

# partition()

```
partition:
    # save stuff in the stack
    addi sp, sp, -24
    sd ra, 0(sp)
    sd s10, 8(sp)
    sd s11, 16(sp)

    # init pivot to high (a3)
    add t0, a1, a3
    lbu t0, 0(t0)

    addi t2, a2, -1    # (i) index of the smaller element => t2 = low - 1
    mv t6, a2          # t6 = j = low
    addi t5, a3, -1    # t5 = high-1

    partition_forloop:
    bgt t6, t5, partition_forloop_end    # if t6 > t5 then partition_forloop_end
        add s11, a1, t6    # s11 = *arr[j]
        lbu t1, 0(s11)     # t1 = *(arr[j])

        bgtu t1, t0, partition_forloop_inner_skip    # if t1>t0 skip (if arr[j]>pivot)
            addi t2, t2, 1    # i++
            add s10, a1, t2    # s10 = *arr[t2] = *arr[i]
            lbu t3, 0(s10)     # t3 = *(arr[i])
            sb t3, 0(s11)     # arr[j] = t3
            sb t1, 0(s10)     # arr[i] = t1
        partition_forloop_inner_skip:

        addi t6, t6, 1    # j++
    j partition_forloop
    partition_forloop_end:

    addi a0, t2, 1    # write return value as i+1

    # swap(&arr[i+1], &arr[high])
    add s10, a1, a0    # s10 = *arr[i+1]
    add s11, a1, a3    # s11 = *arr[high]
    lbu t2, 0(s10)     # t2 = *s10
    lbu t3, 0(s11)     # t3 = *s11
    sb t2, 0(s11)
    sb t3, 0(s10)

    # load stuff back from the stack
    ld ra, 0(sp)
    ld s10, 8(sp)
    ld s11, 16(sp)
    addi sp, sp, 24
partition_bail:
    ret
```

```
def partition(arr, low, high):
    i = (low-1)    # index of smaller element
    pivot = arr[high]    # pivot

    for j in range(low, high):
        # If current element is smaller than or
        # equal to pivot
        if arr[j] <= pivot:

            # increment index of smaller element
            i = i+1
            arr[i], arr[j] = arr[j], arr[i]

    arr[i+1], arr[high] = arr[high], arr[i+1]
    return(i+1)
```

# Debug e verifica delle funzionalità

- **Compilazione**

È necessario eseguire `./compile.sh`

```
calcolatori@calcolatori-VirtualBox: ~/riscv-asm-quicksort
calcolatori@calcolatori-VirtualBox:~/riscv-asm-quicksort$ ./compile.sh
calcolatori@calcolatori-VirtualBox:~/riscv-asm-quicksort$
```

- **Esecuzione**

È obbligatorio eseguirlo con il debugger attivo (altrimenti non sarà possibile vedere il risultato) scrivendo `qemu-riscv64 -g 2323 ./quicksort`

```
calcolatori@calcolatori-VirtualBox: ~/riscv-asm-quicksort (ssh)
calcolatori@calcolatori-VirtualBox:~/riscv-asm-quicksort$ qemu-riscv64 -g 2323 ./quicksort
```

- **Debug**

Per debuggare basterà avviare `./debug.sh`;  
Basterà poi scrivere `display/7b &nomevar`; per visualizzare `&nomevar` a ogni step.  
Ad esempio: `display/7b &testarray`

```
1: x/7xb &testarray
0x11285:      0x01      0x03      0x05      0x06      0x07      0x08      0x0a
(gdb)
[Inferior 1 (Remote target) exited with code 06]
(gdb)
```

```
calcolatori@calcolatori-VirtualBox: ~/riscv-asm-quicksort (ssh)
calcolatori@calcolatori-VirtualBox:~/riscv-asm-quicksort$ ./debug.sh
GNU gdb (GDB) 8.0.50.20170724-git
Copyright (C) 2017 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law. Type "show copying"
and "show warranty" for details.
This GDB was configured as "--host=x86_64-pc-linux-gnu --target=riscv64-unknown-elf".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from quicksort...done.
Remote debugging using :2323
_start () at main.S:18
18      la a1, testarray
(gdb) display/7b &testarray
1: x/7xb &testarray
0x11285:      0x05      0x01      0x03      0x07      0x08      0x0a      0x06
(gdb)
```

```
calcolatori@calcolatori-VirtualBox: ~/riscv-asm-quicksort (ssh)
40      ld s11, 16(sp)
1: x/7xb &testarray
0x11285:      0x01      0x03      0x05      0x06      0x07      0x08      0x0a
(gdb)
41      ld s9, 24(sp)
1: x/7xb &testarray
0x11285:      0x01      0x03      0x05      0x06      0x07      0x08      0x0a
(gdb)
42      addi sp, sp, 32
1: x/7xb &testarray
0x11285:      0x01      0x03      0x05      0x06      0x07      0x08      0x0a
(gdb)
quicksort_exit () at quicksort.S:44
44      ret
1: x/7xb &testarray
0x11285:      0x01      0x03      0x05      0x06      0x07      0x08      0x0a
(gdb)
_start () at main.S:26
26      ecall
1: x/7xb &testarray
0x11285:      0x01      0x03      0x05      0x06      0x07      0x08      0x0a
(gdb)
[Inferior 1 (Remote target) exited with code 06]
(gdb)
```