Quiz 4-Machine Programming II

Top of Form

**Timothy Beckett (username: tdb2q)**

**Attempt 1**

Written: Jun 7, 2023 5:42 PM - Jun 7, 2023 6:12 PM

**Submission View**

Released: Jun 9, 2023 7:00 AM

|  |  |  |
| --- | --- | --- |
| **Question 1** |  | 2 / 2 points |

Consider the following assembly instructions and select the C function that generates it:

funcX:  
    leaq    (%rdi,%rdi,4), %rax  
    leaq    -10(,%rax,4), %rax  
    ret

|  |  |
| --- | --- |
|  | long func1(long n){     return n \*8 -10; } |
|  | long func1(long n){     return n \*20 -10; } |
|  | long func1(long n){     return n \*16-10; } |
|  | long func1(long n){     return n \*9-10; } |
| **Question 2** | |  | 2 / 4 points |

Match the C functions with their respective assembly codes.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | Incorrect Response | \_\_2\_\_ | **(4)** | movzbl    4(%rdi), %eax     ret | |  | \_\_1\_\_ |  | movl    16(%rdi), %eax     ret | |  | \_\_3\_\_ |  | movl    %edi, %eax     sall    $4, %eax     ret | | Incorrect Response | \_\_4\_\_ | **(2)** | leal    0(,%rdi,4), %eax     ret | |  | |  |  | | --- | --- | | **1**. | int fun1(int \*a) {      return a[4]; } | | **2**. | int fun2(int a) {      return a\*4;  } | | **3**. | int fun3(int a) {      return a<<4;  } | | **4**. | char fun4(char \*a) {      return a[4];  } | | |
| **Question 3** | |  | 4 / 4 points | |

Consider the following assembly instructions and select the C function that generates it:

functionX:  
    imulq    %rsi, %rdi  
    imulq    %rcx, %rdx  
    subq    %rdx, %rdi  
    movq    %rdi, %rax  
    salq    $6, %rax  
    ret

|  |  |
| --- | --- |
|  | long functionA(long a, long b, long c, long d){       long e = a\*b;      long f = c\*d;                      return (f-e)\*6;  } |
|  | long functionA(long a, long b, long c, long d){       long e = a\*b;      long f = c\*d;                      return (f-e)\*64;  } |
|  | long functionX(long a, long b, long c, long d){       long e = a\*b;      long f = c\*d;                      return (e-f)\*64;  } |
|  | long functionX(long a, long b, long c, long d){       long e = a\*b;      long f = c\*d;                      return (e-f)\*6;  } |
| **Question 4** | |  | 2 / 2 points |

"ja" and "jb" instructions are used for jumping if the last "flag changing" instruction was on unsigned numbers.

|  |  |  |
| --- | --- | --- |
|  |  | True |
|  |  | False |
| **Question 5** | | |  | 2 / 4 points |

Reconstruct the C code by looking at the assembly code below:

**Make sure not to put any space characters in your answers.**

myfunction:  
    movq    $1, %rax  
    cmpq    $1, %rdi  
    ja    .L8  
    ret  
.L8:  
    subq    $8, %rsp  
    shrq    %rdi  
    call    myfunction  
    addq    $1, %rax  
    addq    $8, %rsp  
    ret

unsigned myfunction2(unsigned n){        
    if (\_\_\_\_\_blank1\_\_\_\_\_\_\_)   
        return 1;    
      
    else {          
        return 1 + myfunction2(\_\_\_\_blank2\_\_\_\_);       
    }   
}

blank1:

\_\_\_n==1\_\_\_Incorrect Response**(/n<=1/)**

blank2:

\_\_\_n/2\_\_\_(50 %)

|  |  |  |
| --- | --- | --- |
| **Question 6** |  | 1 / 1 point |

In C, conditional moves are used because they do not require control transfer and branches are very disruptive to instruction flow.

|  |  |  |
| --- | --- | --- |
|  |  | True |
|  |  | False |
| **Question 7** | | |  | 2 / 2 points |

GCC converts the branching code into conditional moves because they are always safe to use.

|  |  |  |
| --- | --- | --- |
|  |  | True |
|  |  | False |
| **Question 8** | | |  | 0 / 1 point |

register r12 is a caller-saved register.

|  |  |  |
| --- | --- | --- |
| Incorrect Response |  | True |
| Correct Answer |  | False |
|  |  | |
|  | | |

|  |
| --- |
| **Attempt Score:**  15 / 20 |
| **Overall Grade (last attempt):**  14 / 20 |

|  |
| --- |
|  |

Done

|  |
| --- |
|  |

Bottom of Form