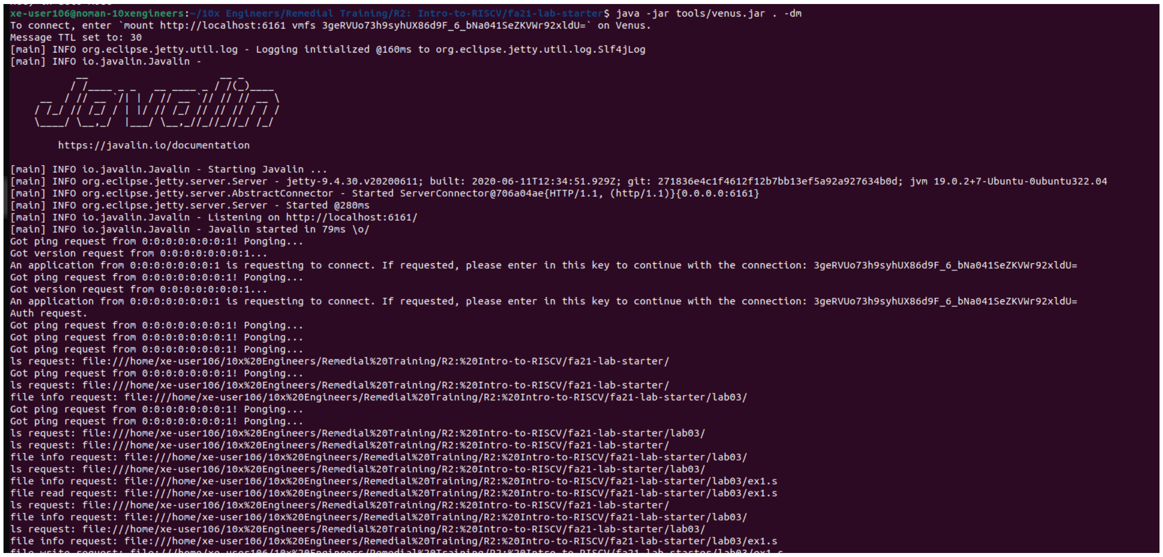
LAB 03 - https://github.com/ImNomanCR7/fa21-lab-starter.git

RISC-V Instruction Formats

Exercise 1: Connecting Files to Venus

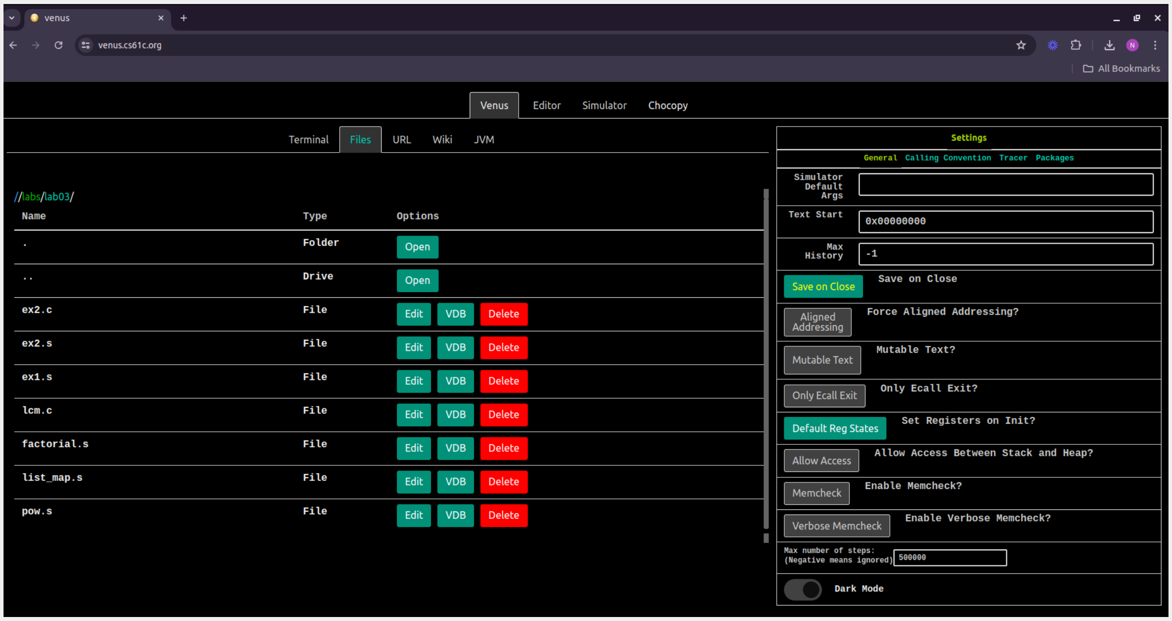
- **Code Snippet:**

```
venus .jar tools/venus.jar . -dm
```



```
xe-user106@noman-10xengineers:~/10x Engineers/Remedial Training/R2: Intro-to-RISCV/fa21-lab-starter$ java -jar tools/venus.jar . -dm
To connect, enter `mount http://localhost:6161 vmfs 3geRVUo73h9syhUX86d9F_6_bNa041SeZKVWr92xldU=` on Venus.
Message TTL set to: 30
[main] INFO org.eclipse.jetty.util.log - Logging initialized @160ms to org.eclipse.jetty.util.log.Slf4jLog
[main] INFO io.javalin.Javalin -
           __                      __ _
          / /____ _ _   __ ____ _ / /(_)____
     __  / // __ `/| | / // __ `// // // __ \
    / /_/ // /_/ / | |/ // /_/ // // // / / /
    \____/ \__,_/  |___/ \__,_//_//_//_/ /_/

        https://javalin.io/documentation

[main] INFO io.javalin.Javalin - Starting Javalin ...
[main] INFO org.eclipse.jetty.server.Server - jetty-9.4.30.v20200611; built: 2020-06-11T12:34:51.929Z; git: 271836e4c1f4612f12b7bb13ef5a92a927634b0d; jvm 19.0.2+7-Ubuntu-0ubuntu322.04
[main] INFO org.eclipse.jetty.server.AbstractConnector - Started ServerConnector@706a04ae{HTTP/1.1, (http/1.1)}{0.0.0.0:6161}
[main] INFO org.eclipse.jetty.server.Server - Started @280ms
[main] INFO io.javalin.Javalin - Listening on http://localhost:6161/
[main] INFO io.javalin.Javalin - Javalin started in 79ms \o/
Got ping request from 0:0:0:0:0:0:0:1! Ponging...
Got version request from 0:0:0:0:0:0:0:1...
An application from 0:0:0:0:0:0:0:1 is requesting to connect. If requested, please enter in this key to continue with the connection: 3geRVUo73h9syhUX86d9F_6_bNa041SeZKVWr92xldU=
Got ping request from 0:0:0:0:0:0:0:1! Ponging...
Got version request from 0:0:0:0:0:0:0:1...
An application from 0:0:0:0:0:0:0:1 is requesting to connect. If requested, please enter in this key to continue with the connection: 3geRVUo73h9syhUX86d9F_6_bNa041SeZKVWr92xldU=
Auth request.
Got ping request from 0:0:0:0:0:0:0:1! Ponging...
Got ping request from 0:0:0:0:0:0:0:1! Ponging...
Got ping request from 0:0:0:0:0:0:0:1! Ponging...
ls request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20Intro-to-RISCV/fa21-lab-starter/
Got ping request from 0:0:0:0:0:0:0:1! Ponging...
ls request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20Intro-to-RISCV/fa21-lab-starter/
file info request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20Intro-to-RISCV/fa21-lab-starter/lab03/
Got ping request from 0:0:0:0:0:0:0:1! Ponging...
Got ping request from 0:0:0:0:0:0:0:1! Ponging...
ls request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20Intro-to-RISCV/fa21-lab-starter/lab03/
ls request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20Intro-to-RISCV/fa21-lab-starter/
file info request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20Intro-to-RISCV/fa21-lab-starter/lab03/
ls request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20Intro-to-RISCV/fa21-lab-starter/lab03/
file info request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20Intro-to-RISCV/fa21-lab-starter/lab03/ex1.s
file read request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20Intro-to-RISCV/fa21-lab-starter/lab03/ex1.s
ls request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20Intro-to-RISCV/fa21-lab-starter/
file info request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20Intro-to-RISCV/fa21-lab-starter/lab03/
ls request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20Intro-to-RISCV/fa21-lab-starter/lab03/
file info request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20Intro-to-RISCV/fa21-lab-starter/lab03/ex1.s
file write request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20Intro-to-RISCV/fa21-lab-starter/lab03/ex1.s
```

Mounted Local Labs



//labs/lab03/

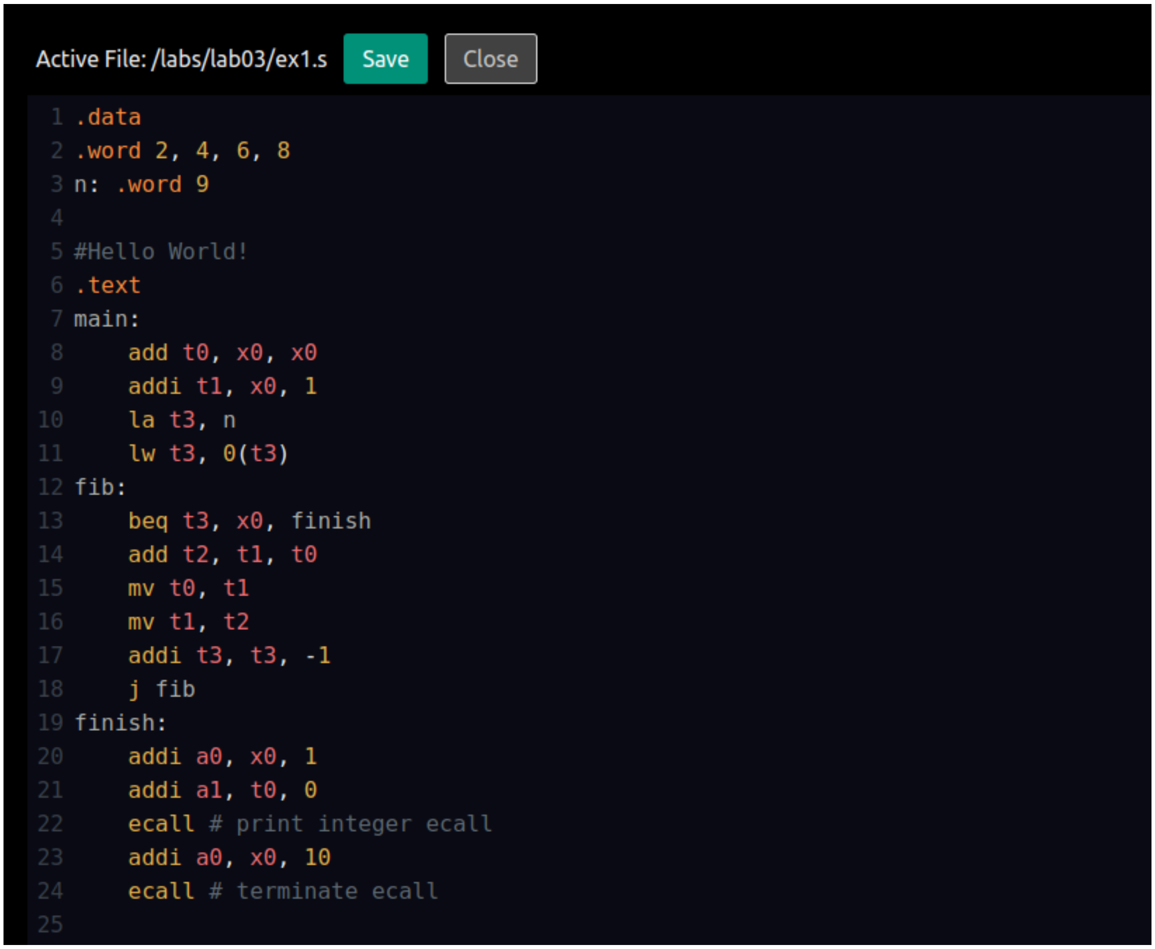
Name	Type	Options
.	Folder	Open
..	Drive	Open
ex2.c	File	Edit VDB Delete
ex2.s	File	Edit VDB Delete
ex1.s	File	Edit VDB Delete
lcm.c	File	Edit VDB Delete
factorial.s	File	Edit VDB Delete
list_map.s	File	Edit VDB Delete
pow.s	File	Edit VDB Delete

Settings

General Calling Convention Tracer Packages

Simulator Default Args	
Text Start	0x00000000
Max History	-1
Save on Close	Save on Close
Aligned Addressing	Force Aligned Addressing?
Mutable Text	Mutable Text?
Only Ecall Exit	Only Ecall Exit?
Default Reg States	Set Registers on Init?
Allow Access	Allow Access Between Stack and Heap?
Memcheck	Enable Memcheck?
Verbose Memcheck	Enable Verbose Memcheck?
Max number of steps: (Negative means ignored)	500000
	Dark Mode

Made an Edit to ex1.s:



```
.data
.word 2, 4, 6, 8
n: .word 9

#Hello World!
.text
main:
    add t0, x0, x0
    addi t1, x0, 1
    la t3, n
    lw t3, 0(t3)
fib:
    beq t3, x0, finish
    add t2, t1, t0
    mv t0, t1
    mv t1, t2
    addi t3, t3, -1
    j fib
finish:
    addi a0, x0, 1
    addi a1, t0, 0
    ecall # print integer ecall
    addi a0, x0, 10
    ecall # terminate ecall

```

Exercise 2: Familiarizing with Venus

i - Directives:

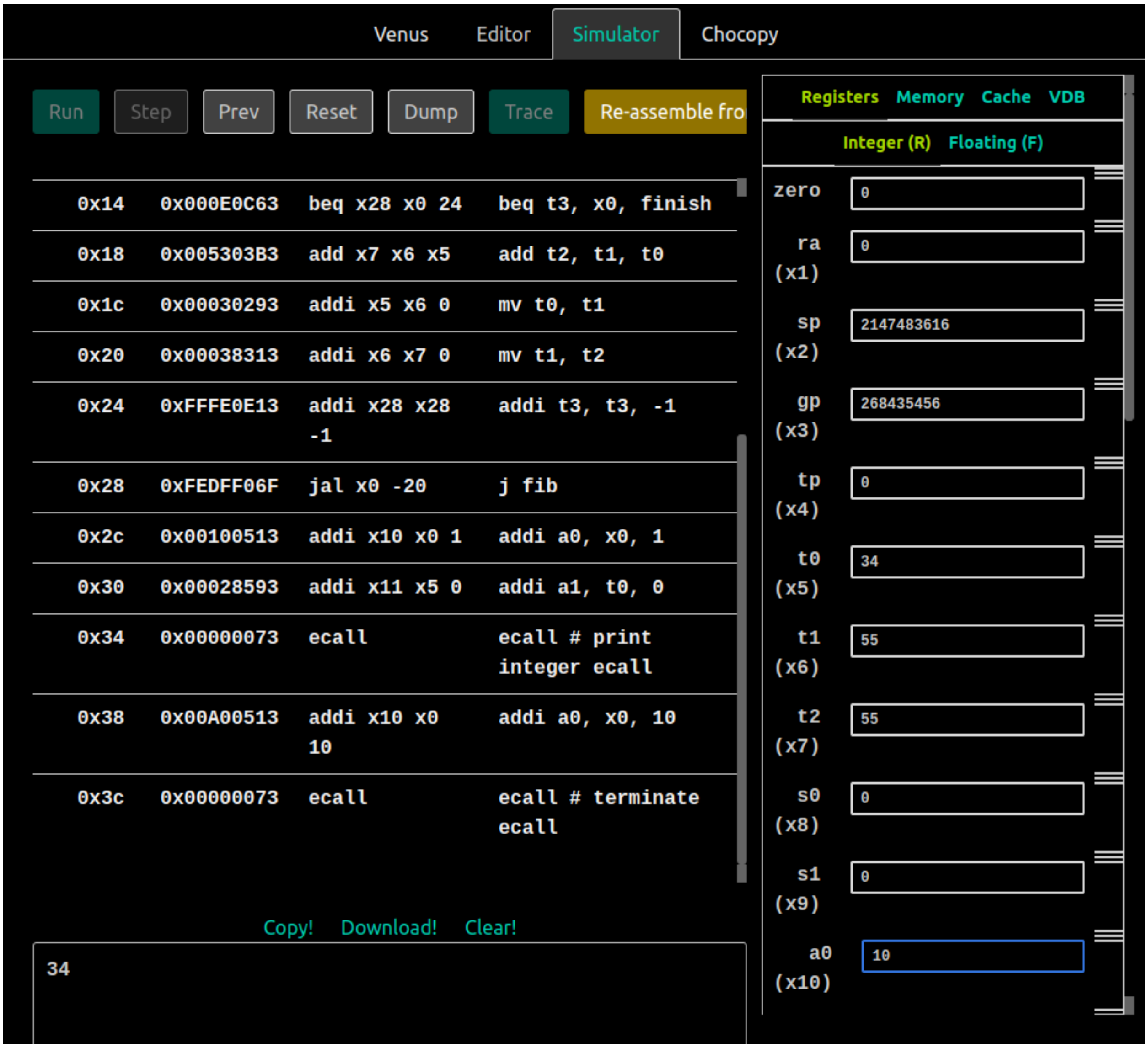
In RISC-V assembly language programming, directives like "**.data**", "**.word**", and "**.text**" are used to organize and specify different sections of memory within a program. The "**.data**" directive defines the data section, where variables and constants are initialized.

The "**.word**" directive, typically within the "**.data**"section, allocates space for 32-bit words of data. The "**.text**" directive defines the text section, containing executable code such as instructions and function definitions.

Together, these directives help structure the program's memory layout, ensuring data and code are appropriately separated and initialized for execution.

ii - Output of the program:

The program outputs the value 34. This number represents the 10th element of the Fibonacci series.



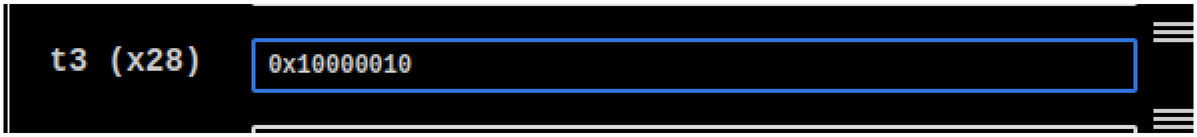
```
0x14   0x000E0C63   beq x28 x0 24      beq t3, x0, finish
0x18   0x005303B3   add x7 x6 x5       add t2, t1, t0
0x1c   0x00030293   addi x5 x6 0       mv t0, t1
0x20   0x00038313   addi x6 x7 0       mv t1, t2
0x24   0xFFFE0E13   addi x28 x28       addi t3, t3, -1
                    -1
0x28   0xFEDFF06F   jal x0 -20         j fib
0x2c   0x00100513   addi x10 x0 1      addi a0, x0, 1
0x30   0x00028593   addi x11 x5 0      addi a1, t0, 0
0x34   0x00000073   ecall              ecall # print
                                       integer ecall
0x38   0x00A00513   addi x10 x0        addi a0, x0, 10
                    10
0x3c   0x00000073   ecall              ecall # terminate
                                       ecall
```

```
34
```

zero	0
ra (x1)	0
sp (x2)	2147483616
gp (x3)	268435456
tp (x4)	0
t0 (x5)	34
t1 (x6)	55
t2 (x7)	55
s0 (x8)	0
s1 (x9)	0
a0 (x10)	10

iii - Address of n:

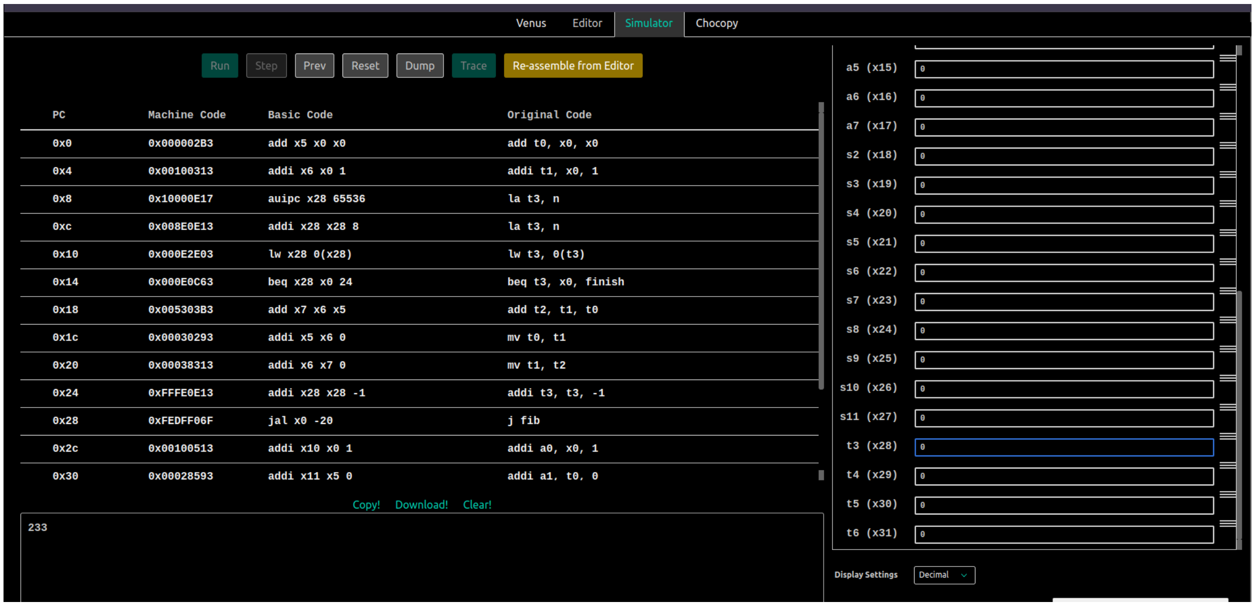
The address of n is **0x10000010** as shown by the screenshot.



t3 (x28) 0x10000010

iv - 13th fib number:

Edited the value of **t3** to 12 to print the 13th Fibonacci number in series.



PC	Machine Code	Basic Code	Original Code
0x0	0x000002B3	add x5 x0 x0	add t0, x0, x0
0x4	0x00100313	addi x6 x0 1	addi t1, x0, 1
0x8	0x10000E17	auipc x28 65536	la t3, n
0xc	0x008E0E13	addi x28 x28 8	la t3, n
0x10	0x000E2E03	lw x28 0(x28)	lw t3, 0(t3)
0x14	0x000E0C63	beq x28 x0 24	beq t3, x0, finish
0x18	0x005303B3	add x7 x6 x5	add t2, t1, t0
0x1c	0x00030293	addi x5 x6 0	mv t0, t1
0x20	0x00038313	addi x6 x7 0	mv t1, t2
0x24	0xFFFE0E13	addi x28 x28 -1	addi t3, t3, -1
0x28	0xFEDFF06F	jal x0 -20	j fib
0x2c	0x00100513	addi x10 x0 1	addi a0, x0, 1
0x30	0x00028593	addi x11 x5 0	addi a1. t0. 0

```
233
```

a5 (x15)	0
a6 (x16)	0
a7 (x17)	0
s2 (x18)	0
s3 (x19)	0
s4 (x20)	0
s5 (x21)	0
s6 (x22)	0
s7 (x23)	0
s8 (x24)	0
s9 (x25)	0
s10 (x26)	0
s11 (x27)	0
t3 (x28)	0
t4 (x29)	0
t5 (x30)	0
t6 (x31)	0

Display Settings: Decimal

Exercise 3: Translating from C to RISC-V Assembly

i. **The register representing the variable k:** Register “**t0**” is used to represent the variable “**k**”. It is initialized to 0 (“**addi t0, x0,0**”) before the loop starts and is incremented within the loop (“**addi t0, t0, 1**”).

ii. **The register representing the variable sum:** Register “**s0**” is used to represent the variable “**sum**”. It is initialized to 0 (**addi s0,x0, 0**) before the loop starts and is updated within the loop (**add s0, s0, t2**).

iii. **The registers acting as pointers to the source and dest arrays:** Registers “**s1**” and “**s2**” are used as pointers to the “**source**”and “**dest” arrays**” respectively. They are loaded with the addresses of the arrays (“**la s1, source**” and “**la s2, dest**”) before the loop starts, and they are manipulated within the loop to access elements of the arrays (**add t1, s1, s3** and **add t3, s2, s3**).

iv. **The assembly code for the loop found in the C code:** It starts with a label (**loop:**) and ends with a branch instruction (**jal x0,loop**). Inside the loop, elements of the “**source**” array are loaded (**lw t2,0(t1)**), and the loop continues until a 0 is encountered in the “**source**” array (**beq t2, x0, exit**).

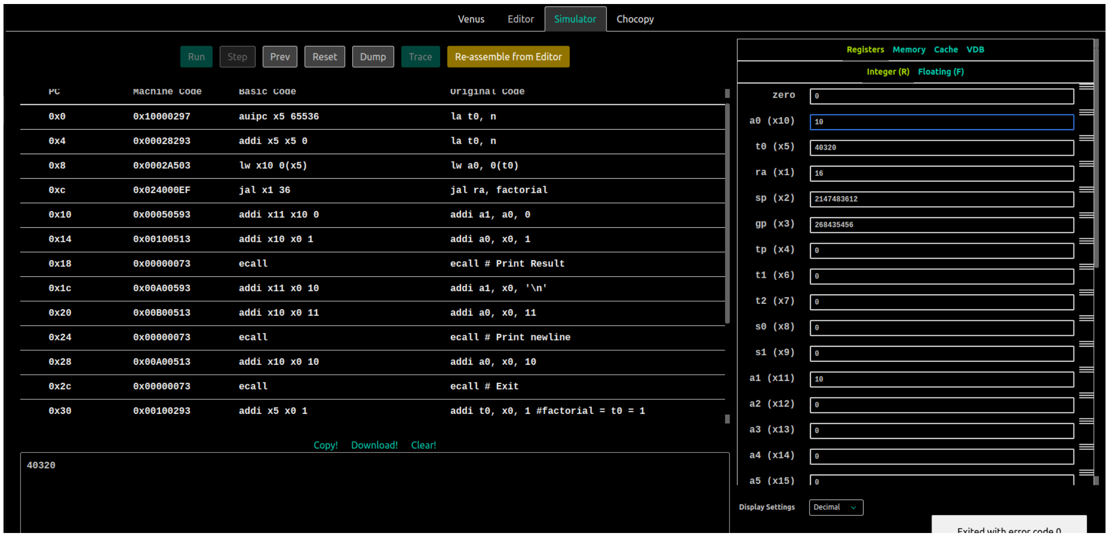
v. **How the pointers are manipulated in the assembly code:** The pointers to the source and dest arrays (**s1 and s2**) are manipulated using arithmetic operations. Within the loop, the index “**t0**” is left-shifted by2 (**slli s3, t0, 2**) to calculate the offset for accessing elements of the arrays. Then, this offset is added to the base address of each array (**add t1, s1, s3** and **add t3,s2, s3**) to obtain the address of the current element in each array.

Exercise 4: Factorial of a Number

- **Code Snippet:**

```
#Author: Noman Rafiq
#Dated: 30 June, 2024
#Description: The program uses a variable n to calculate the factorial
of that number.


.globl factorial

.data
n: .word 8

.text
main:
    la t0, n
    lw a0, 0(t0)
    jal ra, factorial

    addi a1, a0, 0
    addi a0, x0, 1
    ecall # Print Result

    addi a1, x0, '\n'
    addi a0, x0, 11
    ecall # Print newline

    addi a0, x0, 10
    ecall # Exit

factorial:
    # YOUR CODE HERE
    addi t0, x0, 1  #factorial = t0 = 1
loop:
    mul t0, t0, a0  #factorial = 1 * n
    addi a0, a0, -1 # n = n - 1
    bne a0, x0, loop
    mv a0, t0
    jr ra
```

- **Output:**



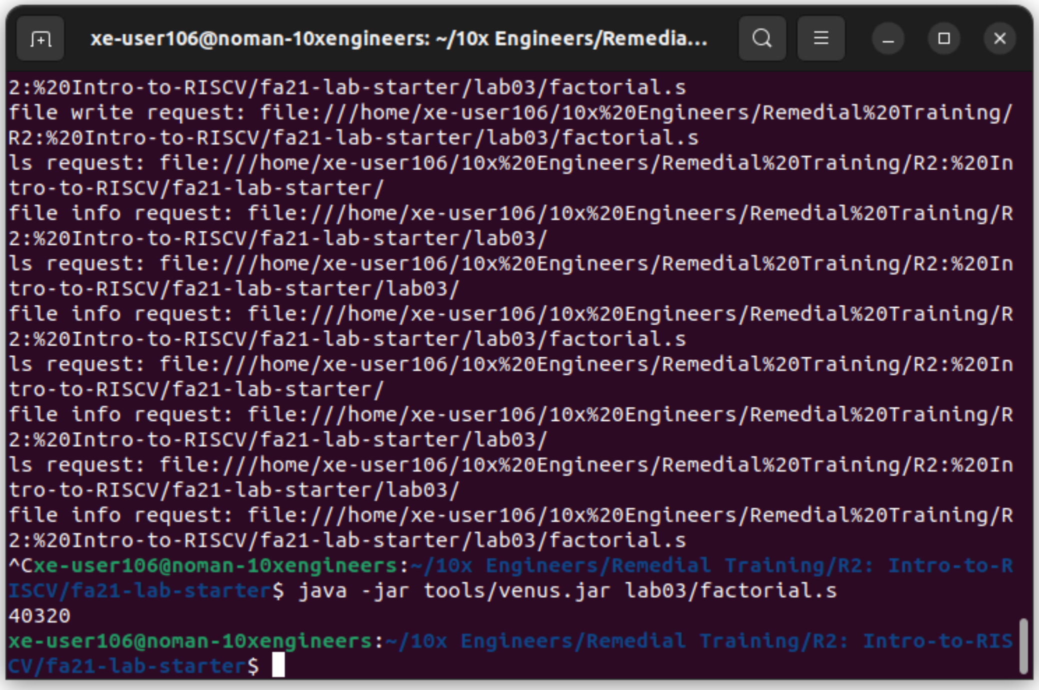
PC	Machine Code	Basic Code	Original Code
0x0	0x10000297	auipc x5 65536	la t0, n
0x4	0x00028293	addi x5 x5 0	la t0, n
0x8	0x0002A503	lw x10 0(x5)	lw a0, 0(t0)
0xc	0x024000EF	jal x1 36	jal ra, factorial
0x10	0x00050593	addi x11 x10 0	addi a1, a0, 0
0x14	0x00100513	addi x10 x0 1	addi a0, x0, 1
0x18	0x00000073	ecall	ecall # Print Result
0x1c	0x00A00593	addi x11 x0 10	addi a1, x0, '\n'
0x20	0x00B00513	addi x10 x0 11	addi a0, x0, 11
0x24	0x00000073	ecall	ecall # Print newline
0x28	0x00A00513	addi x10 x0 10	addi a0, x0, 10
0x2c	0x00000073	ecall	ecall # Exit
0x30	0x00100293	addi x5 x0 1	addi t0, x0, 1 #factorial = t0 = 1

```
40320
```

zero	0
a0 (x10)	10
t0 (x5)	40320
ra (x1)	16
sp (x2)	2147483612
gp (x3)	268435456
tp (x4)	0
t1 (x6)	0
t2 (x7)	0
s0 (x8)	0
s1 (x9)	0
a1 (x11)	10
a2 (x12)	0
a3 (x13)	0
a4 (x14)	0
a5 (x15)	0

Display Settings: Decimal

Exited with error code 0



xe-user106@noman-10xengineers: ~/10x Engineers/Remedia...

```
2:%20Intro-to-RISCV/fa21-lab-starter/lab03/factorial.s
file write request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/
R2:%20Intro-to-RISCV/fa21-lab-starter/lab03/factorial.s
ls request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20In
tro-to-RISCV/fa21-lab-starter/
file info request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R
2:%20Intro-to-RISCV/fa21-lab-starter/lab03/
ls request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20In
tro-to-RISCV/fa21-lab-starter/lab03/
file info request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R
2:%20Intro-to-RISCV/fa21-lab-starter/lab03/factorial.s
ls request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20In
tro-to-RISCV/fa21-lab-starter/
file info request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R
2:%20Intro-to-RISCV/fa21-lab-starter/lab03/
ls request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20In
tro-to-RISCV/fa21-lab-starter/lab03/
file info request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R
2:%20Intro-to-RISCV/fa21-lab-starter/lab03/factorial.s
^Cxe-user106@noman-10xengineers:~/10x Engineers/Remedial Training/R2: Intro-to-R
ISCV/fa21-lab-starter$ java -jar tools/venus.jar lab03/factorial.s
40320
xe-user106@noman-10xengineers:~/10x Engineers/Remedial Training/R2: Intro-to-RIS
CV/fa21-lab-starter$ 
```

Exercise 5: Function Calling with Map

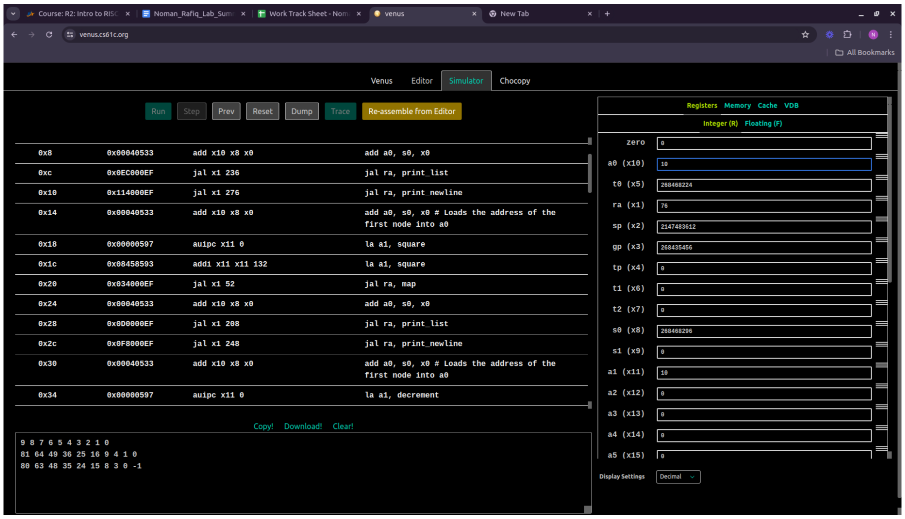
- **Code Snippet:**

```
#Author: Noman Rafiq
#Dated: 30 June, 2024
```

```
#Description: The program uses a map function that performs an in-place
upate of a linked-list by applying a called function to all the nodes in
the list.

.globl map

.text
main:
    jal ra, create_default_list
    add s0, a0, x0 # a0 (and now s0) is the head of node list

    # Print the list
    add a0, s0, x0
    jal ra, print_list
    # Print a newline
    jal ra, print_newline

    # === Calling `map(head, &square)` ===
    # Load function arguments
    add a0, s0, x0 # Loads the address of the first node into a0

    # Load the address of the "square" function into a1 (hint: check out
"la" on the green sheet)
    ### YOUR CODE HERE ###
    la a1, square


    # Issue the call to map
    jal ra, map

    # Print the squared list
    add a0, s0, x0
    jal ra, print_list
    jal ra, print_newline

    # === Calling `map(head, &decrement)` ===
    # Because our `map` function modifies the list in-place, the
decrement takes place after
    # the square does

    # Load function arguments
    add a0, s0, x0 # Loads the address of the first node into a0
    # Load the address of the "decrement" function into a1 (should be very similar to before)
    ### YOUR CODE HERE ###
    la a1, decrement

    # Issue the call to map
    jal ra, map

    # Print decremented list
    add a0, s0, x0
    jal ra, print_list
    jal ra, print_newline

    addi a0, x0, 10
    ecall # Terminate the program

map:
    # Prologue: Make space on the stack and back-up registers
    ### YOUR CODE HERE ###
    addi sp, sp, -12
    sw s0, 0(sp)
    sw s1, 4(sp)
    sw ra, 8(sp)

    beq a0, x0, done # If we were given a null pointer (address 0), we're done.

    add s0, a0, x0 # Save address of this node in s0
    add s1, a1, x0 # Save address of function in s1

    # Remember that each node is 8 bytes long: 4 for the value followed by 4 for the pointer to next.
    # What does this tell you about how you access the value and how you access the pointer to next?

    # Load the value of the current node into a0
    # THINK: Why a0?
    ### YOUR CODE HERE ###
    lw a0, 0(s0)
    # Call the function in question on that value. DO NOT use a label (be prepared to answer why).
    # Hint: Where do we keep track of the function to call? Recall the parameters of "map".
    ### YOUR CODE HERE ###
    jalr ra, s1, 0

    # Store the returned value back into the node
    # Where can you assume the returned value is?
    ### YOUR CODE HERE ###
    sw a0, 0(s0)

    # Load the address of the next node into a0
    # The address of the next node is an attribute of the current node.
    # Think about how structs are organized in memory.
    ### YOUR CODE HERE ###
    lw a0, 4(s0)

    # Put the address of the function back into a1 to prepare for the recursion
    # THINK: why a1? What about a0?
    ### YOUR CODE HERE ###
    mv a1, s1

    # Recurse
    ### YOUR CODE HERE ###
    jal ra, map

done:
    # Epilogue: Restore register values and free space from the stack
    ### YOUR CODE HERE ###
    lw s0, 0(sp)
    lw s1, 4(sp)
    lw ra, 8(sp)
    addi sp, sp, 12
    jr ra # Return to caller

# === Definition of the "square" function ===
square:
    mul a0, a0, a0
    jr ra
```

```
# === Definition of the "decrement" function ===
decrement:
    addi a0, a0, -1
    jr ra

# === Helper functions ===
# You don't need to understand these, but reading them may be useful

create_default_list:
    addi sp, sp, -12
    sw ra, 0(sp)
    sw s0, 4(sp)
    sw s1, 8(sp)
    li s0, 0            # Pointer to the last node we handled
    li s1, 0            # Number of nodes handled
loop:                   # do...
    li a0, 8
    jal ra, malloc      #     Allocate memory for the next node
    sw s1, 0(a0)        #     node->value = i
    sw s0, 4(a0)        #     node->next = last
    add s0, a0, x0      #     last = node
    addi s1, s1, 1      #     i++
    addi t0, x0, 10
    bne s1, t0, loop    # ... while i!= 10
    lw ra, 0(sp)
    lw s0, 4(sp)
    lw s1, 8(sp)
    addi sp, sp, 12
    jr ra

print_list:
    bne a0, x0, print_me_and_recurse
    jr ra               # Nothing to print
print_me_and_recurse:
    add t0, a0, x0      # t0 gets current node address
    lw a1, 0(t0)        # a1 gets value in current node
    addi a0, x0, 1      # Prepare for print integer ecall
    ecall
    addi a1, x0, ' '    # a0 gets address of string containing space
    addi a0, x0, 11     # Prepare for print char syscall
    ecall
    lw a0, 4(t0)        # a0 gets address of next node
    jal x0, print_list  # Recurse. The value of ra hasn't been changed.

print_newline:
    addi a1, x0, '\n'   # Load in ascii code for newline
    addi a0, x0, 11
    ecall
    jr ra

malloc:
    addi a1, a0, 0
    addi a0, x0, 9
    ecall
    jr ra
```

- **Output:**

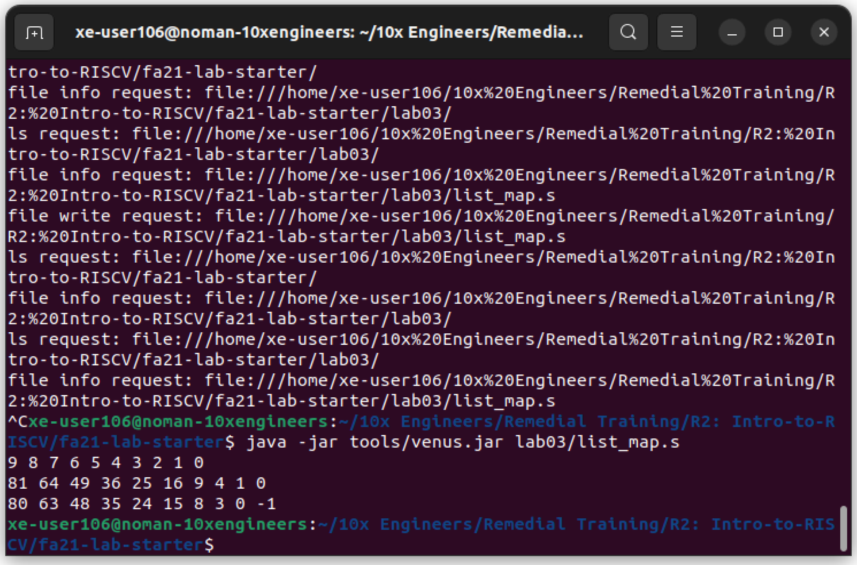


```
0x8     0x00040533   add x10 x8 x0      add a0, s0, x0
0xc     0x0EC000EF   jal x1 236         jal ra, print_list
0x10    0x114000EF   jal x1 276         jal ra, print_newline
0x14    0x00040533   add x10 x8 x0      add a0, s0, x0 # Loads the address of the
                                        first node into a0
0x18    0x00000597   auipc x11 0        la a1, square
0x1c    0x08458593   addi x11 x11 132   la a1, square
0x20    0x034000EF   jal x1 52          jal ra, map
0x24    0x00040533   add x10 x8 x0      add a0, s0, x0
0x28    0x0D0000EF   jal x1 208         jal ra, print_list
0x2c    0x0F8000EF   jal x1 248         jal ra, print_newline
0x30    0x00040533   add x10 x8 x0      add a0, s0, x0 # Loads the address of the
                                        first node into a0
0x34    0x00000597   auipc x11 0        la a1, decrement
```

```
9 8 7 6 5 4 3 2 1 0
81 64 49 36 25 16 9 4 1 0
80 63 48 35 24 15 8 3 0 -1
```

zero	0
a0 (x10)	10
t0 (x5)	268468224
ra (x1)	76
sp (x2)	2147483612
gp (x3)	268435456
tp (x4)	0
t1 (x6)	0
t2 (x7)	0
s0 (x8)	268468296
s1 (x9)	0
a1 (x11)	10
a2 (x12)	0
a3 (x13)	0
a4 (x14)	0
a5 (x15)	0

Display Settings: Decimal



```
tro-to-RISCV/fa21-lab-starter/
file info request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R
2:%20Intro-to-RISCV/fa21-lab-starter/lab03/
ls request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20In
tro-to-RISCV/fa21-lab-starter/lab03/
file info request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R
2:%20Intro-to-RISCV/fa21-lab-starter/lab03/list_map.s
file write request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/
R2:%20Intro-to-RISCV/fa21-lab-starter/lab03/list_map.s
ls request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20In
tro-to-RISCV/fa21-lab-starter/
file info request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R
2:%20Intro-to-RISCV/fa21-lab-starter/lab03/
ls request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R2:%20In
tro-to-RISCV/fa21-lab-starter/lab03/
file info request: file:///home/xe-user106/10x%20Engineers/Remedial%20Training/R
2:%20Intro-to-RISCV/fa21-lab-starter/lab03/list_map.s
^Cxe-user106@noman-10xengineers:~/10x Engineers/Remedial Training/R2: Intro-to-R
ISCV/fa21-lab-starter$ java -jar tools/venus.jar lab03/list_map.s
9 8 7 6 5 4 3 2 1 0
81 64 49 36 25 16 9 4 1 0
80 63 48 35 24 15 8 3 0 -1
xe-user106@noman-10xengineers:~/10x Engineers/Remedial Training/R2: Intro-to-RIS
CV/fa21-lab-starter$
```