2010.8.10

オペレーティングシステム試験問題

(配点：(1) 10 点, (2) 20 点, (3-1) 6 点, (3-2) 7 点, (3-3) 7 点
(4-1) 5 点, (4-2) 15 点, (4-3) 6 点, (5-1) 6 点, (5-2) 6 点, (5-3) 6 点, (5-4) 6 点)

【注意】問題(1)～(3)と(4)～(5)はそれぞれ別の答案用紙に解答すること。

(1) 読取り速度 10 枚／秒のカードリーダと、プリント速度 20 ページ／分のプリンタが各 1 台ある単一 CPU の計算機システムを考える。このシステムで、以下の 3 個のジョブを実行するとき、最後に終了するジョブの終了時刻を最短にするスケジューリングを図で示し、各ジョブの終了時刻を答えよ。ただし、各ジョブは時刻 0 で到着し、各ジョブは入力・計算・出力の処理をこの順番で実行し、それらの処理中では横取りは生じないものとする。

ジョブ	入力カード枚数	計算時間	出力ページ数
ジョブ A	1000 枚	20 秒	5 ページ
ジョブ B	50 枚	180 秒	10 ページ
ジョブ C	50 枚	5 秒	60 ページ

(2)単一 CPU のマルチプログラミング環境におけるプロセスのスケジューリングについて考える。マルチプログラミング環境では、通常、複数の実行可能なプロセスがレディ(ready)キューに登録され、CPU が空いたときキューから先頭のプロセス 1 個が取り出されて実行される。プロセス P1～P4 が到着時刻と処理時間が以下の表のように与えられているとき、

- 到着順(First Come First Served; FCFS)
- 処理時間順(Shortest Processing Time First; SPT)
- 残余処理時間順(Shortest Remaining Time First; SRT)
- ラウンドロビン(Round Robin; RR)

のそれぞれのスケジューリング方式について、各プロセスのターンアラウンドタイム(プロセスが到着してから終了するまでの時間) とそれらの平均値を求めよ。ただし、時刻 0 で CPU は空いているものとする。また、ラウンドロビンでは、タイムスライスを 10 として、新規到着プロセスはレディキューの末尾につながれ、実行中のプロセスが終了した場合は次のタイムスライスまで待って別のプロセスをディスパッチするものとする。

プロセス	到着時刻	処理時間
P1	0	50
P2	8	30
P3	18	3
P4	28	10

(3)単一 CPU のマルチプログラミング環境で、次の 2 つのプロセス P1 と P2 が並行に動作している状況を考える。

P1

```
wait(s1);          ①
buffer = a1[0];    ②
signal(s2);        ③
wait(s1);          ④
buffer = a1[1];    ⑤
signal(s2);        ⑥
```

P2

```
wait(s2);          ⑦
a2[0] = buffer;    ⑧
signal(s1);        ⑨
wait(s2);          ⑩
a2[1] = buffer;    ⑪
signal(s1);        ⑫
```

上のプログラムで、buffer は整数型変数、a1, a2 は大きさ 2 の整数型の配列(a1[0]と a1[1]には既に初期値が設定されている）で、s1, s2 はセマフォ（初期値は、s1=1, s2=0）であり、buffer, s1, s2 は P1 と P2 で共有されているものとする。また、wait と signal はセマフォについての操作であり、それぞれ次のような動作をする。

wait(s)：セマフォ s の値が正なら値を 1 減らして次の文へ進む。s の値が 0 ならブロックする（プロセスの実行が中断する)。

signal(s)：以前にセマフォ s で wait を実行してブロックしているプロセスがあるとき、そのプロセスのうち 1 つの実行を再開させる（レディキューにつなぐ)。ブロックしているプロセスがないときは、s の値を 1 増やす。

このとき以下の小問に答えよ。ただし、以下では初期状態として、プロセス P1, P2 が共にレディキューにつながれており、最初に P1 がディスパッチされるものとする。また、プロセスのスケジューリングではプリエンプション（横取り）はなく、wait 操作によるブロック以外はプロセスの実行は中断しないものとする。

(3-1) プロセス P1, P2 の文①～⑫はどのような順番で実行されるか答えよ。

(3-2) P1, P2 のセマフォを s1 だけにする（すなわち、③, ⑥, ⑦, ⑩の行の s2 を s1 に置き換える）と、①～⑫の実行の順番はどのようになるか答えよ。

(3-3) P2 のプログラムで s1 と s2 を入れ替えて、⑦、⑩の行を wait(s1)に、⑨、⑫の行を signal(s2)に置き換えると、どのような不都合が生じる可能性があるか 50 字以内で述べよ。

(4) 仮想記憶を実現するためのページング方式においては、ページ置き換えアルゴリズムが重要になる。そのうち FIFO アルゴリズムを対象とした場合の動作例を、以下に示す。ただし、参照ストリング を “0 1 2 3 4 5 1 0 3 2 0 4 5 1”、ページ枠数を 4 とする。なお、ページ枠の内容として FIFO キューの変化の様子を示していることに注意せよ。

参照ストリング	0	1	2	3	4	5	1	0	3	2	0	4	5	1
ページ枠の内容（FIFO キュー）	0	0	0	0	1	2	3	4	5	1	1	0	3	2
	−	1	1	1	2	3	4	5	1	0	0	3	2	4
	−	−	2	2	3	4	5	1	0	3	3	2	4	5
	−	−	−	3	4	5	1	0	3	2	2	4	5	1

(4-1) ページ置き換えアルゴリズムとして、FIFO アルゴリズム以外に、以下のような動作を行う OPT アルゴリズムがある。どのような動作を行っているか、アルゴリズムを簡単に説明せよ。また、他のページ置き換えアルゴリズムと比較した時の特徴を述べよ。なお、上の例と異なり、ページ枠の内容そのものを示していることに注意せよ。

参照ストリング	0	1	2	3	4	5	1	0	3	2	0	4	5	1
ページ枠の内容	0	0	0	0	0	0	0	0	0	0	0	4	4	4
	−	1	1	1	1	1	1	1	1	1	1	1	1	1
	−	−	2	2	4	5	5	5	5	5	5	5	5	5
	−	−	−	3	3	3	3	3	3	2	2	2	2	2

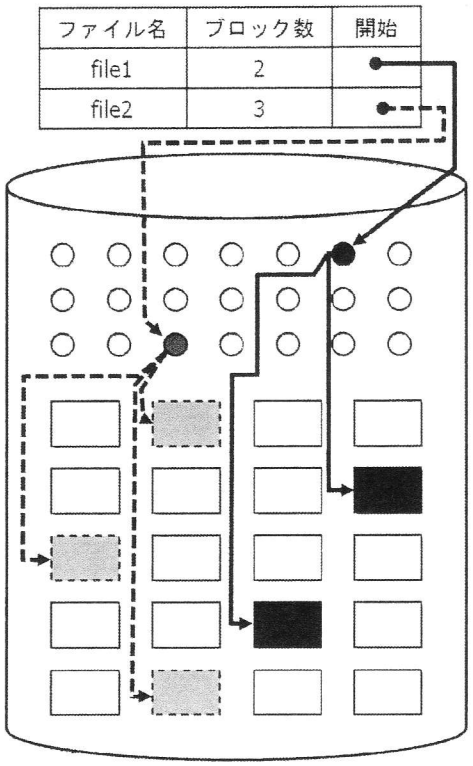
(4-2) LRU アルゴリズムについて、その動作を簡単に説明せよ。また、小問 (4-1) に示したのと同様に、参照ストリングとして “0 1 2 3 4 5 1 0 3 2 0 4 5 1”、ページ枠数を 4 とした場合の、ページ枠の内容の変化の様子を示せ。

(4-3) 一般には、ページ置き換えアルゴリズムとして LRU アルゴリズムがよいとされているが、それを忠実にハードウェア実装するのは困難で、LRU アルゴリズムを簡略化したものが実装されている。簡略化されたアルゴリズムはどのようなものか例を挙げて説明せよ。

(5) ファイルシステムの実装方法として、索引ブロックを用いた割付手法（索引自体をディスクの索引用ブロックとして、ファイル本体のデータブロックとは別におく）がある（次ページ図参照）。

(5-1) データブロックのアドレスを 32 ビット（4 バイト）であらわすものとし、データブロックのサイズを 4 K バイトとする。索引ブロックを 128 バイトとした場合、索引ブロック 1 つで表すことのできる最大ファイルサイズはいくらか。

(5-2) 索引ブロックのサイズを変更しないまま、扱えるファイルサイズを大きくするためには、データブロックのサイズを大きくする方法が考えられる。ただし、その結果、引き起こされる問題もある。どのような問題が発生するか説明せよ。



(5-3) データブロックのサイズを変更せずに、大きいファイルサイズを扱うための拡張方法として、UNIX で実現されている方法を簡単に説明せよ。

(5-4) 索引ブロックを用いた割付手法は、連結リストの索引を用いた割付手法（ブロックに対するポインタのリンク情報を索引としてメモリ上におく手法）に比較すると、メモリ消費量を軽減できることが利点として挙げられる。一方、その欠点として、１つのデータブロックに収まるような小さいサイズのファイルの読み書きでも、索引ブロックおよびデータブロックに対する２回のディスクアクセスが必要になり、アクセス時間が増大することが予測される。しかし、実際には、このようなアクセス時間の増大はさほど発生しない。その理由を述べよ。