```
/**
 * substr.s
 * PARAMETERS: X0 (STRING)
 *             X1 (RANGE BEGIN)
 *             X2 (RANGE END)
 * OUTPUT    : X0 (POINTER TO SUBSTRING)
 * ALL REGISTERS PRESERVED EXCEPT X0
 */

.text
.global substr
    substr:
            // storing X0-X19 registers, as malloc will not preserve most of the
se
            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 x19, x0     // copying string in x19
            mov x20, x1     // copying begin in x20
            mov x21, x2     // copying end in x21
            mov x0, x21
            sub x0, x0, x20
            sub x0, x2, x1  // using difference for malloc
            cmp x0, #0      // if the difference is less or equal to 0, input is
 invalid. exit routine
            b.ge substrPreLoop
            mov x0, #0      // invalid input, throw null
            b substrEnd

    substrPreLoop:
            add x0, x0, #1  // need one extra byte for null
            bl malloc       // calling malloc with requests bytes
            mov x1, #0
    substrLoop:
            ldrb w17, [x19, x20]    // loading byte of given string into w17
            strb w17, [x0, x1]      // storing w17 into new string
            add x1, x1, #1
            add x20, x20, #1        // incrementing
            cmp x20, x21            // comparing x19 to x20
            b.lt substrLoop         // if increment ≥ end, goto end label
            mov w17, #0             // storing null
            strb w17, [x0, x1]
    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
```