CSPB 2400 - Park - Computer Systems

Started on	Saturday, 10 February 2024, 8:57 PM
State	Finished
Completed on	Saturday, 10 February 2024, 9:11 PM
Time taken	14 mins 34 secs
Marks	12.00/12.00
Grade	**10.00** out of 10.00 (**100**%)

Question **1**

Correct

Mark 1.00 out of 1.00

The assembly code on the right partially implements the C function shown on the left. Fill in the missing instruction to correctly implement the C function on the left.

```
int a,b,x,y;
int foo() {
  if (a < b) {
    return x;
   } else {
     return y;
   }
}
```

```
foo:
   movl    y, %eax
   movl    b, %edx
   cmpl    %edx, a
   jge  ✔      L3
   movl    x, %eax
L3:
   ret
```

ja | jl | js | jne | jle | jb | jns | je | jg | jae | jbe

x | y | c | b | z | a

!= | <= | == | < | >= | >

Your answer is correct.

Question **2**

Correct

Mark 1.00 out of 1.00

The assembly code on the right partially implements the C function shown on the left. Fill in the missing instruction to correctly implement the C function on the left.

```
int a,b,x,y;
int foo() {
  if (a == b) {
    return x;
   } else {
     return y;
   }
}
```

```
foo:
   movl    y, %eax
   movl    b, %edx
   cmpl    %edx, a
   jne  ✔      L3
   movl    x, %eax
L3:
   ret
```

jl | jb | js | ja | jle | je | jge | jae | jbe | jg | jns

b | a | y | x | z | c

!= | == | < | <= | >= | >

Your answer is correct.

Question **3**

Correct

Mark 1.00 out of 1.00

The assembly code on the right partially implements the C function shown on the left. Fill in the missing instruction to correctly implement the C function on the left.

```
int a,b,x,y;
int foo() {
  if (a <= ✔ b) {
    return x;
   } else {
     return y;
   }
}
```

```
foo:
   movl    y, %eax
   movl    b, %edx
   cmpl    %edx, a
   jg     L3
   movl    x, %eax
L3:
   ret
```

jg | jge | jbe | jns | jle | jae | jb | js | je | jne | jl | ja

b | c | a | z | y | x

>= | > | == | != | <

Your answer is correct.

Question **4**

Correct

Mark 1.00 out of 1.00

The assembly code on the right partially implements the C function shown on the left. Fill in the missing instruction to correctly implement the C function on the left.

```
unsigned int a,b,'
int x,y;
int foo() {
  if (a >= b) {
    return x;
   } else {
     return y;
   }
}
```

```
foo:
   movl    y, %eax
   movl    b, %edx
   cmpl    %edx, a
   jb  ✔       L3
   movl    x, %eax
L3:
   ret
```

jg | je | jge | jae | jle | jns | jl | ja | jne | jbe | js

x | a | b | y | z | c

== | <= | != | >= | > | <

Your answer is correct.

Question **5**

Correct

Mark 1.00 out of 1.00

The assembly code on the right partially implements the C function shown on the left. Fill in the missing instruction to correctly implement the C function on the left.

```
int foo(a) {
    if (a <= 5) {
        return -10;
    }
    else {
        return 10;
    }
}
```

```
foo:
   cmpl $0x5, %edi
   jg  ✔  L1
   mov  $0xfffffff6, %eax
   jmp   L2
L1:
   mov $0xa, %eax
L2:
   retq
```

jne | jae | jge | js | jns | jl | jbe | jle jb | je | ja

b | x | y | c | z | a

>= | <= | > | == | < | !=

Your answer is correct.

Question **6**

Correct

Mark 1.00 out of 1.00

The assembly code on the right partially implements the C function shown on the left. Fill in the missing instruction to correctly implement the C function on the left.

```
int
```

```
foo(long a) {
    if (a > 0) {
        return 0;
    }
    else {
        return 1;
    }
```

```
    testl %rdi,%rdi
    jle   L1
    movl  $0x0, %eax
    jmp   L2
L1:
    movl  $0x1, %eax
L2:
    retq
```

jns | jge | jg | jae | jbe | jne | js | ja | jl | jb | je

b | c | a | y | z | x

> | != | <= | >= | == | <

Your answer is correct.

Question **7**

Correct

Mark 1.00 out of 1.00

The assembly code on the right partially implements the C function shown on the left. Fill in the missing instruction to correctly implement the C function on the left.

```
int sum = 2;
int i;
int foo() {
   i = 0;
   while (i < 5){
      sum = sum + 3;
      i++;
   }
return sum;
}
```

```
foo:
        movl    $0, i
        jmp     .L2
.L3:
        movl    sum, %eax
        addl    $3, %eax
        movl    %eax, sum
        movl    i, %eax
        incl    %eax
        movl    %eax, i
.L2:
        movl    i, %eax
        cmpl    $4 ✔ , %eax
        jle     .L3
        movl    sum, %eax
        ret
```

$3 $2 $5 $6 $1

Your answer is correct.

Question **8**

Correct

Mark 1.00 out of 1.00

The assembly code on the right partially implements the C function shown on the left. Fill in the missing instruction to correctly implement the C function on the left.

```
int sum = 2;
int i;
int foo() {

for (i = 0; i < 5; i++)
{
    sum = sum + 3;
}

return sum;
}
```

```
foo:
        movl    $0, i
        jmp     .L2
.L3:
        movl    sum, %eax
        addl    $3, %eax
        movl    %eax, sum
        movl    i, %eax
        addl    $1, %eax
        movl    %eax, i
.L2:
        movl    i, %eax
        cmpl    $4 ✔ , %eax
        jle     .L3
        movl    sum, %eax
```

$3 $6 $5 $1 $2

Your answer is correct.

Question **9**

Correct

Mark 1.00 out of 1.00

The assembly code on the right partially implements the C function shown on the left. Fill in the missing instruction to correctly implement the C function on the left.

```
int sum = 2;
int i;
int foo() {

for (i = 0; i < [3] ✔ ; i++)
{
    sum = sum + 3;
}

return sum;
}
```

```
foo:
        movl    $0, i
        jmp     .L2
.L3:
        movl    sum, %eax
        addl    $3, %eax
        movl    %eax, sum
        movl    i, %eax
        addl    $1, %eax
        movl    %eax, i
.L2:
        movl    i, %eax
        cmpl    $2, %eax
        jle     .L3
        movl    sum, %eax
```

[6] [5] [1] [2] [4]

Your answer is correct.

Question **10**

Correct

Mark 1.00 out of 1.00

In a C function, we have switch statement with four cases.

Given the assembly code and jump table below, what is the value of the largest **case** in the switch statement? 18 ✔ (enter the value in base-10 format).

```
movq    8(%rbp), %rax
subq    $15, %rax
cmpq    $3,  %rax
ja      .L2
jmp     .L6(, %rax,4)
```

```
.L6
  .quad .L3
  .quad .L2
  .quad .L4
  .quad .L5
```

Question **11**

Correct

Mark 1.00 out of 1.00

Write a C function **func** that performs the actions of the following assembly code:

```
func:
        cmpl    %edi, %esi
        jge     .L5
        xorl    %eax, %eax
        ret
.L5:
        setg    %al
        movzbl  %al, %eax
        ret
```

For example:

Test	Result
`test(5,4);`	`OK1`

Answer: (penalty regime: 0 %)

```
int func(int a, int b) {
    if (a > b) {
        return 0;
    }
    else {
        return 1;
    }
}
```

	Test	Expected	Got	
✔	`test(5,4);`	`OK1`	`OK1`	✔
✔	`test(rand(), rand());`	`OK2`	`OK2`	✔
✔	`test(rand(), rand());`	`OK3`	`OK3`	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **12**

Correct

Mark 1.00 out of 1.00

Write a C function **func** that performs the actions of the following assembly code. Your function must be named **func**.

The function takes four arguments passed in registers **%edi, %esi, %edx and %rcx**.

The function returns a single argument in **%eax**.

```
func:
        movl    %edi, %eax
.L3:
        movslq  %eax, %rdi
        cmpl    %esi, (%rcx,%rdi,4)
        je      .L2
        addl    $1, %eax
        cmpl    %edx, %eax
        jl      .L3
        movl    $-1, %eax
.L2:
        ret
```

For example:

Test	Result
`test(1,target1);`	`OK1`

Answer: (penalty regime: 10, 20, ... %)

```
int func(int index, int value, int limit, int *array) {
    for (; index < limit; ++index) {
        if (array[index] == value) {
            return index;
        }
    }
    return -1;
}
```

	Test	Expected	Got	
✔	`test(1,target1);`	`OK1`	`OK1`	✔
✔	`test(2,target2);`	`OK2`	`OK2`	✔
✔	`test(3,target3);`	`OK3`	`OK3`	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.