

Operating Systems PhD Qual

Spring, 2006

Do all problems.

1. In UNIX, files are untyped. That is, the file-related calls in the UNIX API allow the application to read or write byte sequences of any length. In contrast, some older operating systems supported typed files. These operating systems were aware of certain file types—such as a B-tree or a sequence of fixed-size records—and their APIs included calls that allowed the application to read or write data units appropriate to the type of the file.

Discuss the advantages and disadvantages of the typed and untyped models for files.

2. Tanenbaum's discussion of page replacement algorithms includes the "optimal" algorithm. The optimal page replacement algorithm has knowledge of future page reference behavior and always removes from memory the page that will be used farthest in the future. The optimal algorithm cannot be implemented in a real system, but it is useful as the base case in simulations. The optimal algorithm is said to cause no more page faults than practical algorithms.

Explain the following seeming paradox: how can the optimal algorithm cause fewer page faults than practical algorithms, since it seems that its only effect (compared to practical algorithms) is to cause a page fault to happen later rather than earlier during the simulation?

3. Tanenbaum presents the following solution to the producer/consumer problem using semaphores:

```
/* ints used as semaphores */
int mutex = 1;
int empty = N;
int full = 0;

void producer(void) {
    int item;
```

```

while (1) {
    item = produce_item();
    down(&empty);
    down(&mutex);
    insert_item(item);    /* insert into buffer */
    up(&mutex);
    up(&full);
}
}

void consumer(void) {
    int item;

    while (1) {
        down(&full);
        down(&mutex);
        item = remove_item(item);    /* remove from buffer */
        up(&mutex);
        up(&empty);
        consume_item();
    }
}

```

Explain the logic of this solution in enough detail to demonstrate that you understand it. In particular, explain the following:

- a. Why each semaphore is initialized to the indicated value.
 - b. Why the pair of operations “down(&mutex)” and “up(&mutex)” are nested inside the other semaphore operations rather than outside. What would be the effect if the operations on the `mutex` semaphore were placed outside the other semaphore operations?
 - c. Why the `up` and `down` operations on the `empty` and `full` semaphores are executed by different functions.
4. Explain how a “digital signature” is computed and what it is useful for.