オペレーティングシステム I 令和 3 年度 後期末試験

(2022.02.07 重村 哲至)　　IE4　　＿＿＿番　氏名＿＿＿＿＿＿ 模範解答 ＿＿＿＿＿＿

注意：以下で「プロセス」と「スレッド」は同じ意味で用いられていることがある．

1　語句に関する問題

次の文章の空欄に最適な言葉を語群から記号で答えなさい．ただし，(10) には数値を答えること．
(1 点 ×30 問＝ 30 点)

プログラムの中で資源の利用に関して競合が発生する可能性のある部分は **(1)** と呼ばれる．複数のスレッドが同時に **(1)** に入らないように **(2)** を行う必要がある．**(3)** セクションで **(1)** に入る権利を得る処理を，**(4)** セクションで権利を返却する処理をする．

(5) プロセッサシステムでは割込みを禁止にすることで **(2)** を行うことができるが，**(6)** プロセッサシステムでは **(7)** 命令などを用いる必要がある．**(5)** プロセッサシステムでは **(1)** で **(8)** することを防げば十分である．

セマホはスレッドの待ち行列と **(9)** を持つデータ構造である．**(2)** を行うためには初期値が **(10)** のセマホを用い **(3)** セクションで **(11)** を，**(4)** セクションで **(12)** を行う．**(11)** はスレッド状態を **(13)** 遷移させることがある．

プロセス間で行うメッセージ通信には，通信相手を指定するために **(14)** を用いる間接指定方式と用いない直接指定方式がある．受信するメッセージを選択するために **(15)** を使用できる方式と使用できない方式もある．また，受信システムコールを発行した時にメッセージが届いていない場合，プロセスが待ち状態になる **(16)** 方式とシステムコールがエラーで終了する **(17)** 方式がある．

モニタはリソース管理機能を持ったプログラミング言語の **(18)** である．モニタには手続きが排他的に実行されるようにする **(19)** が備えられている．名前を付けて宣言できる条件変数には **(20)**，**(21)** の 2 つの操作ができる．**(21)** によって実行可能になったスレッドは **(22)** に実行される．

プロセスが互いに資源の解放を待ち，永遠に処理を進めることができない状態を **(23)** と言う．**(23)** の原因は，プロセスが資源を確保したまま他の資源を待つ状態になる **(24)**，複数のプロセスが互いを待ち合う状態になる **(25)** である．**(24)** を避けるためには **(26)** を用いて一度に資源を確保する方法，**(25)** を避けるためには資源を確保する **(27)** に制約を設ける方法が考えられる．しかし，一度に資源を確保する方法は資源の **(28)** を悪くする．また，**(27)** に制約を設ける方法は一部のプロセスを **(29)** に扱うかもしれない，**(30)** の問題は **(27)** を決められない例である．

語群：

(あ) P_and，(い) P 操作，(う) signal，(え) V 操作，
(お) wait，(か) TS，(き) エグジット，
(く) エントリー，(け) ガード，(こ) カウンタ，
(さ) クリティカルセクション，(し) シングル，
(す) タグ，(せ) デッドロック，(そ) プリエンプション，
(た) ブロック，(ち) マルチ，(つ) リンク，
(て) ただち，(と) 確保待ち，(な) 食事する哲学者，
(に) 循環待ち，(ぬ) 順序，(ね) 相互排除，
(の) 抽象データ型，(は) 同期，(ひ) 非同期，
(ふ) 不公平，(へ) 利用効率

(1)	(さ)	(2)	(ね)	(3)	(く)	(4)	(き)
(5)	(し)	(6)	(ち)	(7)	(か)	(8)	(そ)
(9)	(こ)	(10)	1	(11)	(い)	(12)	(え)
(13)	(た)	(14)	(つ)	(15)	(す)	(16)	(は)
(17)	(ひ)	(18)	(の)	(19)	(け)	(20)	(お)
(21)	(う)	(22)	(て)	(23)	(せ)	(24)	(と)
(25)	(に)	(26)	(あ)	(27)	(ぬ)	(28)	(へ)
(29)	(ふ)	(30)	(な)				

オペレーティングシステムⅠ 令和３年度 後期末試験

(2022.02.07 重村 哲至)　　IE4　　＿＿＿番　氏名＿＿＿＿模範解答＿＿＿＿

2　相互排除

次の TaC 風のアセンブリ言語プログラムは，スピンロックによる相互排除機構です．このプログラムに関する以下の問いに答えなさい．なお，プログラムを書く際は TeC のニーモニックを使ってオペランドまで正確に書くこと．(5 点 ×4 問=20 点)

```
; エントリーセクション
L1    DI                ; 割込み禁止
      ##(a)##
      EI                ; 割込み許可
      JMP   L1

L2
; クリティカルセクション
      ...

; エグジットセクション
      LD    G0, #0
      ##(b)##
      EI
      ...

FLG   DC    0      ; 初期値 0 のフラグ

;TS命令
; 書式： TS G0, FLG
; 意味： メモリ(FLG)の値をG0にロードしロードし
;        た値によりZフラグを変化させる．次に
;        メモリ(FLG)に1を書き込むその間，他の
;        プロセッサはモリにアクセスできない．
```

1. マルチプロセッサシステムでも使用できるエントリーシーケンスを完成するために，##(a)##に補うべき数行のプログラムを以下に書きなさい．なお，TS 命令の仕様は上のリスト中に記載しています．

```
TS    G0, FLG
JZ    L2
```

2. プログラム中##(b)##に補うべき命令 (1 命令) を以下に書きなさい．

```
ST    G0, FLG
```

3. クリティカルセクションは割込み禁止で実行されます．割込み禁止にしなければならない理由を３０文字程度で説明しなさい．

クリティカルセクションでプリエンプションすることを防ぐため．

4. エントリーセクションでスピンロックしている間，一時的に割込みを許可する理由を５０文字程度で説明しなさい．

割込み禁止が長く継続すると，割込みを取りこぼす等の弊害が生じるので，割込みを受け付ける機会を作っている．

オペレーティングシステム I 令和 3 年度 後期末試験

(2022.02.07 重村 哲至)　　IE4　　＿＿＿番　氏名＿＿＿＿＿模範解答＿＿＿＿＿

3　実行順序

以下の C 言語風のプログラムにおいて二つの関数 procA() と procB() は，二つのスレッドによって並行実行されます．また，printf() 関数は複数スレッドの環境でも，正常に動作するものとします．なお，出力を答えるとき，何も出力がない場合は「出力なし」と書くこと．

1. 次のプログラムについて答えなさい．
(5 点 ×3 問=15 点)

```
Semaphore S1 = 1;  // 初期値1のセマフォ
Semaphore S2 = 0;  // 初期値0のセマフォ
void procA() {
  P( &S1 );
  printf("A-1\n");
  V( &S2 );
  P( &S1 );
  printf("A-2\n");
  V( &S2 );
}
void procB() {
  P( &S2 );
  printf("B-1\n");
  V( &S1 );
  P( &S2 );
  printf("B-2\n");
}
```

(a) 出力を書きなさい．

A-1
B-1
A-2
B-2

(b) 終了時のセマフォ S1 の値を答えなさい．

S1 = 0

(c) 終了時のセマフォ S2 の値を答えなさい．

S2 = 0

2. 次のプログラムについて答えなさい．
(5 点 ×2 問=10 点)

```
Semaphore S1 = 1;  // 初期値1のセマフォ
Semaphore S2 = 1;  // 初期値1のセマフォ
void procA() {
  P( &S1 );
  printf("A-1\n");
  P( &S2 );
  printf("A-2\n");
  V( &S2 );
  V( &S1 );
}
void procB() {
  P( &S2 );
  printf("B-1\n");
  P( &S1 );
  printf("B-2\n");
  V( &S1 );
  V( &S2 );
}
```

(a) 出力が次の 2 行で始まった場合の残りの出力を答えなさい．

```
A-1
A-2
```

B-1
B-2

(b) 出力が次の 2 行で始まった場合の残りの出力を答えなさい．

```
A-1
B-1
```

出力なし

オペレーティングシステムⅠ 令和３年度 後期末試験

(2022.02.07 重村 哲至)　　IE4　　＿＿＿番　氏名＿＿＿＿＿＿模範解答＿＿＿＿＿＿

4　モニタによるセマフォの実装

次は授業で紹介した仮想言語のモニタを用いてセマフォを実装した例です．条件変数には c.wait(), c.signal() と c.queue() の操作ができるものします．(c.queue() は条件変数の待ち行列の長さを返します.)

```
monitor Semaphore {
  int cnt;                  // カウンタ
  Condition c;              // 条件変数
  Semaphore(int n) {        // 初期化プログラム
    cnt = n;                // セマフォの初期値
  }
  public void P() {         // P操作
    if (##(a)##) {
      cnt--;
    } else {
      ##(b)##;
    }
  }
  public void V() {         // V操作
    if (c.queue()==0) {     // 待プロセスなし
      ##(c)##;
    } else {
      ##(d)##;
    }
  }
}
```

1. プログラム中「##(?)##」に適切な記述を答えなさい．(4 点 ×4 問=16 点)

##(a)##	cnt>0
##(b)##	c.wait()
##(c)##	cnt++
##(d)##	c.signal()

2. 次はモニタで実装したセマフォの利用例です．例は生産者と消費者問題の解です．プログラム中「##(?)##」に適切な記述を答えなさい．
(3 点 ×3 問=9 点)

```
// リングバッファ
Data       buffer[N];
// 空きスロット数を管理するセマホ
Semaphore emptySem = new Semaphore(N);
// 使用中スロット数を管理するセマホ
Semaphore fullSem  = new Semaphore(0);
// 生産者スレッド
void producerThread() {
  int in = 0;
  for ( ; ; ) {
    Data d = produce();
    emptySem.P();          // 空きを確認
    buffer[ in ] = d;
    in = (in + 1) % N;
    ##(1)##;
  }
}
// 消費者スレッド
void consumerThread() {
  int out = 0;
  for ( ; ; ) {
    ##(2)##;
    Data d = buffer[ out ];
    out = (out + 1) % N;
    ##(3)##;
    consume( d );
  }
}
```

##(1)##	fullSem.V()
##(2)##	fullSem.P()
##(3)##	emptySem.V()