Computer Science Department
California State University, Fullerton

CPSC 240-09 Computer Organization and Assembly Language
Final Exam
9:00 AM to 10:15 AM
Tuesday, May 16, 2023

Student Name: Gustavo Couto Vanin

Last 4 digits of ID: 885517276

Note:

- University regulations on academic honesty will be strictly enforced.
- You have 75 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 quiz.

Final Exam

1. Download the “CPSC-240-09 Final Exam.docx” document.
2. Use x86-64 assembly language to implement the following C/C++ program.

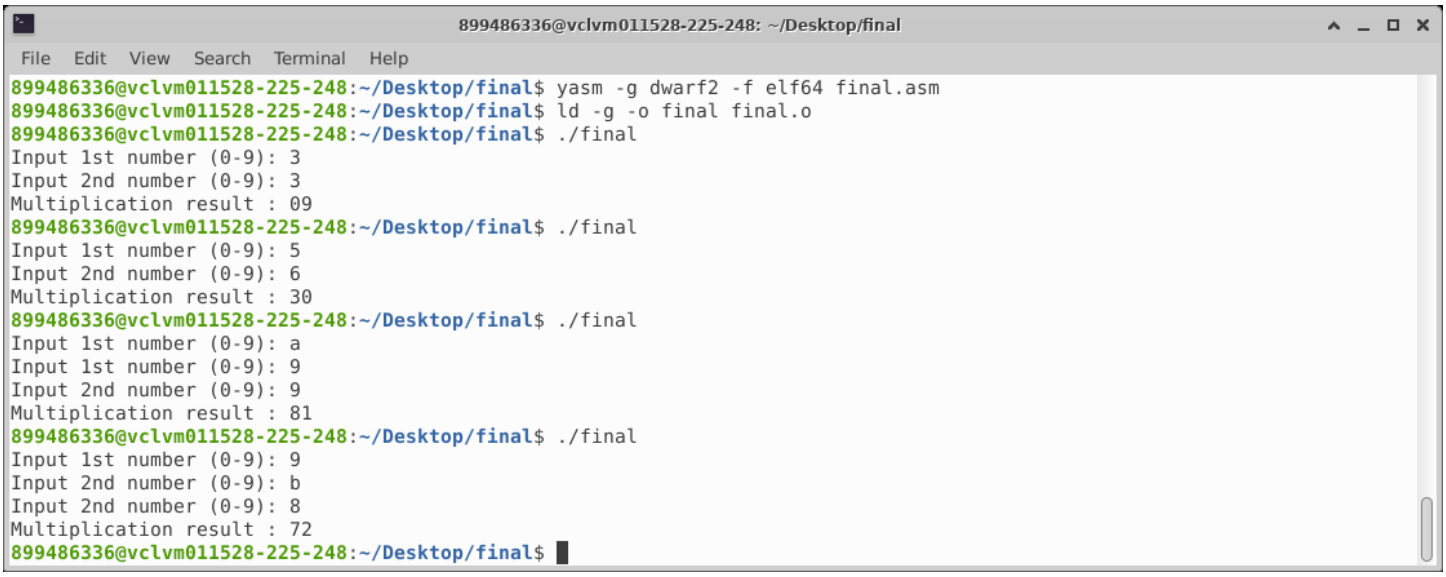
```
#begin define print(addr, n)
    rax = 1;
    rdi = 1;
    rsi = addr of string;
    rdx = n;
    syscall;
#end
#begin define scan(addr, n)
    rax = 1;
    rdi = 1;
    rsi = addr of buffer;
    rdx = n;
    syscall;
#end
char num1, num2, result;
char buf[2];
char msg1[24] = "Input 1st number (0~9): ";
char msg2[24] = "Input 2nd number (0~9): ";
char msg3[24] = "Multiplication result : ";
char ascii[3] = "00\n";
void main() {
    rbx = &msg1;
    call toNumber(rbx);
    num1 = al;
    rbx = &msg2;
    call toNumber(rbx);
    num2 = al;

    al = num1;
    bl = num2;
    call multiplication();
    result = al;

    di = short(result);
    call toAscii();
    cout << msg3;
    cout << ascii;
}
void toNumber(char[] message) {
    do {
      cout << message;
      cin >> buf;
    } while(buf < '0' && buf > '9');
    al = atoi(buf);
}
void multiplication() {
    ax = al * bl;
}
void toAscii() {
    ascii = itoa(result);
}
```

3. After assembling and linking, run the executable file to display the simulation results in the Terminal Emulator window as the following example.
4. Insert source code and the simulation results (Terminal Emulator window) to the bottom of the document.
5. Save the file in pdf or docx format and submit the pdf or docx file to Canvas before the deadline.
6. Deadline is 10:15 am on 05/16/2023.

Simulation result example:



[Attach your assembly source code here:]

```
%macro print 2
   mov rax, 1     ;SYS_write
   mov rdi, 1     ;standard output device
   mov rsi, %1       ;output string address
   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
   result resb 1
   buf resb 2
section .data
   msg1 db "Input 1st number (0~9): ", 24
   msg2 db "Input 2nd number (0~9): ", 24
   msg3 db "Multiplication result : ", 24
   ascii db "00\n"
section .text
   global _start
_start:
   print msg1, 24   ;cout << msg1
   scan num1, 2     ;cin >> buf
   mov rbx, num1    ;rbx = buf[0]
   call toNumber    ;toNumber() -> Convert rbx to number
   mov byte[num1], bl  ;Move rbx in buf[0]

   print msg2, 24   ;cout << msg1
   scan num2, 2     ;cin >> buf
   mov rbx, num2    ;rbx = buf[1]
   call toNumber    ;toNumber() -> Convert rbx to number
   mov byte[num2], bl  ;Move rbx in buf[1]


   xor rax, rax
```

```
   xor rbx, rbx

   mov al, byte[num1]
   mov bl, byte[num2]
   mul bl
   mov byte[result], al

   mov di, word[result]
   call toAscii

   print result, 1

   jmp exit

exit:
   mov rax, 60      ;terminate program
   mov rdi, 0    ;exit status
   syscall    ;calling system services

;ASCII to number function
toNumber:
   mov rax, 0    ;clear rax
   mov rdi, 10      ;rdi = 10

   mov cl, byte[rbx]   ;cl = [buffer+rsi]
   and cl, 0fh      ;convert ascii to number
   add al, cl    ;al = number
   adc ah, 0     ;ah = 0

   mov bl, al  ;num = ax
```

```
ret

toAscii:
   mov al, byte[result]
   mov bl, 10     ;bx = 10
next2:
   mov ah, 0      ;dx = 0
   div bl      ;dx=(dx:ax)%10, ax=(dx:ax)/10
   mov byte[result], al  ;ascii+0 = al + 30h

ret
```

[Attach Terminal Emulator window here:]

```
(gdb) x/ub &result
0x402052:       18
(gdb)
```

```
Input 1st number (0~9): 9
Input 2nd number (0~9): 2
```