

Do your quiz in one of the following Odin files...

**2240/b/quiz11.txt** or **2240/b/quiz11.jpg**

1. x86 in-line assembly for the GCC Compiler follows which syntax standard?

- A. Intel
- B. PG&E
- C. AT&T
- D. PB&J

2. Look at the two lines of code below. The printf function does not have a responsibility, by convention, to protect the rcx register. What instruction should you write following the call to printf to restore the rcx register to the state prior to the printf call?

```
push rcx
call printf
```

3. In x86 in-line assembly, what does the following statement do?

```
asm ("movl %%eax, %%ebx");
```

- A. copies ebx's contents into eax.
- B. copies eax's contents into ebx.
- C. copies bx's contents to ax.
- D. moves the letter 'l' into eax and ebx.

4. Write a complete x86 in-line assembly statement that will move 25 into register **rcx**. A complete statement please.

5. What is a clobber list?

- A. Unused registers.
- B. Registers that are available for use.
- C. The output constraints.
- D. Registers that your in-line code will change.

6. If the bits in an x86 AH register are shifted two to the left, which other registers might be affected.

- A. AL only.
- B. AX and EAX only.
- C. AX, EAX, and RAX only.
- D. No other registers will be affected.

7. What is true about the following x86 in-line assembly code?

```
asm ( "movl %2, %%eax;"
      "movl %%eax, %0;"
      : "=r"(sum)
      : "r"(x), "r"(y)
      : "%eax"
    );
```

- A. Variable `y` is copied to variable `sum`.
- B. The value of variable `x` is placed in register `eax`.
- C. The variable `sum` is an input variable.
- D. Zero is moved to the `eax` register.

8. The following x86 in-line assembly code is designed to copy one register to another, but it is not correct. Rewrite the statement correctly, while maintaining the intended operation.

```
asm ("movl %%rax, %%rbx");
```

9. Look at the following x86 statements. Just after the statements execute, what value will the `EAX` register hold.

```
xor EAX, EAX    ; zero the EAX register
mov AL, 128     ; move 128 to AL register
shl EAX, 1      ; bit-shift EAX register one to the left
```

10. What do these two statements together accomplish?

```
push eax
pop ecx
```

- A. puts `ecx` on the stack
- B. puts `eax` on the stack
- C. copies `eax` to `ecx`
- D. copies `ecx` to `eax`