hw6: Assembly Language 2 Results for VARDAAN KAPOOR (He/him)

Score for this attempt: **8** out of 8
Submitted Apr 17 at 8:30pm
This attempt took 23 minutes.

Question 1

1 / 1 pts

```
<doubleIt>:
        pushl %ebp
        movl %esp, %ebp
        pushl %ebx
        pushl %esi
        movl X(%esp), %ecx
        addl %ecx, %ecx
        movl %ecx, %eax
        popl %ecx
        popl %ebx
        leave
        ret
```

```
int doubleIt(int x){
        return x + x;
}
```

The value of **X** that would result in the correct execution of `doubleIt` is:

Correct!

16

orrect Answers

16 (with margin: 0)

Question 2

1 / 1 pts

Consider the following assembly code:

```
movl 0x8(%ebp),%eax
movl (%eax),%edx
movl 0xc(%ebp),%eax
movl (%eax),%eax
```

```
addl %edx,%eax
movl 0xc(%ebp),%edx
movl %eax,(%edx)
```

If `0x8(%ebp)` and `0xc(%ebp)` refer to the function arguments `a` and `b` respectively, which one of the following choices represent the C equivalent of the assembly code above?

```
void func(int* a, int* b) {
    int tmp = *b
    *b = *a;
    *a = tmp;
    return;
}
```

```
void func(int* a, int* b) {
    a = b + a;
    return;
}
```

```
void func(int* a, int* b) {
    b = a + b;
    return;
}
```

```
void func(int* a, int* b) {
    *a = *a + *b;
    return;
}
```

Correct!

```
void func(int* a, int* b) {
    *b = *a + *b;
    return;
}
```

Question 3

1 / 1 pts

If `P` calls `G(x, y)`, then which of the following statements below are correct ?

1. Arguments passed to G are stored in G's stack frame.

2. Return address to P is stored in G's stack frame.
3. G can safely overwrite %eax, %edx or %ecx without saving their data first.
4. G cannot safely overwrite %ebx, %esi or %edi without saving their data first.

- 1 and 3
- 1, 2, and 3

Correct!

- 3 and 4
- 2 and 3
- 1, 2, 3, and 4
- 2 and 4

Question 4

1 / 1 pts

```
<func>:
        pushl %ebp
        movl %esp, %ebp
        subl $16, %esp
        movl 8(%ebp), %ecx
        movl 12(%ebp), %edx
        movl $0, %eax
        cmpl %ecx, %edx
        jge L1
        movl $1, %eax
L1:
        leave
        ret
```

Which one of the comparisons below does the call `func(x,y)` compute?

- y <= x

Correct!

- y < x
- x < y

x <= y

y == x

Question 5

1 / 1 pts

Consider the following function where **X**, **Y**, **Z**, and **W** represent incomplete code.

```
int math(int a, int b, int c) {
   int res1 = X;
   int res2 = Y;
   int res3 = Z;
   int res4 = W;
   return res4;
}
```

The function above is implemented in assembly code below where the function's parameters `a`, `b` and `c` are at the effective addresses `0x8(%ebp)`, `0xC(%ebp)` and `0x10(%ebp)`, respectively.

```
movl 0xC(%ebp),%eax
notl %eax
notl %eax
movl %eax,-0x10(%ebp)
movl 0x10(%ebp),%eax
movl 0xC(%ebp),%edx
addl %edx,%eax
movl %eax,-0xC(%ebp)
movl 0x10(%ebp),%eax
imull $11,%eax
movl %eax,-0x8(%ebp)
movl -0xC(%ebp),%eax
sall $0x5,%eax
movl %eax,-0x4(%ebp)
```

Choose the correct options for **X**, **Y**, **Z** and **W** to complete the function so that it corresponds with its assembly code.

- **X** is [Select]
- **Y** is [Select]
- **Z** is [Select]
- **W** is [Select]

Answer 1:

Correct!

b

Answer 2:

Correct!

b + c

Answer 3:

Correct!

c * 11

Answer 4:

Correct!

res2 << 5

Question 6

1 / 1 pts

Consider the following funtion:

```
int recursive(int n) {...}
```

The assembly code equivalent of the above function is:

```
recursive:
           pushl %ebp
           movl %esp,%ebp
           pushl %ebx
           subl $0x14,%esp
           cmpl $0x1,0x8(%ebp)
           je .L1
           cmpl $0x2,0x8(%ebp)
           jne .L2
.L1:
           movl 0x8(%ebp),%eax
           jmp .L3
.L2:
           movl 0x8(%ebp),%eax
           subl $0x1,%eax
           movl %eax,(%esp)
           call recursive
           movl %eax,%ebx
           movl 0x8(%ebp),%eax
           subl $0x2,%eax
           movl %eax,(%esp)
           call recursive
           imul %ebx,%eax
.L3:
           addl $0x14,%esp
           popl %ebx
```

```
            popl %ebp
            ret
```

What would be the values returned for the code below?

```
int ret_val_1 = recursive(1);
int ret_val_2 = recursive(2);
```

- ret_val_1 is 0 and ret_val_2 is 1
- ret_val_1 is 1 and ret_val_2 is 1

Correct!

- ret_val_1 is 1 and ret_val_2 is 2
- ret_val_1 is 1 and ret_val_2 is 0
- ret_val_1 is 2 and ret_val_2 is 1

Question 7

1 / 1 pts

Consider the following funtion:

```
int recursive(int n) {...}
```

The assembly code equivalent of the above function is:

```
recursive:
            pushl %ebp
            movl %esp,%ebp
            pushl %ebx
            subl $0x14,%esp
            cmpl $0x1,0x8(%ebp)
            je .L1
            cmpl $0x2,0x8(%ebp)
            jne .L2
.L1:
            movl 0x8(%ebp),%eax
            subl $1,%eax
            jmp .L3
.L2:
            movl 0x8(%ebp),%eax
            subl $0x1,%eax
            movl %eax,(%esp)
            call recursive
            movl %eax,%ebx
            movl 0x8(%ebp),%eax
            subl $0x2,%eax
```

```
            movl %eax,(%esp)
            call recursive
            addl %ebx,%eax
.L3:
            addl $0x14,%esp
            popl %ebx
            popl %ebp
            ret
```

What would be the value returned by the code below?

```
int return_val = recursive(5);
```

(Hint: Consider determining and then tracing the equivalent code in C.)

- 4
- 8
- 6
- 3 — Correct!
- 10

Question 8

1 / 1 pts

```
<func>:
        pushl %ebp
        movl %esp, %ebp
        subl $16, %esp
        movl 8(%ebp), %ecx
        movl 12(%ebp), %edx
        movl $0, %eax
        cmpl %ecx, %edx
        jle L1
        movl $1, %eax
L1:
        leave
        ret
```

Which one of the following assembly code fragments is equivalent to the above assembly code fragment?

```
<func>:
        pushl %ebp
        movl %esp, %ebp
        subl $16, %esp
        movl 8(%ebp), %ecx
        movl 12(%ebp), %edx
        movl $0, %eax
        subl %ecx, %edx
        cmpl %edx, $0
        jl L1
        movl $1, %eax
L1:
        leave
        ret
```

Correct!

```
<func>:
        pushl %ebp
        movl %esp, %ebp
        subl $16, %esp
        movl 8(%ebp), %ecx
        movl 12(%ebp), %edx
        movl $0, %eax
        subl %ecx, %edx
        cmpl %edx, $0
        jge L1
        movl $1, %eax
L1:
        leave
        ret
```

```
<func>:
        pushl %ebp
        movl %esp, %ebp
        subl $16, %esp
        movl 8(%ebp), %ecx
        movl 12(%ebp), %edx
        movl $1, %eax
        subl %ecx, %edx
        cmpl %edx, $0
        jge L1
        movl $0, %eax
L1:
        leave
        ret
```

```
<func>:
        pushl %ebp
        movl %esp, %ebp
        subl $16, %esp
        movl 8(%ebp), %ecx
        movl 12(%ebp), %edx
        movl $0, %eax
        subl %edx, %ecx
        cmpl %ecx, $0
        jge L1
        movl $1, %eax
L1:
        leave
        ret
```

```
<func>:
        pushl %ebp
        movl %esp, %ebp
        subl $16, %esp
        movl 8(%ebp), %ecx
        movl 12(%ebp), %edx
        movl $0, %eax
        subl %edx, %ecx
        cmpl %ecx, $0
        jl L1
        movl $1, %eax
L1:
        leave
        ret
```

Quiz Score: **8** out of 8