hw5: Assembly Language 1

Due Apr 10 at 11:59pm **Points** 8 **Questions** 8
Available Apr 5 at 12am - Apr 11 at 11:59pm **Time Limit** 40 Minutes
Allowed Attempts 2

Instructions

There is one question that requires knowing how cmp and jumps work. L19 will cover these instructions.
It is worth trying at least one of your hw5 quiz attempts before Lecture L19.
Suggestions:

- Try at least one attempt on this homework WITHOUT using this reference.
- You may use the reference if you wish, but it won't be available for exams.

This quiz was locked Apr 11 at 11:59pm.

Attempt History

	Attempt	Time	Score
KEPT	Attempt 2	6 minutes	8 out of 8
LATEST	Attempt 2	6 minutes	8 out of 8
	Attempt 1	13 minutes	7 out of 8

Score for this attempt: **8** out of 8
Submitted Apr 10 at 8:15pm
This attempt took 6 minutes.

Question 1

1 / 1 pts

Assume the initial value for registers: `%eax` = 37, `%ebx` = 73, `%ecx` = 0, `%esp` = 0x800, and initial value stored at address 0x800 is 73. Which one of the following sequences of assembly instructions would store a value of 37 at address 0x800 and a value of 73 in the register `%ecx`?

Correct!

[✓ `popl %ecx, pushl %eax, pushl %ebx`]

`pushl %eax, pushl %ebx, popl %ecx`

`popl %ecx, pushl %ebx, pushl %eax`

`pushl %eax, popl %ecx, pushl %eax`

Question 2

1 / 1 pts

Which of the following instructions are valid?

1. `subl (%esp), (%edx)`
2. `subw %eax, $0x108`
3. `subb %ah, %dh`
4. `addl %eax, %ebx, %ecx`
5. `addl 0x13(,%edi,4), %esi`

2, 3 and 5

1, 2 and 4

1, 2, 4 and 5

1, 2, 3 and 5

Correct!

[✓ 3 and 5]

Question 3

1 / 1 pts

Consider the following assembly code:

```
pushl %ebp
movl %esp, %ebp
subl $0x40, %esp
movl %ebx, 0x14(%esp)
movl $1, %ebx
```

Which one of the choices below is able to undo the effects of the assembly code above?

( )
```
popl %ebp
movl %ebp, %esp
movl -0x26(%ebp), %ebx
addl $0x40, %esp
```

Correct!

[✓]
```
movl 0x14(%esp), %ebx
movl %ebp, %esp
popl %ebp
```

( )
```
popl %ebp
movl %ebp, %esp
addl $0x40, %esp
movl 0x14(%esp), %ebx
```

( )
```
movl -0x26(%ebp), %ebx
addl $0x40, %esp
movl %ebp, %esp
popl %ebp
```

( )
```
movl 14(%esp), %ebx
addl $40, %esp
movl %ebp, %esp
popl %ebp
```

Question 4

1 / 1 pts

Variables a and b are stored at -0x8(%ebp) and -0x4(%ebp) respectively.

```
movl -0x4(%ebp), %eax
movl (%eax), %edx
movl -0x8(%ebp), %eax
```

```
addl %eax, %edx
movl %edx, -0x8(%ebp)
```

Chose **X** and **Y** such that the following C statement is equivalent to the assembly code above:

```
a = X + Y;
```

- X = a and Y = b
- X = a and Y = &b
- X = *a and Y = *b
- Correct! [✓ X = a and Y = *b]
- X = *a and Y = b

Question 5

1 / 1 pts

Consider the following assembly instruction:

```
leal (%ecx,%edx,2), %eax
```

The values stored in registers `%ecx` and `%edx` are 0x200 and 0x100, respecively. The value at address 0x400 is 0x1, 0x401 is 0x2, 0x402 is 0x3 and 0x404 is 0x4.

What would be the final value of `%eax`?

- 0x2
- 0x3
- 0x500
- Correct! [✓ 0x400]

○ 0x1

Question 6

1 / 1 pts

Select **ALL** the operand specifiers that produce an effective address of `0x114`.

Assume that the initial values of `%ecx` and `%edx` are `0x100` and `0x4`, respectively.

- ☐ `(%ecx,%edx,5)`
- Correct! ☑ `0x14(%ecx)`
- Correct! ☑ `0x4(%ecx,%edx,4)`
- Correct! ☑ `0x114`
- ☐ `0x5(%ecx,%edx)`

Question 7

1 / 1 pts

Consider the following assembly code:

```
loop_func:
        pushl %ebp
        movl %esp, %ebp
        subl $16, %esp
        movl $0, -4(%ebp)
        jmp .L2
.L3:
        movl 8(%ebp), %eax
        addl %eax, -4(%ebp)
        subl $1, 8(%ebp)
.L2:
        cmpl $2, 8(%ebp)
        jg .L3
        movl -4(%ebp), %eax
        leave
        ret
```

If `-4(%ebp)` corresponds to local variable `sum` and `8(%ebp)` corresponds to function argument `n`, which one of the choices below is the correct C equivalent of the assembly code above?

Correct!

[✓]
```
int func(int n){
  int sum = 0;

  while (n > 2){
    sum = sum + n;
    n--;
  }

  return sum;
}
```

```
int func(int n){
  int sum = 0;

  while (n > 2){
    n--;
    sum += n;
  }

  return sum;
}
```

```
int func(int n){
  int sum = 0;

  do {
    sum = sum + n;
    n--;
  } while (n > 2);

  return sum;
}
```

```
int func(int n){
  int sum = 0;

  if (n > 2){
    sum += n;
    n--;
  }

  return sum;
}
```

Question 8

1 / 1 pts

Select **ALL** the assembly instructions that can be used to set the register `%ebx` to zero.

- [ ] `sall $34, %ebx`
- [✓] `xorl %ebx, %ebx` Correct!
- [ ] `orl $0, %ebx`
- [✓] `andl $0, %ebx` Correct!
- [✓] `movl $0, %ebx` Correct!

Quiz Score: **8** out of 8