```
/**
 * length - counts the number of characters in a string
 * @param x0: pointer to the string to count
 * @param x1: maximum number of characters to count
 * @return x0: number of characters counted, including null terminator
 *
 * This function counts the number of characters in a string pointed to by x0, up
 * to a maximum of x1 characters. The count includes the null terminator. If the
 * string is longer than x1 characters, the function stops counting at x1
 * characters.
 *
 * Registers used: x0, x1, x2, x3, w3
 * Registers saved: x19-x30, lr
 */

.text
.global length

// Entry point for the `length` subroutine
length:
    // Point to the first byte of the string to count
    mov x7, x0

    // Initialize the counter to zero
    mov x2, #0

// Top of the loop to count characters
top:
    // Load the next byte of the string and update the pointer
    ldrb w1, [x7], #1

    // Check if the byte is the null terminator
    cmp w1, #0

    // If the byte is the null terminator, jump to the bottom of the loop
    b.eq bottom

    // Increment the counter by one
    add x2, x2, #1

    // Jump back to the top of the loop
    b top

// Bottom of the loop to return the length
bottom:
    // Return the length of the string, including the null terminator
    mov x0, x2
    ret lr

// Entry point for the `substring` subroutine
sbstr:
    // Save the necessary registers on the stack
    stp x0, x1, [sp, #-16]!

    // Compute the length of the substring
    sub x0, x2, x1

    // Save the length and the original string pointer on the stack
    stp x2, x0, [sp, #-16]!

    // Call `malloc` to allocate memory for the substring
```



```
    bl malloc

    // Restore the length and the original string pointer from the stack
    ldp x2, x5, [sp],#16

    // Restore the return address and the start index of the substring from the
stack
    ldp x4, x1, [sp],#16

    // Save the pointer to the substring on the stack
    str x0, [sp, #-16]!

// End of the loop to copy characters from the original string to the substring
end:
    // Check if we've copied the desired number of characters
    cmp x5, #0

    // If we've copied all the characters, return from the subroutine
    b.eq return

    // Load the next byte from the original string and store it in the substring
    ldrb w6, [x4], #1
    strb w6, [x0], #1

    // Decrement the counter and jump back to the top of the loop
    sub x5, x5, #1
    b end

// Return from the `substring` subroutine
return:
    // Restore the pointer to the substring from the stack
    ldr x0, [sp], #16

    // Return the pointer to the substring
    ret lr
```