Andrew Riley

CSCE 451-500

Dr. Jhy Liu

27 Jan 2020

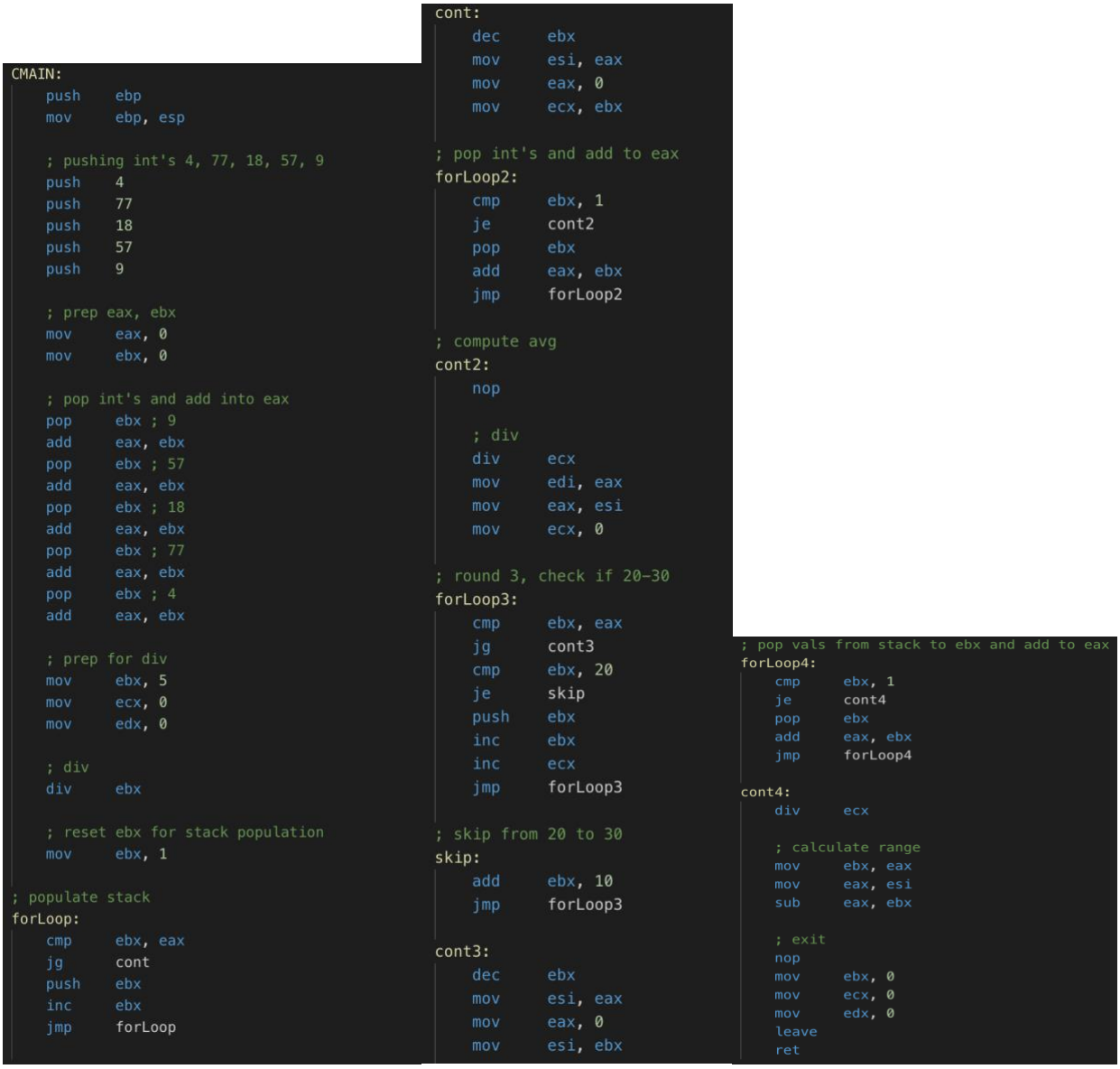
Homework 1 Report

In this assignment we covered three different problems: computing the average of five different values by using *push* and *pop* to interact with the stack, computing averages for two more sequences of numbers using different variations of the *jmp* command to implement low-level for loops, and tying it all together to populate the stack with a sequence of numbers using *jmp* commands to replicate *if* statements and finding the product and sums of values within the sequence. I refreshed myself on how to implement loops and conditional statements in x86. I used the same *.asm* files I had made for the Spring of 2019 class that I was enrolled in, but reviewed all the code that I had written to ensure that I fully understood the significance of all the commands and ensure I had a better grasp of the Assembly instructions and why I had used what I did. I made no edits because I believed that the code that I had written worked well and had no glaring errors or improvements that could be made. Based on the knowledge I gained, I can more effectively recognize the presence of loops and conditional statements, as well as the creation of specific variables based on sizes. I have included the *io.inc* file that is automatically added to the beginning of each SASM program for reference; however, I have removed any necessity for it when moving over to the macOS side. As per critiques on the homework, I don't really see any glaring issues with it. The homework seems relatively straight forward to complete and many guides can be found online to help the student especially when it comes to the bonus (I didn't complete the bonus because I had neglected to start until later due to working nearly full time with the Division of IT).

```
CMAIN:
        push    ebp
        mov     ebp, esp

        ; pushing int's 4, 77, 18, 57, 9
        push    4
        push    77
        push    18
        push    57
        push    9

        ; prep eax, ebx
        mov     eax, 0
        mov     ebx, 0

        ; pop int's and add into eax
        pop     ebx ; 9
        add     eax, ebx
        pop     ebx ; 57
        add     eax, ebx
        pop     ebx ; 18
        add     eax, ebx
        pop     ebx ; 77
        add     eax, ebx
        pop     ebx ; 4
        add     eax, ebx

        ; prep for div
        mov     ebx, 5
        mov     ecx, 0
        mov     edx, 0

        ; div
        div     ebx

        ; exit
        leave
        ret
```

Problem 1:

I set up my development environment in SASM and ran it through the debugger and compiler that was provided within the IDE; however, I took my screenshots in VS Code and ensured that I could compile and run the program within macOS Mojave (v. 10.14.6) using the commands *nasm -f macho {.asm file} -o {.o file}* and *ld -macosx_version_min 10.14.0 -lSystem -o {binary file} {.o file}*.



```
CMAIN:
    push    ebp
    mov     ebp, esp

    ; pushing int's 4, 77, 18, 57, 9
    push    4
    push    77
    push    18
    push    57
    push    9

    ; prep eax, ebx
    mov     eax, 0
    mov     ebx, 0

    ; pop int's and add into eax
    pop     ebx ; 9
    add     eax, ebx
    pop     ebx ; 57
    add     eax, ebx
    pop     ebx ; 18
    add     eax, ebx
    pop     ebx ; 77
    add     eax, ebx
    pop     ebx ; 4
    add     eax, ebx

    ; prep for div
    mov     ebx, 5
    mov     ecx, 0
    mov     edx, 0

    ; div
    div     ebx

    ; reset ebx for stack population
    mov     ebx, 1

; populate stack
forLoop:
    cmp     ebx, eax
    jg      cont
    push    ebx
    inc     ebx
    jmp     forLoop
```

```
cont:
    dec     ebx
    mov     esi, eax
    mov     eax, 0
    mov     ecx, ebx

; pop int's and add to eax
forLoop2:
    cmp     ebx, 1
    je      cont2
    pop     ebx
    add     eax, ebx
    jmp     forLoop2

; compute avg
cont2:
    nop

    ; div
    div     ecx
    mov     edi, eax
    mov     eax, esi
    mov     ecx, 0

; round 3, check if 20-30
forLoop3:
    cmp     ebx, eax
    jg      cont3
    cmp     ebx, 20
    je      skip
    push    ebx
    inc     ebx
    inc     ecx
    jmp     forLoop3

; skip from 20 to 30
skip:
    add     ebx, 10
    jmp     forLoop3

cont3:
    dec     ebx
    mov     esi, eax
    mov     eax, 0
    mov     esi, ebx
```

```
; pop vals from stack to ebx and add to eax
forLoop4:
    cmp     ebx, 1
    je      cont4
    pop     ebx
    add     eax, ebx
    jmp     forLoop4

cont4:
    div     ecx

    ; calculate range
    mov     ebx, eax
    mov     eax, esi
    sub     eax, ebx

    ; exit
    nop
    mov     ebx, 0
    mov     ecx, 0
    mov     edx, 0
    leave
    ret
```

Problem 2:

I set up my development environment in SASM and ran it through the debugger and compiler that was provided within the IDE; however, I took my screenshots in VS Code and ensured that I could compile and run the program within macOS Mojave (v. 10.14.6) using the commands *nasm -f macho {.asm file} -o {.o file}* and *ld -macosx_version_min 10.14.0 -lSystem -o {binary file} {.o file}*.

```
CMAIN:
    mov ebp, esp; for correct debugging
    push    ebp
    mov     ebp, esp
    mov     eax, 0
    mov     ebx, 0
    mov     ecx, 0

; push vals to stack
forLoop:
    cmp     eax, 100
    jg      cont
    push    eax
    inc     eax
    cmp     eax, 43
    je      meaningOfLife
    inc     ecx
    jmp     forLoop

; push 42, 9 more times to stack
meaningOfLife:
    cmp     ebx, 9
    je      forLoop
    push    42
    inc     ebx
    inc     ecx
    jmp     meaningOfLife

cont:
    ; num iterations
    mov     eax, ecx

    ; reset reg's
    mov     eax, 1
    mov     ebx, 0
    mov     ecx, 0
    mov     edx, 0

; pop vals off stack to edx, check if under 30, save sum to ebx
forLoop2:
    pop     edx
    add     ebx, edx
    cmp     edx, 0
    je      EXIT
    cmp     edx, 30
    jl      multiply
    jmp     forLoop2
```

```
; multiply any vals < 30, store in eax
multiply:
    mul     edx
    jmp     forLoop2

; exit pgm
EXIT:
    nop
    mov     eax, 0
    mov     ebx, 0
    mov     ecx, 0
    mov     edx, 0
    mov     esi, 0
    mov     edi, 0
    leave
    ret
```

Problem 3:

I set up my development environment in SASM and ran it through the debugger and compiler that was provided within the IDE; however, I took my screenshots in VS Code and ensured that I could compile and run the program within macOS Mojave (v. 10.14.6) using the commands *nasm -f macho {.asm file} -o {.o file}* and *ld -macosx_version_min 10.14.0 -lSystem -o {binary file} {.o file}*.