

**Computer Science Department**  
**California State University, Fullerton**

CPSC 240-01 Computer Organization and Assembly Language

Final Exam

1:00 PM to 2:40 PM

Tuesday, May 14, 2024

Student Name: Ryan Nishikawa

Last 4 digits of ID: 6761

**Note:**

- University regulations on academic honesty will be strictly enforced.
  - You have 100 minutes to complete this Quiz.
  - Open books, slides and sample programs.
  - Turn off or turn vibration your cell phone.
  - Use YASM assembler for the program design.
  - Copy and paste your assembly source code and Terminal Emulator window to the end of the word file and save it in pdf or docx format.
  - Submit you pdf or docx file to Canvas before the deadline.
- NOTE: Email submissions will not be graded.**
- Any content submitted after the due date will be regarded as a make-up exam.

1. Design an assembly program that reads 3 single digit from keyboard to variables (num1, num2, and num3), calculates “sum=num1+num2” and “quotient=sum/num3”, and displays the sum and quotient to the terminal.
2. The program needs to define “scan” and “print” macros for sys\_read and sys\_write, define “calculate” and “toString” functions for calculation and converting integer to ASCII for displaying.
3. When calling “scan” and “print” macros, the caller needs to pass the buffer address and number of characters. When calling the “calculate” function, the caller needs to pass num1, num2, num3, &sum, &quotient. When calling the “toString” function, the caller needs to pass &sum/&quotient and &ascii.
4. The following pseudo code provides students with a reference when writing assembly code. Students can directly convert the following pseudo code into assembly code.

```

;finalExam_01.asm
#begin define print(addr, n)
    rax = 1;
    rdi = 1;
    rsi = addr of string;
    rdx = n;
    syscall;
#end
#begin define scan(&addr, n)
    rax = 1;
    rdi = 1;
    rsi = &addr;
    rdx = n;
    syscall;
#end

char num1, num2, num3, sum, quotient;
char buffer[2];
char ascii[10];
char msg1[24] = "Input 1st number (0~9): ";
char msg2[24] = "Input 2nd number (0~9): ";
char msg3[24] = "Input 3rd number (0~9): ";
char msg4[6] = "sum = ";
char msg5[11] = "quotient = ";
void main() {
    print msg1, 24;
    scan buffer, 2
    num1 = atoi(buf);
    print msg2, 24;
    scan buffer, 2
    num2 = atoi(buf)
    print msg3, 24;
    scan buffer, 2
    num3 = atoi(buf)

    call calculate(num1, num2, num3, &sum, &quotient);
    call toString(&sum, &ascii)
    print msg4, 6;
    print ascii, 4;
    call toString(&quotient, &ascii)
    print msg5, 11;
}

```

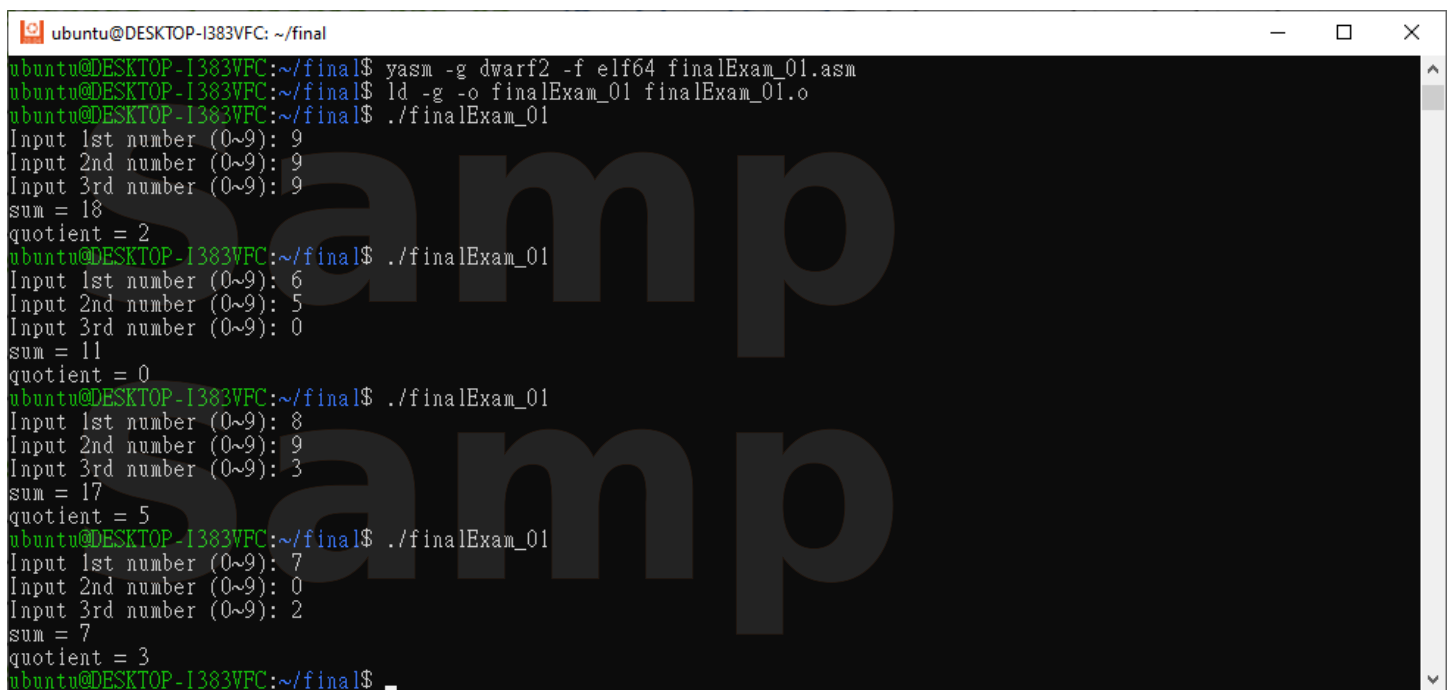
```

    print ascii, 4;
}
void calculate(num1, num2, num3, &sum, &quotient) {
    sum = num1 + num2;
    if(num3 != 0)
        quotient = sum / num3;
    else
        quotient = 0;
}
void toString(&argument, &ascii) {
    ascii = itoa(argument);
}

```

5. After assembling and linking, run the executable file to display the simulation results in the Terminal Emulator window as the following example.
6. Insert source code and the simulation results (Terminal Emulator window) to the bottom of the document.
7. Save the file in pdf or docx format and submit the pdf or docx file to Canvas before the deadline.
8. Deadline is 2:40 pm on 5/14/2024.

### Simulation result example:



```

ubuntu@DESKTOP-I383VFC: ~/final
ubuntu@DESKTOP-I383VFC:~/final$ yasm -g dwarf2 -f elf64 finalExam_01.asm
ubuntu@DESKTOP-I383VFC:~/final$ ld -g -o finalExam_01 finalExam_01.o
ubuntu@DESKTOP-I383VFC:~/final$ ./finalExam_01
Input 1st number (0~9): 9
Input 2nd number (0~9): 9
Input 3rd number (0~9): 9
sum = 18
quotient = 2
ubuntu@DESKTOP-I383VFC:~/final$ ./finalExam_01
Input 1st number (0~9): 6
Input 2nd number (0~9): 5
Input 3rd number (0~9): 0
sum = 11
quotient = 0
ubuntu@DESKTOP-I383VFC:~/final$ ./finalExam_01
Input 1st number (0~9): 8
Input 2nd number (0~9): 9
Input 3rd number (0~9): 3
sum = 17
quotient = 5
ubuntu@DESKTOP-I383VFC:~/final$ ./finalExam_01
Input 1st number (0~9): 7
Input 2nd number (0~9): 0
Input 3rd number (0~9): 2
sum = 7
quotient = 3
ubuntu@DESKTOP-I383VFC:~/final$

```

[Attach your assembly source code here:]

```

%macro print 2
mov rax, 1 ;SYS write
mov rdi, 1 ;where to write
mov rsi, %1 ;address of string
mov rdx, %2 ;number of character

```

```
syscall ;calling system services
```

```
%endmacro
```

```
%macro scan 2
```

```
mov rax, 0 ;SYS_read
```

```
mov rdi, 0 ;standard input device
```

```
mov rsi, %1 ;input buffer address
```

```
mov rdx, %2 ;number of character
```

```
syscall ;calling system services
```

```
%endmacro
```

```
section .bss
```

```
num1 resb 1
```

```
num2 resb 1
```

```
num3 resb 1
```

```
sum resb 1
```

```
quotient resb 1
```

```
buffer resb 2
```

```
ascii resb 10
```

```
asciiLen resb 1
```

```
argument resb 1
```

```
section .data
```

```
LF equ 10
```

```
NULL equ 0
```

```
SYS_exit equ 60
```

```
EXIT_SUCCESS equ 0
```

```
msg1 db "Input 1st number (0~9): ", NULL
```

```
msg2 db "Input 2nd number (0~9): ", NULL
```

```
msg3 db "Input 3rd number (0~9): ", NULL
```

```
msg4 db "sum = ", NULL
```

```
msg5 db "quotient = ", NULL
```

```
section .text
```

```
global start
```

```
start:
```

```
print msg1, 24
```

```
scan buffer, 2
```

```
mov al, byte[buffer]
```

```
and al, 0fh
```

```
mov byte[num1], al
```

```
print msg2, 24
```

```
scan buffer, 2
```

```
mov al, byte[buffer]
```

```
and al, 0fh
```

```
mov byte[num2], al
```

```
print msg3, 24
```

```
scan buffer, 2
```

```
mov al, byte[buffer]
```

```
and al, 0fh
```

```
mov byte[num3], al
```

```
mov dil, byte[num1]
```

```
mov sil, byte[num2]
```

```
call calculate
```

```
print msg4, 6
```

```
mov al, byte[sum]
```

```
call toString
```

```
print msg5, 11
```

```
mov al, byte[quotient]
```

```
call toString
```

```
;end
```

```
mov rax, SYS_exit ;terminate |excuting process
```

```
mov rdi, EXIT_SUCCESS ;exit status
```

```
syscall
```

```
calculate:
```

```
mov al, dil
```

```
add al, sil
```

```
mov byte[sum], al
```

```
mov bl, byte[num3]
```

```
cmp bl, 0
```

```
jnz div0
```

```

mov byte[quotient], bl
jmp done1
div0:
div bl
mov byte[quotient], al
done1:
ret

```

```

toString:
; Part A - Successive division
mov rcx, 0
mov rbx, 10
divideLoop:
mov edx, 0
div rbx
push dx
inc rcx
cmp eax, 0
jne divideLoop
inc cl
mov byte[asciiLen], cl
dec cl

```

```

; Part B - Convert remainders and store
mov rbx, ascii
mov rdi, 0
popLoop:
pop ax
add al, "0"
mov byte [rbx+rdi], al
inc rdi
loop popLoop
mov byte[rbx+rdi], LF
print ascii, [asciiLen]
ret

```

[Attach Terminal Emulator window here:]

```
ryannishikawa@ryannishikawa-MacBookAir:~/cpssc240/final$ ./final
Input 1st number (0~9): 9
Input 2nd number (0~9): 9
Input 3rd number (0~9): 9
sum = 18
quotient = 2
ryannishikawa@ryannishikawa-MacBookAir:~/cpssc240/final$ ./final
Input 1st number (0~9): 6
Input 2nd number (0~9): 5
Input 3rd number (0~9): 0
sum = 11
quotient = 0
ryannishikawa@ryannishikawa-MacBookAir:~/cpssc240/final$ ./final
Input 1st number (0~9): 8
Input 2nd number (0~9): 9
Input 3rd number (0~9): 3
sum = 17
quotient = 5
ryannishikawa@ryannishikawa-MacBookAir:~/cpssc240/final$ ./final
Input 1st number (0~9): 7
Input 2nd number (0~9): 0
Input 3rd number (0~9): 2
sum = 7
quotient = 3
```