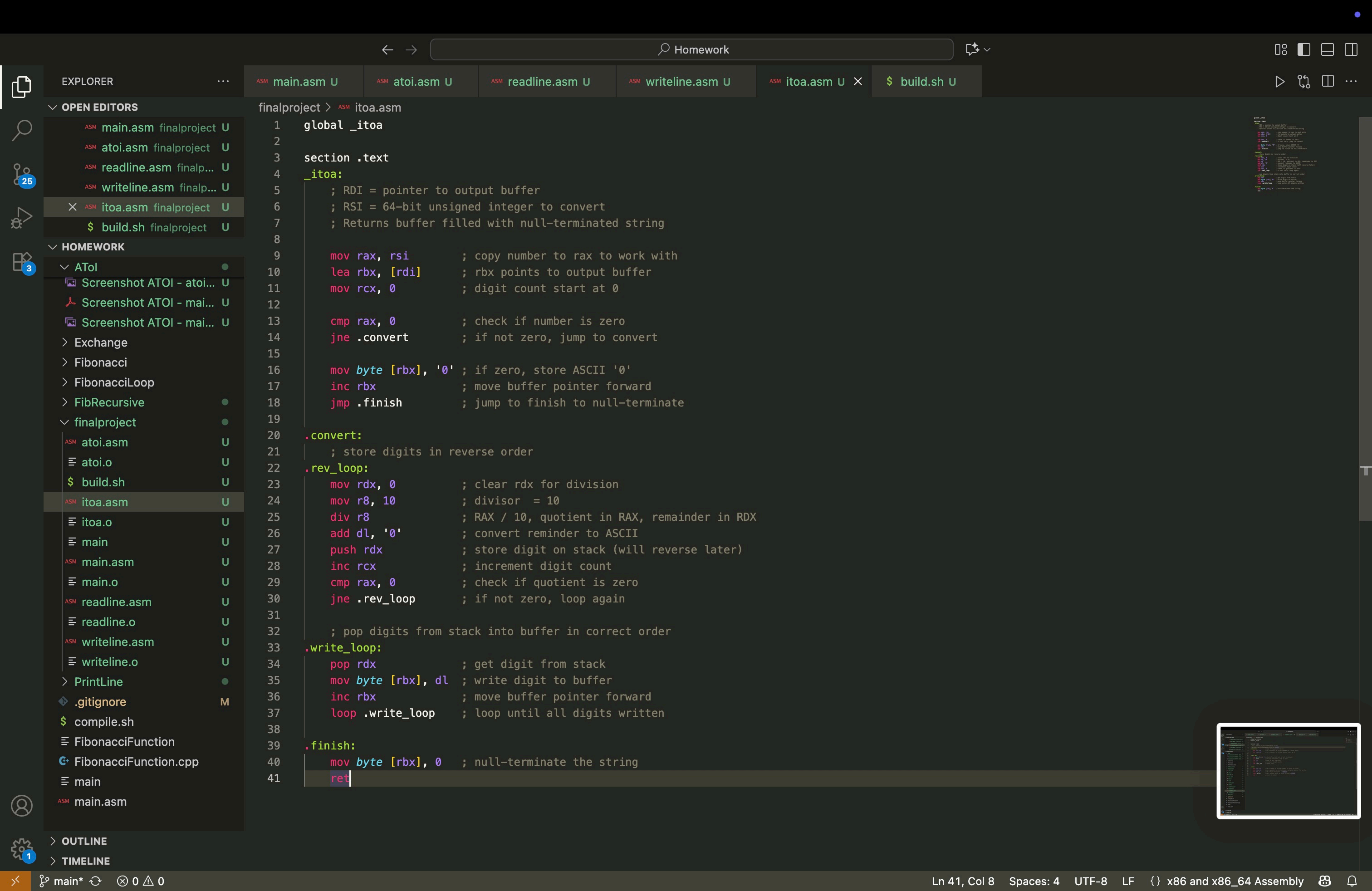

```
global _itoa

section .text
_itoa:
    ; RDI = pointer to output buffer
    ; RSI = 64-bit unsigned integer to convert
    ; Returns buffer filled with null-terminated string

    mov rax, rsi        ; copy number to rax to work with
    lea rbx, [rdi]      ; rbx points to output buffer
    mov rcx, 0          ; digit count start at 0

    cmp rax, 0          ; check if number is zero
    jne .convert        ; if not zero, jump to convert

    mov byte [rbx], '0' ; if zero, store ASCII '0'
    inc rbx             ; move buffer pointer forward
    jmp .finish         ; jump to finish to null-terminate

.convert:
    ; store digits in reverse order
.rev_loop:
    mov rdx, 0          ; clear rdx for division
    mov r8, 10          ; divisor  = 10
    div r8              ; RAX / 10, quotient in RAX, remainder in RDX
    add dl, '0'         ; convert reminder to ASCII
    push rdx            ; store digit on stack (will reverse later)
    inc rcx             ; increment digit count
    cmp rax, 0          ; check if quotient is zero
    jne .rev_loop       ; if not zero, loop again

    ; pop digits from stack into buffer in correct order
.write_loop:
    pop rdx             ; get digit from stack
    mov byte [rbx], dl  ; write digit to buffer
    inc rbx             ; move buffer pointer forward
    loop .write_loop    ; loop until all digits written

.finish:
    mov byte [rbx], 0   ; null-terminate the string
    ret
```