Some examples related to sequential consistency


1. What might p2 return if run concurrently with p1?
   ```c
   int data = 0, ready = 0;
   void p1 () {
       data = 2000;
       ready = 1;
   }
   int p2 () {
       while (!ready) {}  
       return data;
   }
   ```

2. Can both critical sections run?
   ```c
   int flag1 = 0, flag2 = 0;
   int main () {
       tid id = thread_create (p1, NULL);
       p2 (); thread_join (id);
   }
   void p1 (void *ignored) {
       flag1 = 1;
       if (!flag2) {
           critical_section_1 ();
       }
   }
   void p2 (void *ignored) {
       flag2 = 1;
       if (!flag1) {
           critical_section_2 ();
       }
   }
   ```

3. If a processor can read its own writes early, then both functions below can return 2:
   ```c
   /*
   * keyword "register" tells compiler to place the variable in a
   * register, not on the stack.
   */
   int flag1 = 0, flag2 = 0;
   int p1 (void *ignored)       int p2 (void *ignored)
       {
       register int f, g;     register int f, g;
       flag1 = 1;             flag2 = 1;
       f = flag1;             f = flag2;
       g = flag2;             g = flag1;
       return 2*f + g;        return 2*f + g;
   ```