Quiz 1: disabling interrupts

Considering only the issue of scheduling multiple processes/threads in a multiprogramming environment…

… what's so bad about disabling interrupts in order to synchronize operations?

Disabling interrupts would prevent timer interrupts that the scheduler depends on to regain control.

- Absent interrupts, priority scheduling cannot be guaranteed.
- This could disrupt the operation of entirely unrelated processes.
- Programmer errors could hang entire system.
- May lose critical or non-critical device-generated interrupts.

Quiz 2: busy-waiting

Explain what is objectionable about the original implementation of the Pintos function `timer_sleep()` shown below:

```
// Suspends execution for approximately TICKS timer ticks.
void
timer_sleep (int64_t ticks) {

  int64_t start = timer_ticks ();

  ASSERT (intr_get_level () == INTR_ON);
  while (timer_elapsed (start) < ticks)
    thread_yield ();
}
```

Quiz 3: Dynamic Partitions vs Paging

What is the fundamental difference between

- dynamic partition memory management
- pure paging (w/o virtual memory)

With dynamic partition management, the process is loaded into contiguous storage, leading to fragmentation and limiting the level of multitasking.

With paging, the process is loaded into frames which need not have any particular spatial relationship, eliminating external fragmentation and the limit on the level of multitasking.

Quiz 4: Address Translation

Suppose that we have paged memory management with 4 KiB pages.

Assume the following logical address:

```
ABDC EFGH
```

What can be said about the following?

- page #: `ABCDE`
- frame #: Nothing without knowing the relevant page table entry.
- offset: `FGH`

Quiz 5: Page Replacement

Suppose that we have demand-paged virtual memory, and that a local page replacement policy is used. Suppose also that a process P requests a non-resident page.

If the optimal page replacement policy were used, what resident page would be selected for replacement?

The optimal replacement policy will replace the resident page whose next access is farthest into the future (which is, of course, impractical).

Since the policy is applied locally, the replaced page will be the resident page that belongs to the requesting process and whose next access is farthest in the future.

Quiz 6: Page Size

When paging is used, the page size is always chosen to be a power of 2.

Explain precisely why this is true.

Be sure to address whether the primary motive is efficiency or security or…

Given a logical address L and a page size P, the page number and offset are given by:

page number = L / P

offset = L % P

Since L is expressed in binary, if the page size is a power of 2, then no arithmetic is necessary since:

L / P = n - k high-order bits of L

L % P = k low-order bits of L

Quiz 7: Page Fault

On a demand-paged VM system, the running process references a page that is *not resident*.

What does that mean?

It means that the referenced page is not currently in physical memory.

Quiz 8: Page Fault

What must the OS do when the running process generates a page fault?

1. Block the running process.
2. Select a frame F to receive the page to be loaded.
3. If necessary, queue a request with the storage device to write the page in frame F back to disk.
4. Queue a request with the storage device to copy the relevant page from disk to memory.
5. Select another process to run.