# Comp. Arch - Unit 2

DATE: 1/4/2022

## EXERCISE 1

Write a function that, given an array of integers, shuffles the array in such a way that the 1st element is the smallest.

```
lw   t0, n        # length
la   t1, x        # address 1st el.
lw   t2, 0(t1)    #  1st elem
ciclo: lw  t3, 4(t1)   #  2nd element

bge  t2, t3, store
mv t2, t3
store:
addi t1, t1, 4
addi t0, t0, -1
bne  t0, zero, ciclo
```

- data

```
n: .word 3
x: .word 3, 2, 1
```

- text

```
primo: mv  t0, a0      )        # a0 address of the array
       mv  t1, a1      )        # a1 the length
       lw  t2, (t0)             t2 to store the smallest
       mv  t3, t0               t3 its address
       addi  t0, t0, 4
       addi  t1, t1, -1
       beq   t1, zero, fine
ciclo: lw  t4, 0(t0)
       bge  t4, t2, salta
       mv   t2, t4
       mv   t3, t0
salta: addi t0, t0, 4
       addi t1, t1, -1
       bne  t1, zero, ciclo

       lw   t5, 0(a0)      # loads 1st element in t5
       sw   t5, 0(t3)      # mette initial element al posto di quello small
       sw   t2, 0(a0)      # mette small nel primo

       jalr  zero, ra, 0
```

## EXERCISE

Use primo to sort an array in the data segment.

- data

```
x: .word  3, 2, -7, 5, 6, 4
n: .word  6
```

```
.text
        la  a0, x          # address of array
        lw  a1, n          # length array

ciclo:  jal  primo         # pseudo-instr.  ra is implicit
        addi a0, a0, 4
        addi a1, a1, -1
        bne a1, zero, ciclo
```

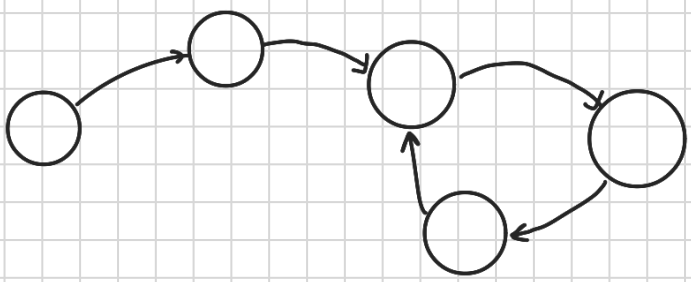
## EXERCISE 3

Given an array and two integers a and b, count the number of elements x[i] such that $a \leq x[i] \leq b$

```
.data
x:  .word  4, 5, 6, 10
a:  .word  3
b:  .word  7
q:  .word  4

.text
        la  t0, x
        lw  t1, n
        lw  t2, a
        lw  t3, b
        li  a0, 0

ciclo:  lw  t4, 0(t0)
        bgt t2, t4, fatto
        ble t3, t4, fatto
        addi a0, a0, 1

fatto:  addi t0, t0, 4
        addi t1, t1, -1
        bne t1, zero, ciclo

        li a7, 1
        ecall
        li a7, 10
        ecall
```

```
        lw  t0, x      address array
        lw  t1, a
        lw  t2, b

        la  t3, q
ciclo:  lw  t4, 0(t0)
        bgt t4, t1, check
        ble t4, t2, check2

check:  ble t4, t2, awesome
check2: bge t4, t1, awesome

awesome:  addi a0, a0, 1

        addi t0, t0, 4
        addi t3, t3, -1
        beq t3, zero, ciclo
```

## EXERCISE

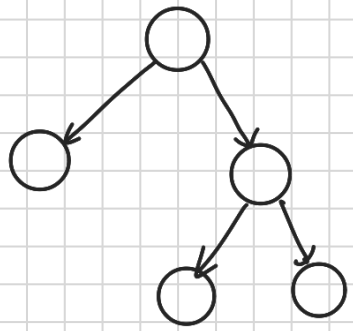


```
.data

  lista: .word  head
  head:

.text

    lw   t0, lista
    li   t1, 1

ciclo: beq  t0, zero, noloop
       sw   t1, 0(t0)
       lw   t0, 4(t0)
       lw   t2, 0(t0)
       beq  t2, zero, ciclo
```

..... SCRATCH THAT - LET'S TRY NEW ONE

```
ciclo: beq  t0, zero, noloop
       lw   t2, 0(t0)
       beq  t2, t1, loop
       sw   t1, 0(t0)
       lw   t0, 0(t0)
       beq  t0, zero, ciclo

noloop:
```

EXERCISE



```
.data
node: .word  7, left_son, right_son

albero: bne  a0, zero, ric
        jalr zero, ra, 0

ric:  addi sp, sp, -12
      sw   ra, 0(sp)
      sw   a0, 4(sp)
      lw   a0, 4(a0)
      jal  albero
      sw   a0, 8(sp)
      lw   a0, 4(sp)
      lw   a0, 8(a0)
      jal  albero
      lw   t0, 8(sp)
      add  a0, a0, t0
      addi a0. a0, 1
      lw   ra, 0(sp)
      addi sp, sp, 12
      jalr zero, ra, 0
```