**Sorting Algorithm**

**Bubble Sort**

**C++ code:**

int i, j;

for (i = 0; i < n - 1; i++) {

for (j = 0; j < n - i - 1; j++) {

if (arr[j] > arr[j + 1]) {

swap(arr[j], arr[j + 1]);

}

}

X5- array

X6 – n

X28 - I

X10 = n-1

X29= n-I-1

X30 = j

X9 – offsetted base address arr[j]

X31 – j\*4

X7 = offsetted base address arr[j+1]

X18 = arr[j]

X12 = arr[j+1]

**Assembly code RISC V:**

li x5 0 # x5 holds the base address of arr

li x6 6 # x6 holds the value of n

# Initialising array

li x28 8

Sd x28 7(x5)

li x28 4

sdx28 15(x5)

li x28 2

sd x28 23(x5)

li x28 15

sdx28 31(x5)

li x28 1

sd x28 39(x5)

li x28 6

sd x28 47(x5)

li x28, 0 # i = 0

addi x10 x6 -1 # x10 = n - 1

LoopI:

bge x28, x10, EndI # i >= n check, end outer

sub x29, x6, x28 # x29 = n - i

addi x29 x29 -1 # x29 = n - i - 1

li x30, 0 # j = 0

LoopJ:

bge x30, x29, EndJ # j >= n - i - 1check, end inner

# Access arr[j] and arr[j + 1]

slli x31, x30, 3 # x31 = j \* word size

add x9, x5, x31 # x9 = &arr[j]

addi x7, x9, 8 # x7 = &arr[j + 1]

ld x18, 0(x9) # x18 = arr[j]

ld x12, 0(x7) # x12 = arr[j + 1]

bge x12, x18, no\_swap # if arr[j+1] >= arr[j], no swap

# Swap arr[j] and arr[j + 1]

sd x12, 0(x9) # arr[j] = x12

sd x18, 0(x7) # arr[j + 1] = x18

no\_swap:

addi x30, x30, 1 # j++

beq x0 x0 LoopJ

EndJ:

addi x28, x28, 1 # i++

beq x0 x0 LoopI

EndI:

**Machine code:**

00000293

00600313

00800e13

01c2a3a3

00400e13

01c2a7a3

00800e13

01c2aba3

00f00e13

01c2afa3

00100e13

03c2a3a3

00600e13

03c2a7a3

00000e13

fff30513

04ae5263

41c30eb3

fffe8e93

00000f13

03df5663

003f1f93

01f284b3

00848393

0004a903

0003a603

01265663

00c4a023

0123a023

001f0f13

fc000ce3

001e0e13

fc0000e3