Short Answer Questions (12 points)

*Please respond to the following in complete sentences. For full credit, answers should not be longer than **two sentences**.*

When Prof. Zilles says "expressibility is one of the main reasons that people write code using assembly language", what does he mean? (4 points)

Assembly language is more expressible than high-level language in that there are some things that you can't write in HLLs that you can in assembly language. For example, all of the move to/from coprocessor 0 instructions in the interrupt handlers are not something that you could write in most high level languages.

What is the advantage of using interrupt-based I/O over polling? (4 points)

With interrupt-based I/O, the processor can do something else while it is waiting. With polling, the processor has to periodically query the I/O device, which can make it hard to accomplish anything while waiting.

Why shouldn't programmers use the assembler temporary when programming in assembly? (4 points)

The assembler temporary ($at) is used by the assembler to implement pseudo-instructions. The assembler will not save and restore the current value of $at when it uses it, so if you put a value in $at that you care about, it will be lost when the assembler uses $at.

Multiple Choice (13 points)

TRUE / FALSE (A = True, B = False)

1. An abstraction layer is a way of hiding the implementation details of a particular set of functionalit **True**
2. A MIPS pseudo-instruction is an instruction that is implemented in hardware on some MIPS pro **False** designs and in software on other MIPS processor designs.
3. A pointer to an integer is four bytes long, but a pointer to a character is only one byte **False, all ptrs are 4B**
4. We use the `jump-and-link (jal)` instruction for function calls because `jal` records the r address for use by the called function. **True**
5. The Status register doesn't need to be written; bits in the Status register are automatically cleared when you acknowledge interrupts. **False. Is a true statement for the Cause register**

6. Assuming 32-bit integers and that all data types need to be naturally aligned, how large is the following data structure: (a) 4 bytes, (b) 9 bytes, (c) 12 bytes, (d) 13 bytes, (e) 16 bytes

```
struct my_struct {
   int A;
   char b;
   int C[2];
};
```

7. Indicate which of the following statements is **false**, or select (e) if all statements are true.

   a. Advocates of RISC machines thought they'd achieve higher performance through higher clock frequencies.
   b. Advocates of CISC machines thought they'd achieve higher performance because their machines would get more done each cycle.
   c. RISC machines were developed in the 1960's because their simple instructions made assembly programming simpler. **False**
   d. A common differentiation between CISC and RISC machines is that many CISC machines permit retrieving instruction operands from memory.

8. Assembling is the step of the compilation process where:

   a. high-level language code is "assembled" to produce assembly code.
   b. assembly code is "assembled" to produce machine code.
   c. multiple object files are "assembled" to create a single executable.

9. The 90/10 rule is:

   a. Computer architects should focus on optimizing things that represent 90% of the execution time, ignoring the remaining 10%.
   b. Processors spend 90% of the time executing 10% of the code.
   c. Programmers spend 90% of their time debugging 10% of the code.
   d. Real instructions should be used 90% of the time, using pseudo-instructions only 10% of the time.
   e. Caller-saved registers should be used 90% of the time, and callee-saved registers the other 10% of the time.

10. Indicate which of the following statements is **false**, or select (e) if all statements are true.

    a. Exceptions are program errors; it is not necessary to be able to recover the program state after an exception.
    b. Interrupts don't need to be handled right away, they can be deferred for a few cycles.
    c. Exceptions are associated with a particular executing instruction.
    d. Additional interrupts may become pending when the program is already in the interrupt handler.

Multiple Choice, cont.

11-13. Consider the following piece of code and indicate where in memory each of the data structures will be allocated.

```
int var = 0;                                          // question 11  c

void
the_best_function_evar(int my_argument) {
  int place_to_store_stuff[64];                       // question 12  a
  ...
  int *I_need_more_space =
       (int *) malloc(my_argument * sizeof(int));     // question 13  b
}
```