**Final Exam**

**CS311 – Spring 2021**

This is a **closed** book exam and no lecture materials including EBE software, codes etc. are allowed to be used. Show all working steps where required**.** Partial credit is given for all questions.

**Honor Code Statement:** I pledge that this written exam is solely my work, and that I have neither given, nor received help from anyone.

Name: Tyler S. Unsworth ID: 42221732 Signature: Tyler S. Unsworth

**Question 1:**  15 Marks

Complete the following code segment which calculates the number of **1’s** in a word.

You are required to:

1. **while** loop which checks iteratively in a **word**
2. store the size in the **RCX** register
3. decrement down the **RCX** register for each loop
4. store the number of **1’s** in memory **sum**

**segment .data**

data dw 1011011011001010b ; there are 9 1’s && 7 0’s

sum dq 0

size db 16

**segment .text**

**global main**

main:

mov rcx, [data] ; moves the value of data into the rcx register

while\_loop: ; Declaration of while loop function

rol ax,1 ; rotate register to the left if it's 1

jc ignore ; jumps if >=

inc word[sum] ; increment the sum every 1 it sees

ignore: ; Declaration of ignore function

dec word[size] ; decrements the size lable

cmp word[size], 1 ; compares the word to 1

jnz while\_loop ; repeats the loop while it's not 0

**Question 2:**  15 Marks

(answer 7pt, working 8pt)

Given the following table and **4 KB** page size in a Windows machine, compute the logical address in base-16.

|  |  |  |
| --- | --- | --- |
| **Bits (LSB-MSB)** | **Table name** | **Size in decimal format** |
| 0 – 11 | page offset | 705 |
| 12 – 20 | page table index | 218 |
| 21 – 29 | page directory index | 383 |
| 30 – 38 | page directory pointer index | 328 |
| 39 – 47 | PML4 index | 0 |

**Answer: Shown on paper, submitted**

**Question 3:**  20 Marks

Given the following data segment, write code in main which replaces the value in memory a **directly** with the contents in memory b in in the 2nd byte (bolded) location.

**segment .data**

a dd 0x1234**56**78

b db 0xbb

**segment .text**

**global main**

main:

mov rax, [a] ; moves the value of a into rax register

mov rdx, [b] ; moves the value of b into rdx register.

mov rbx, 0xff ; creates an 8 bit mask

shl rbx, 8 ; shifts rbx register by 8

not rbx ; negates

and rax, rbx

shl rdx, 8 ; shifts rdx to the left 8

or rax, rdx

mov [a], rax ; moves rax into the value of a

**Question 4:**  25 Marks

Complete the following code segment which does a memory compare between **a** and **b.** The memory compare must be done in a function called memcmp, which is called from the **main** function.

The following must be done:

1. Set up proper stack frame including frame macro for both **main** and **memcmp** functions
2. **a** and **b** must be passed by **register** from **main** to **memcmp** function
3. memory compare occurs from left to right.
4. The difference between the *first non-equal* values must be passed back to main through appropriate register.
5. If the strings are equal, **-1** must be returned in RAX.
6. Appropriate string compare opcode must be used.

**segment .data**

a db "Final Exam!"

b db "Final Exan!"

**segment .text**

**global main**

**global memcmp**

memcmp:

str1 equ local1

str2 equ local2

push rbp

mov rbp, rsp

frame 1, 2, 0

sub rsp, frame\_size

mov [rbp + str1], rcx

mov [rbp + str2], rcx

mov rdi, [a]

mov rsi, [b]

mov rcx, 10

repe CMPSW

cmp rcx, 0

jz equal

movzx eax, byte[rdi-1]

movzx ecx, byte[rsi-1]

sub rax, rcx

ret

equal:

xor eax, eax

leave

ret

main:

push rbp

mov rbp, rsp

frame 0, 0, 2

sub rsp, frame\_size

call memcmp

ret

**Question 5:**  25 Marks

The Fibonacci equation can be given as: and the recurrence is whereas The Fibonacci sequence can then be given as the following: 1, 1, 2, 3, 5, 8, 13, 21, …

Write a code using an **iterative** function called **fib** to calculate the xth number (given in **x**) of the Fibonacci sequence and leave it in **result**. You **must** use the **stack** to store the numbers.

**segment .data**

x dq 6

result dq 0

**segment .text**

**global main**

**global fib**

main:

push rbp ; pushed to the stack

mov rbp, rsp ; moves the rsp register to rbp

frame 0,0,3 ; Declaration of the frame

sub rsp, frame\_size ; subtract the frame\_size from rsp register

lea rcx,[x] ; loads the value of x into rcx

; Declatation of fib call

call fib

leave ; copies rbp register to rsp register

ret

fib: ; Declaration of fib function which calculates the xth digit

digit equ local1 ; stores what’s in local1 to digit

push rbp ; pushes register to stack

mov rbp,rsp ; moves rsp register to rbp register

frame 1,1,1 ; decaration of new frame

sub rsp, frame\_size ; subtracts from frane\_size

**ran out of time**