Computer Science 340

Operating Systems

1989 TEST, part B : SKETCH ANSWERS

Here is an outline of possible answers to questions 5 and 6 of the test. These are not THE only acceptable answers; but they're (abbreviated versions of) the answers I hoped I'd get. The answers are written (mostly) in this typeface.

Alan Creak,
June 1989.

Material written in this italic typeface is comment added after marking the test.

QUESTION 5.

off-lining :

1	Copy F on SSC from cards to tape.	
2	Read F from tape to disc on LFE.	
3	Execute P on LFE reading F from disc and writing R to disc.	
4	Copy R on LFE from disc to tape.	
5	Copy R on SSC from tape to paper.	

spooling :

1	Copy F on SSC from cards to disc on LFE.	
2	Execute P on LFE reading F from disc and writing R to disc.	
3	Copy R on LFE from disc to paper.	

off-lining :

1	Copy F on SSC from cards to tape.	(1/c + 1/t) I S
2	Read F from tape on LFE.	(1/t) I R
3	Execute P on LFE.	(1/L) (I + O) R
4	Copy R on LFE to tape.	(1/t) O R
5	Copy R on SSC from tape to paper.	(1/t + 1/p) O S

spooling :

1	Read F from cards on LFE.	(1/c) I R
2	Execute P on LFE.	(1/L) (I + O) R
3	Copy R on LFE to paper.	(1/p) O R

Total cost for off-lining =

$$(1/c + 1/t)\,I\,S + (1/t)\,I\,R + (1/L)(I + O)\,R + (1/t)\,O\,R + (1/t + 1/p)\,O\,S$$

$$= \{(1/c + 1/t)\,I + (1/t + 1/p)\,O\}\,S + \{(1/t + 1/L)(I + O)\}\,R$$

If c = p = z, say :

Total cost for off-lining =

$$\{(1/z + 1/t)(I + O)\}\,S + \{(1/t + 1/L)(I + O)\}\,R$$

$$= \{(1/z + 1/t)\,S + (1/t + 1/L)\,R\}\,(I + O)$$

Total cost for spooling =

$$(1/c)\,I\,R + (1/L)(I + O)\,R + (1/p)\,O\,R$$

If c = p = z, say :
Total cost for spooling =

$$\{(1/z + 1/L)\,R\}\,(I + O)$$

If off-lining is cheaper than spooling :

$$\{(1/z + 1/t)\,S + (1/t + 1/L)\,R\}\,(I + O) < \{(1/z + 1/L)\,R\}\,(I + O)$$
$$(1/z + 1/t)\,S + (1/t + 1/L)\,R < (1/z + 1/L)\,R$$
$$(1/z + 1/t)\,S < (1/z - 1$$

Using the values given :

$$2.02\,S < 1.98\,R;\ \text{or}\ R > 1.02\,S.$$

This question didn't turn out quite as I'd expected, because I composed it in the belief that the test would be open-book. When it became clear that Robert had worked to the other assumption, it was too late to do much about it except have a closed-book test, and at that point I should have checked my questions rather more carefully. I would not otherwise have thrown mediæval terms like "off-lining" at you without some sort of supporting definition; even "spooling"'s a shade marginal.

A lot of people didn't answer the question as set - very commonly, the second part was omitted or merged with the first or third parts. That's unwise for two reasons :

- *It messes up the marking scheme. As I regard the test as an attempt to find out how well you know operating systems rather than a test of your obedience to arbitrary instructions I go to some trouble to try to interpret what you give me, even if it isn't what I expect; but it's often difficult to identify the precise components which I hope to find, and this inevitably leads to some inconsistency.*
- *You get the wrong answer. A lot of the people who didn't "Write down expressions for the cost of each of the steps"[1] got some of them wrong. I split the question up deliberately so that you wouldn't attempt to compose complicated expressions without giving due thought to their components.*

A lot of people lost marks because they missed bits out - parts of the file transfers, or all of spooling, or part of the costs, or whatever.

First part.

A few people included time to load (and sometimes to compile and link) P. I hadn't intended that, and I don't think it's realistic : it certainly isn't part of the time needed "to process a data file … through a programme P".

I asked you to describe "the sequence of operations" involved in the operation - then later I asked you to "describe briefly" the corresponding spooling job. I'd intended both "describe"s to be interpreted in the same way, but a lot of people gave me a list of steps for the first and a chatty description for the second.

A number of people thought that SSC was controlled by LFE, with data transferred between the two on a hard-wired link of some sort. Off-lining was used before decent interrupt systems were available to make spooling possible; any sort of interprocessor communication (except along terminal-like lines at a few kByte s^{-1}) just wasn't feasible in practical systems.

There was some disagreement as to just what constitutes an "operation" : as I hadn't defined it (I couldn't, without giving you most of the answer), that's fair enough, and I accepted anything plausible.

Several people wanted to transfer the files on disc rather than tape. This did happen occasionally towards the end of off-lining, but wasn't common; it didn't happen sooner because it needed large demountable disc packs. It isn't as easy as it sounds to attach one disc drive to two computers.

Second part.

*UNITS AND DIMENSIONS. Oh, dear. The first cost on the first script I read was written as c*I*S, with dimensions of $ kB^2 s^{-2}. Isn't it obvious that it should have been (I/c)*S ? The units have to cancel out to give what you want - in this case, $. Someone else actually wrote "I kB * c kB/s"; and one or two people added together things of different dimension - like I/t + L + O/t. PLEASE THINK A LITTLE WHEN WORKING OUT ALGEBRA : and it's horrifying to contemplate what might have happened if I'd asked you to work it out on a calculator, which leaves no evidence of the intermediate steps at all.*

(After all that, I should add that some people got it all right without any trouble : well done ! But there weren't nearly enough of them.)

Again, different people had different ideas of what a "step" was. I accepted anything sensible.

Several people described off-lining as using a magnetic tape transfer for one of the input and output files, but not the other. That really isn't very sensible; you only get the full advantage by speeding up both transfers.

Quite a lot of people ignored the first assumption, and used either I or O (instead of I + O) as a measure of the size of the processing. Once again, they suffered : the whole point of the "I + O" was to make the algebra simpler. (Of course, it isn't an unreasonable assumption.) In the long run, it makes no difference to the answer, because the processing time is the same in both cases.

Do not make your own assumptions until you have to, and then state clearly what you have assumed and why. In this question, you didn't need any extra assumptions; in particular, there was no need to assume numerical values for the sizes of the files.

(In hindsight, it would have been neater to rephrase the third assumption as something like "The time taken by disc operations is negligible" - which would have given the same answer without mentioning unrealism. In fact, the question as set isn't even all that unrealistic, as programmes not uncommonly read or wrote their own tapes directly; but to make sense of reading directly from or writing directly to tape, you have to talk about buffering, which introduces complications of its own.)

Third part.

The numbers are realistic : I took them, more or less, from "Programming business computers" (D.D. McCracken, H.Weiss, and T.-H. Lee (Wiley, 1959))

If you get a result that looks silly, at least add a comment to that effect : it shows me that, even if the answer's wrong, you're bright enough not to believe it.

1 : Which, I've just noticed, is not well formed : read "Write down an expression for the cost of each of the steps". Nobody complained about the grammar.

QUESTION 6.

buffer.in and *buffer.out* are channels (from context). The programme copies from *buffer.in* to *buffer.out*, always simultaneously reading into one of *x* and *y* and writing from the other, and alternately reading into *x* and *y*.

```
WHILE TRUE
   control ? request
      IF
         ( request = get )
            SEQ
               in ? buff[ tail ]
               tail := tail + 1
               IF
                  ( tail > 9 )
                     tail := 0
         ( request = put )
            SEQ
               out ! buff[ head ]
               head := head + 1
               IF
                  ( head > 9 )
                     head := 0
```

`WHILE TRUE`	repeat forever.
`control ? request`	responds to any signal on the `control` channel. The signal is place in the variable `request.`
`IF( request = get )`	tests the value of `request`, and executes the dependent statement if it's “get”.
`SEQ`	executes its dependent statements sequentially.
`in ? buff[ tail ]`	reads from channel `in` into the tail of the buffer queue.
`tail := tail + 1`	advances the tail pointer.
`IF( tail > 9 )tail := 0`	makes it cyclic.
`( request = put )`	part of the IF : tests the value of `request`, and executes the dependent statement if it's “put”.
`SEQ`	executes its dependent statements sequentially.
`out ! buff[ head ]`	writes to channel `out` from the head of the buffer queue.
`head := head + 1`	advances the tail pointer.
`IF( head > 9 )head := 0`	makes it cyclic.

This time I did allow for the closed-bookness of the test by providing the “Extra information for question 6” sheet - which was, of course, the occam example I'd given out earlier. Despite that precaution, it was far from clear that everyone had looked at the sheet; indeed, one person's answer ended in a note something like “I've only just noticed the Extra information”.
This question was not well done. It seemed to me not unreasonable to expect that experienced computists should be able to understand a programme written in a reasonably straightforward, even if unfamiliar, procedural programming language, and to make some sort of shot at knocking together a programme of their own - particularly when they had a pattern to copy !
First part.

> *I asked you to “Explain the meaning” of the code : that doesn't mean I want an interpretation of what each line does. Hardly anybody wrote (I suspect hardly anybody noticed) that the code* ***copies*** *its input to its output, though surely that's the obvious word to use. The usual operating system instruction is “copy” - even Unix uses “cp”, which for Unix is astonishingly comprehensible.*

Second part.

> *The answer is a fairly straightforward adaptation of the “button ? request” bit on the “Extra information ..” sheet.*