```
/**
 * substr.s
 * PARAMETERS: X0 (STRING)
 *             X1 (RANGE BEGIN)
 * OUTPUT    : X0 (POINTER TO SUBSTRING)
 * This routine goes from the begin index in X1 to the end of a given string in
X0,
 * and returns the resulting substring in X0.
 * ALL REGISTERS PRESERVED EXCEPT X0
 */

    .text
    .global substrend
substrend:
        // storing X0-X19 registers, as malloc will not preserve most of these
        str X1,     [sp, -16]!
        stp X2, X3, [sp, -16]!
        stp X4, X5, [sp, -16]!
        stp X6, X7, [sp, -16]!
        stp X8, X9, [sp, -16]!
        stp X10, X11, [sp, -16]!
        stp X12, X13, [sp, -16]!
        stp X14, X15, [sp, -16]!
        stp X16, X17, [sp, -16]!
        stp X18, X19, [sp, -16]!
        stp X20, X21, [sp, -16]!
        str lr, [sp, -16]!
        mov x20, x1     // copying begin in x20
        mov x19, x0     // copying string in x19
        mov x1, #10000  // our string maximum for length routine
        bl length       // need length of string for malloc
        add x0, x0, #1  // accounting for null terminator
        sub x0, x0, x20

substrPreLoop:
        bl malloc       // calling malloc with requests bytes
        mov x21, #0
substrLoop:
        ldrb w17, [x19, x20]    // loading byte of given string into w17
        strb w17, [x0, x21]     // storing w17 into new string
        add x21, x21, #1          // incrementing
        add x20, x20, #1        // incrementing
        cmp w17, #0
        b.ne substrLoop         // if increment ≥ end, goto end label
substrEnd:
         // popping registers back from stack
        ldr lr, [sp], 16
        ldp X20, X21, [sp], 16
        ldp X18, X19, [sp], 16
        ldp X16, X17, [sp], 16
        ldp X14, X15, [sp], 16
        ldp X12, X13, [sp], 16
        ldp X10, X11, [sp], 16
        ldp X8, X9, [sp], 16
        ldp X6, X7, [sp], 16
        ldp X4, X5, [sp], 16
        ldp X2, X3, [sp], 16
        ldr X1,     [sp], 16
        ret lr
```