Levi de Lima Pereira Junior - 121210472 Roteiro 6

```
.text
main:
    addi a0, zero, 1
    add a1, a0, a0
    addi a2, zero, 5
loop:
    beq a0, a2, fim
    slli a1, a1, 1
    addi a0, a0, 1
    j loop
fim:
    nop
```

1. Qual é a operação realizada pelo código acima?

O código faz uma soma imediata entre o valor armazenado no registrador zero (x0) que é 0 e 1 e armazena no registrador a0 (x10).

Depois ele faz uma soma entre o valor armazenado no registrador a0 (x10) e a0 (x10) e armazena em a1 (x11) que será 2.

Depois ele faz uma soma imediata entre 0 (x0) e 5 que dá 5 e armazena no registrador a2 (x12).

Em seguida ele faz uma comparação entre os valores dos registradores a0 (x10) e a2 (x12) e se for igual ele leva para o endereço fim que encerra o código, ou seja, ele pula 16 bytes. Caso ele não passe na condição ele faz um shift para a esquerda no valor do registrador a1 (x11) e armazena no registrador a1 (x11).

Em seguida, ele faz uma soma imediata entre o valor armazenado no registrador a0 (x10) e 1 e armazena o resultado no registrador a0 (x10).

Depois ele faz um jump para o endereço loop, pulando -12 bytes.

2. Qual é o conteúdo (em decimal) dos registradores a0, a1 e a2 ao final da execução do programa?

```
a0 = 5

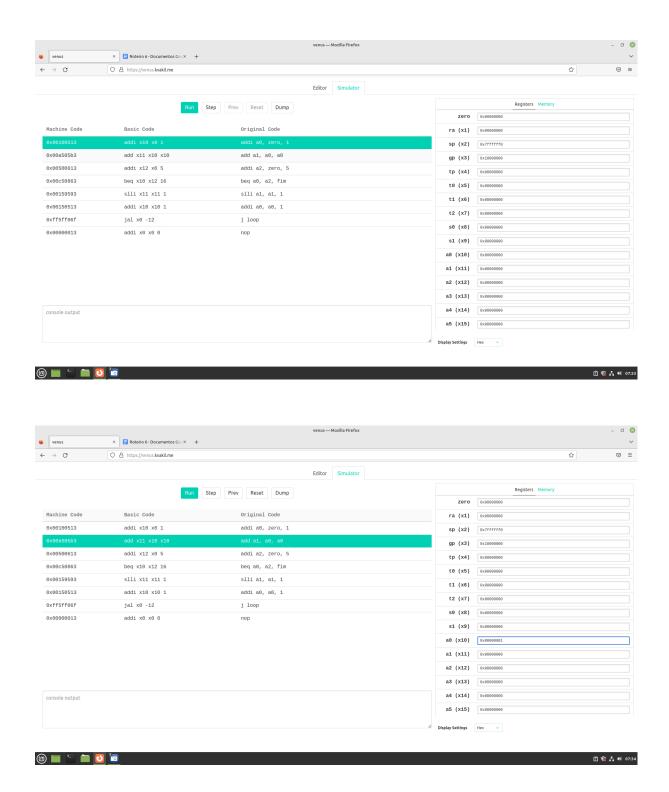
a1 = 32

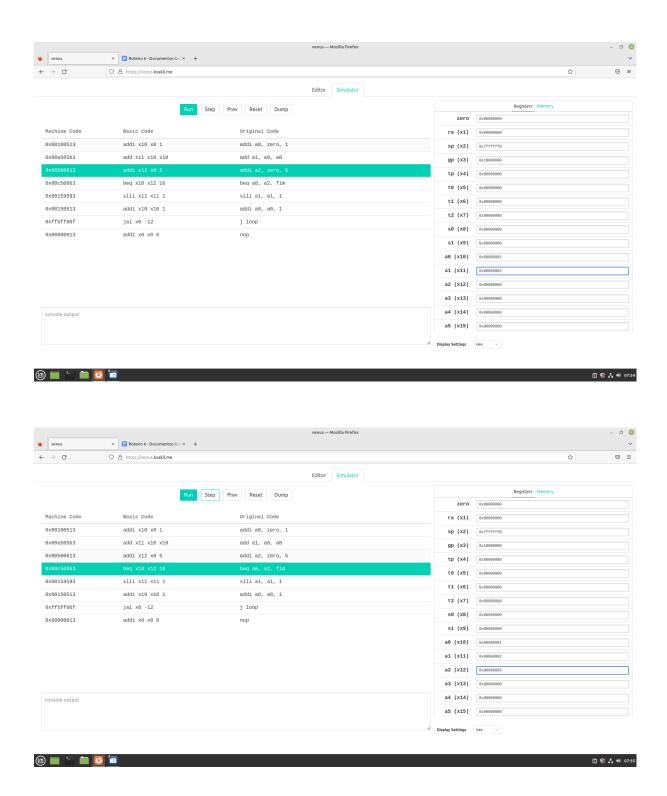
a2 = 5
```

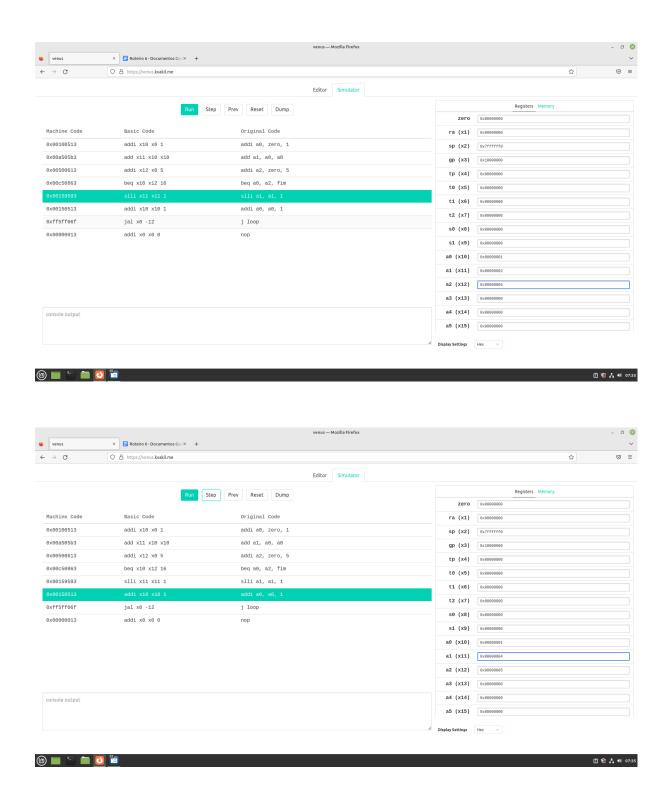
3. Quais são os valores de loop e fim?

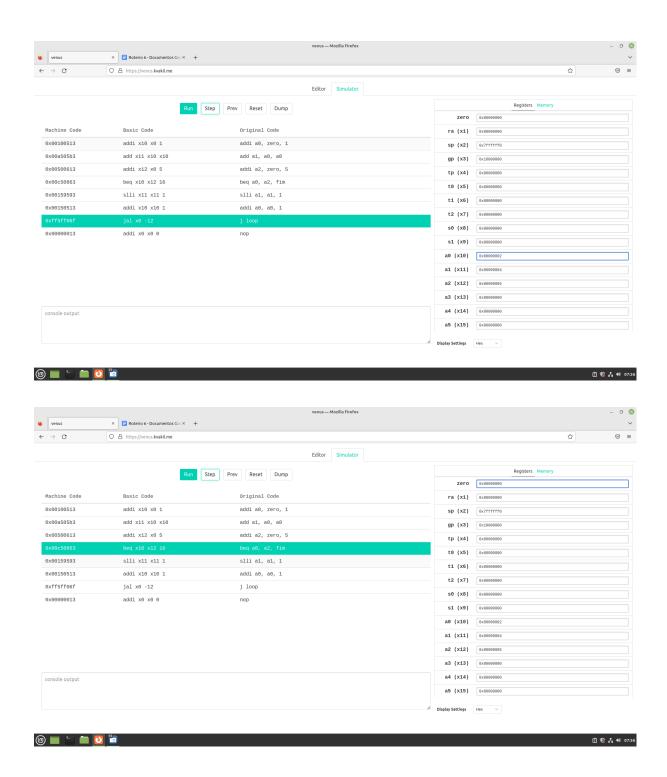
loop: 0x0000000C fim: 0x000001C

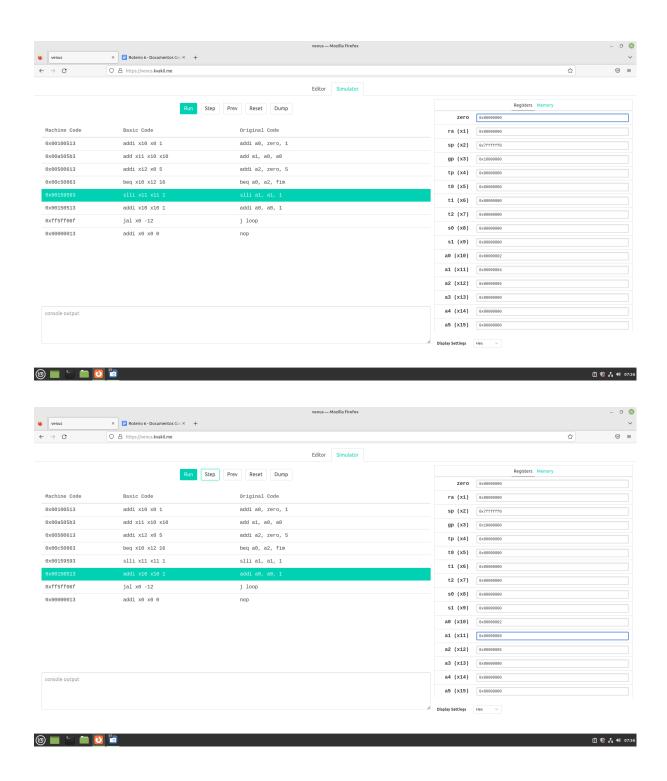
Adicione "prints" da tela do simulador ao arquivo PDF. Esses "prints" deverão mostrar todo o ambiente, incluindo o código em Assembly que foi executado e os valores exibidos na interface após a execução de cada instrução (código de máquina armazenado na memória, valores do pc e dos demais registradores envolvidos).

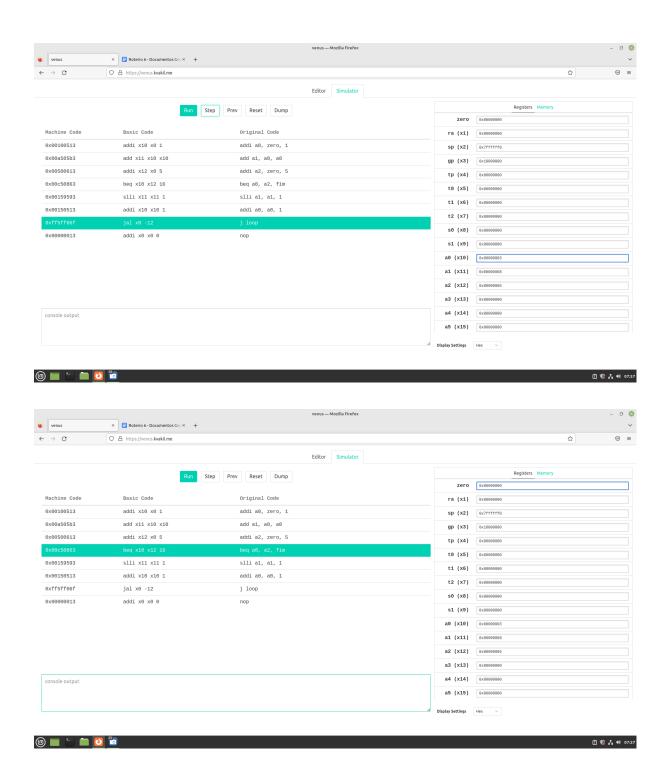


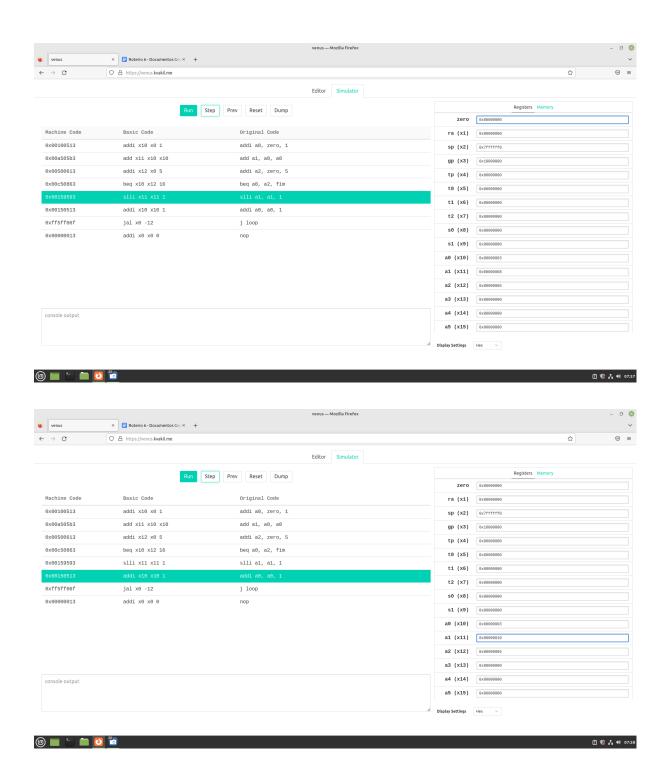


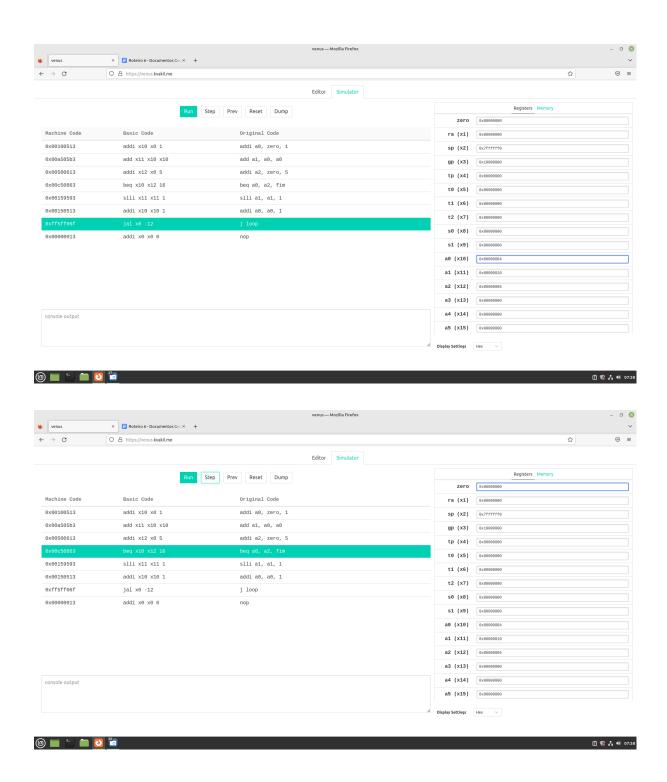


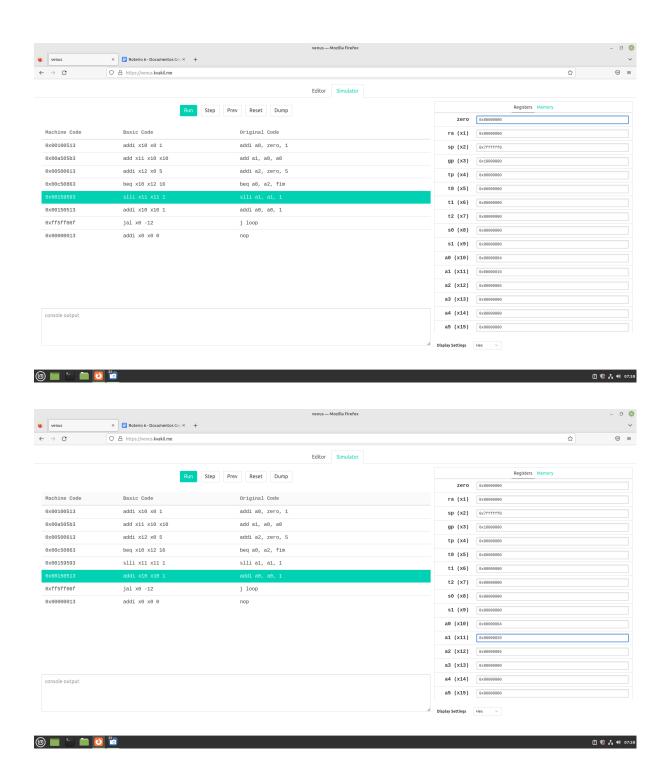


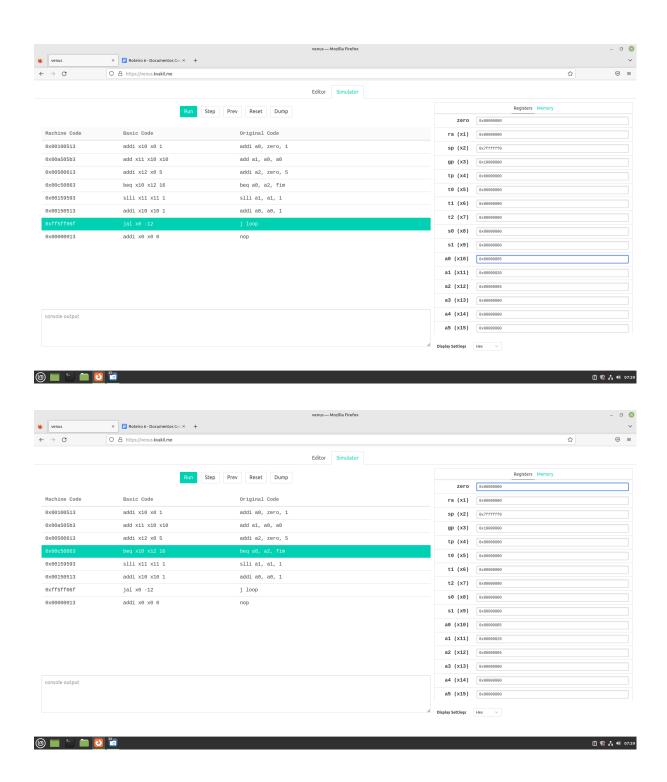


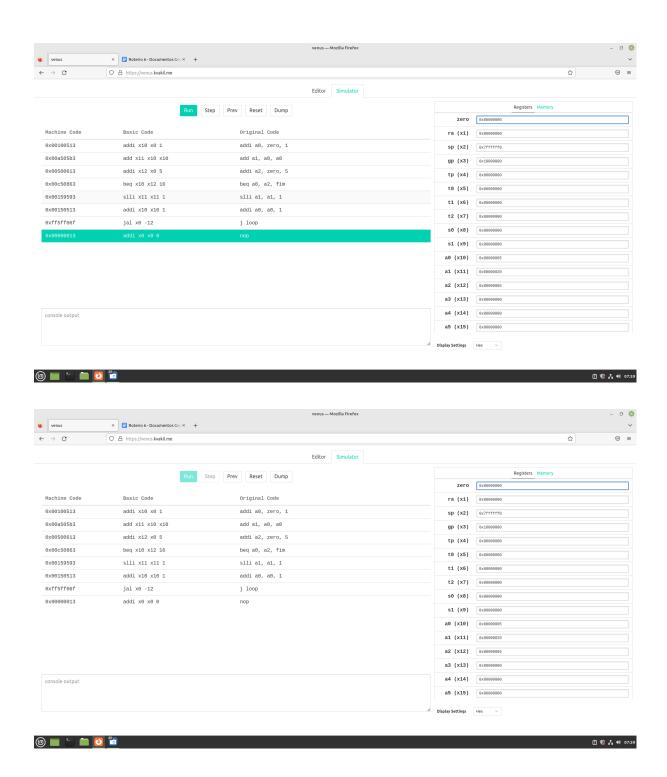












2 - Implementar, a partir do arquivo inst.101, o trecho de código abaixo para RISC-V.

```
0x00200513
0x00400593
0x00b50663
0x00a50633
0x0080006f
0x00b58633
0x0000013
0000000 00010 00000 000 01010 0010011
addi x10, zero, 2
0000000 00100 00000 000 01011 0010011
addi x11, zero, 4
0000000 01011 01010 000 01100 1100011
beq x10, x11, 12
0000000 01010 01010 000 01100 0110011
add x12, x10, x10
000000010000000000 00000 1101111
jal x0, 8
0000000 01011 01011 000 01100 0110011
add x12, x11, x11
0000000 00000 00000 000 00000 0010011
addi x0, x0, 0
addi x10, zero, 2
addi x11, zero, 4
beq x10, x11, 12
add x12, x10, x10
jal x0, 8
add x12, x11, x11
addi x0, x0, 0
ou
addi x10, zero, 2
addi x11, zero, 4
beq x10, x11, jump
add x12, x10, x10
jal x0, fim
jump: add x12, x11, x11
fim: addi x0, x0, 0
```

1. Qual é a operação realizada pelo código acima?

Primeiro ele faz uma soma imediata entre o valor armazenado no registrador x0 e o número 2 e atribui o resultado ao registrador x10.

Depois ele faz uma soma imediata entre o valor do registrador x0 e o número 4 e atribui o resultado ao registrador x11.

Depois ele verifica se os valores dos registradores x10 e x11 são iguais, caso sejam iguais ele dá um salto de 12 bytes e caso não seja continua a execução de forma sequencial.

Em seguida, ele faz uma soma entre os valores armazenados nos registradores x10 e x10 e atribui o resultado ao registrador x12.

Ele não faz a execução do armazenamento no registrador x12 para a soma sucessiva entre o valor armazenado no registrador x11 pois o jump fez com que pulasse 8 bytes, ou seja, para a próxima instrução em seguida.

Depois ele faz um JAL (Jump) de 8 bytes para finalizar a execução.

Por fim, ele faz uma soma imediata entre o valor armazenado no registrador x0 e 0 que finaliza o código.

2. Quais são os registradores utilizados no código?

São utilizados os registradores x0, x10, x11 e x12.

Quais são os conteúdos desses registradores ao final da execução do programa?

x0 = 0

x10 = 2

x11 = 4

x12 = 4

4. Adicione "prints" da tela do ambiente remoto ao PDF. Esses "prints" deverão mostrar os valores exibidos na interface após a execução de cada instrução (código de máquina, pc e demais registradores envolvidos).

