fibonacci:

addi t1, zero, 0 # First number

addi t2, zero, 1 # Second number

addi t0, zero, 1 # current index

addi a1, a0, 0 # ld a1, 0(x10)

L1:

add a0, t1, t2 # y =1

addi t1, t2, 0 # 1

addi t2, t3, 0 # 1

addi t0, t0, 1 # index = 2

bge t0, a1, Exit # first number and input number

j L1

Exit:

ret

convert:

add a7, zero, zero

add a2, zero, zero

add a3, zero, zero

add a4, zero, zero

add a5, zero, zero

#addi a7, zero, 48

add t6, a0, zero

Loop:

add a6, t6, zero

#add a1, zero, zero

add a6, a6, a5

lb a7, 0(a6)

beq a7, zero, Exit

addi t3, zero, 43

bne a3, zero, Label1

bne a7, t3, Label1

addi a3, a3, 1

addi a5, a5, 1

beq zero, zero, Loop

Label1:

addi t3, zero, 45

bne a3, zero, Label2

bne a7, t3, Label2

addi a4, zero, 1

addi a3, a3, 1

addi a5, a5, 1

beq zero, zero, Loop

Label2:

addi t4, zero, 48

addi t5, zero, 57

blt a7, t4, Label3

blt t5, a7, Label3

beq zero, zero, Label4

Label3:

addi a0, zero, -1

ret

Label4:

addi t3, zero, 10

mul a2, a2, t3

addi t4, zero, 48

sub a7, a7, t4

add a2, a2, a7

addi a3, zero, 2

addi a5, a5, 1

j Loop

Exit:

add a0,a2,zero

addi t3, zero, 1

beq a4, t3, Label5

beq zero, zero, Label6

Label5:

addi t3, zero, -1

mul a0, a0, t3

Label6:

ret

matrix\_mul:

addi t0, zero, 128 # size

add s11, zero, a1 # B

add t1, zero, zero # i = 0

loop1:

add t2, zero, zero # j = 0

loop2:

add t3, zero, zero # k = 0

add t4, zero, zero # sum = 0

loop3:

lhu t5, 0(a0) # A[i][k]

lhu t6, 0(s11) # B[k][j]

mul t5, t5, t6 # A\*B

add t4, t4, t5 # C += A\*B

andi t4, t4, 1023 # mod 1024

addi a0, a0, 2 # A[i][k+1]

addi s11, s11, 256 # B[k+1][j]

addi t3, t3, 1 # k++

blt t3, t0, loop3 # k<size , continue

# loop3 end #

sh t4, 0(a2) # store back to C[i][j]

addi a2, a2, 2 # C[i][j+1]

addi a0, a0, -256 # A go back

addi a1, a1, 2 # B[k][j+1]

add s11, zero, a1 # B

addi t2, t2, 1 # j++

blt t2, t0, loop2 # j<size , continue

# loop2 end #

addi a0, a0, 256 # A[i+1][k]

addi a1, a1, -256 # B go back (because line 40 add 2\*128=256)

add s11, zero, a1 # B

addi t1, t1, 1 # i++

blt t1, t0, loop1 # i<size , continue

# loop3 end #

ret

matrix\_mul: # 4 64\*64 blocking matrix

addi t0, zero, 64 # blocking size

add s11, zero, a1 # B

add t1, zero, zero # i = 0

loop4:

addi s3, zero, zero # kk=0

loop5:

addi s2, zero, zero # jj = 0

loop1:

add t2, zero, zero # j = 0

loop2:

add t3, zero, zero # k = 0

add t4, zero, zero # sum = 0

loop3:

lhu t5, 0(a0) # A[i][k]

lhu t6, 0(s11) # B[k][j]

mul t5, t5, t6 # A\*B

add t4, t4, t5 # C += A\*B

andi t4, t4, 1023 # mod 1024

addi a0, a0, 2 # A[i][k+1]

addi s11, s11, 256 # B[k+1][j]

addi t3, t3, 1 # k++

blt t3, t0, loop3 # k<size , continue

# loop3 end #

sh t4, 0(a2) # store back to C[i][j]

addi a2, a2, 2 # C[i][j+1]

addi a0, a0, -256 # A go back (!!!)

addi a1, a1, 2 # B[k][j+1]

add s11, zero, a1 # B

addi t2, t2, 1 # j++

blt t2, t0, loop2 # j<size , continue

# loop2 end #

addi a0, a0, 256 # A[i+1][k]

addi a1, a1, -256 # B go back (because line 40 add 2\*128=256)

add s11, zero, a1 # B

addi t1, t1, 1 # i++

blt t1, t0, loop1 # i<size , continue

# loop1 end #

ret