CMPE 310 - Lab 1

The Files

	Category	**What it does**
compile.sh	Helper File	This file runs a few commands that compiles and links assembly.
sumFile	Code File	This is the compiled exe of the code.
sumFile.asm	Code File	This is the assembly code of the program.
sumFile.o	Code File	
tester.js	Helper File	This file is JavaScript program that runs a bunch of tests on the given program. I just used this to run hundreds or thosands of tests against my program.
customTest.txt	Helper File	This is the file that I used to check the code works with another file.
randomInt100.txt	Helper File	This is the file given to us that can be used to test the program.
README.md	Git File	Just a readme
report.pdf	Lab Report	This is the final lab report file.

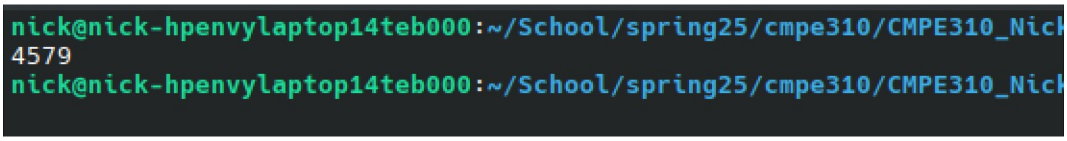
The Pseudo Code

```
BEGIN
    POP argument_count
    COMPARE argument_count WITH 2
    IF argument_count < 2 THEN
        THROW ERROR
    END IF

    POP executable_name
    POP filename

    COPY_STRING(filename, destination = filename_variable)

    OPEN_FILE(filename_variable, mode = read_only)
    IF FILE_OPEN_FAILED THEN
        THROW ERROR
    END IF

    READ_FILE(file_descriptor, buffer, 4096)
    IF READ_FAILED THEN
        THROW ERROR
    END IF

    SET sum = 0
    SET position = buffer_start
    SET bytes_read = number_of_bytes_read_from_file

    WHILE position < buffer_start + bytes_read DO
        SET character = *position
        INCREMENT position

        IF character IS NEWLINE THEN
            IF first_number_flag IS FALSE THEN
                ADD number_to_sum(sum, current_number)
            END IF
            SET current_number = 0
            SET first_number_flag = FALSE
        ELSE IF character IS DIGIT THEN
            CONVERT character TO INTEGER
            MULTIPLY current_number BY 10
            ADD character TO current_number
        END IF
    END WHILE

    IF first_number_flag IS FALSE THEN
        ADD number_to_sum(sum, current_number)
    END IF

    PRINT_NUMBER(sum)
    PRINT NEWLINE

    CLOSE_FILE(file_descriptor)

    EXIT PROGRAM
END
```

The Code

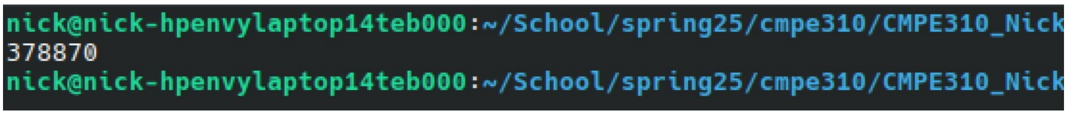
```
section .bss
    buffer resb 4096
    sum resd 1
    filename resb 100

section .data
    msg_open_fail db "error opening file", 10, 0
    msg_read_fail db "error reading file", 10, 0
    newline db 10, 0
    file_descriptor dd 0

section .text
    global _start

    _start:
        pop eax
        cmp eax, 2
        jl exit

        pop eax
        pop ebx

        mov esi, ebx
        mov edi, filename
        call copy_string

        mov eax, 5
        mov ebx, filename
        mov ecx, 0
        int 0x80

        cmp eax, 0
        jl open_fail
        mov [file_descriptor], eax

        mov ebx, eax
        mov eax, 3
        mov ecx, buffer
        mov edx, 4096
        int 0x80

        cmp eax, 0
        jle read_fail
        mov edx, eax

        xor eax, eax
        mov [sum], eax

        mov esi, buffer
        mov ecx, edx
        call add_integers

        mov eax, [sum]
        call print_number

    close_file:
        mov eax, 6
        mov ebx, [file_descriptor]
        int 0x80

    exit:
        mov eax, 1
        xor ebx, ebx
        int 0x80

    open_fail:
        mov eax, 4
        mov ebx, 1
        mov ecx, msg_open_fail
        mov edx, 19
        int 0x80
        jmp exit

    read_fail:
        mov eax, 4
        mov ebx, 1
        mov ecx, msg_read_fail
        mov edx, 19
        int 0x80
        jmp close_file

    copy_string:
        .loop:
            mov al, [esi]
            mov [edi], al
            inc esi
            inc edi
            test al, al
            jnz .loop
        ret

    add_integers:
        xor eax, eax
        xor ebx, ebx
        xor edi, edi
        mov dl, 1

    next_char:
        cmp edi, ecx
        jge done
        mov al, [esi]
        inc esi
        inc edi

        cmp al, 10
        je add_to_sum
        cmp al, '0'
        jl next_char
        cmp al, '9'
        jg next_char
        sub al, '0'
        imul ebx, ebx, 10
        add ebx, eax
        jmp next_char

    add_to_sum:
        cmp dl, 1
        je skip_first_num
        add [sum], ebx

    skip_first_num:
        xor ebx, ebx
        mov dl, 0
        jmp next_char

    done:
        cmp dl, 1
        je ret_skip
        add [sum], ebx

    ret_skip:
        ret

    print_number:
        mov ecx, buffer
        add ecx, 4096
        mov edi, ecx
        mov ebx, 10

    .convert:
        xor edx, edx
        div ebx
        add dl, '0'
        dec edi
        mov [edi], dl
        test eax, eax
        jnz .convert

        mov edx, ecx
        sub edx, edi
        mov eax, 4
        mov ebx, 1
        mov ecx, edi
        int 0x80

        mov eax, 4
        mov ebx, 1
        mov ecx, newline
        mov edx, 1
        int 0x80
        ret
```

The Output



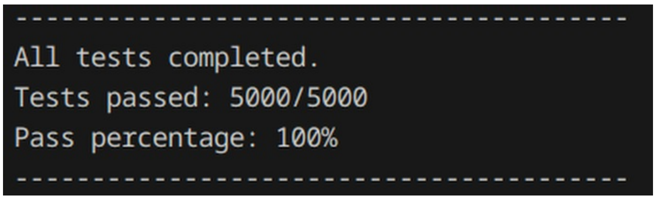
```
nick@nick-hpenvylaptop14teb000:~/School/spring25/cmpe310/CMPE310_Nick
4579
nick@nick-hpenvylaptop14teb000:~/School/spring25/cmpe310/CMPE310_Nick
```

Output of the code



```
nick@nick-hpenvylaptop14teb000:~/School/spring25/cmpe310/CMPE310_Nick
378870
nick@nick-hpenvylaptop14teb000:~/School/spring25/cmpe310/CMPE310_Nick
```

Custom test output



```
-----------------------------------------
All tests completed.
Tests passed: 5000/5000
Pass percentage: 100%
-----------------------------------------
```

Tests passed