Ｃｈａｐｔｅｒ９　基本ソフトウェア

９－１　ＯＳの仕事〔解答・解説〕

問１　イ

問２　ア

問３　ア
〔解説〕イ　再配布は自由にできる
ウ　ソースコードの取得，変更，再配布はできるが，著作権は放棄されていない
エ　オープンソースとはソースコードを公開するという意味であり，有償で配付しても構わない

問４　ウ
〔解説〕コピーレフトとは，オープンソースライセンスにおいて，著作権を保持したまま，プログラムの複製や改変，再配布を制限せず，そのプログラムから派生した二次著作物(派生物)には，オリジナルと同じ配布条件を適用する，という考え方である。

問５　ア
〔解説〕イ　Perl(パール)は、Larry Wall 氏によって開発されたテキスト処理用のプログラム言語。以前は Web ページの CGI の記述によく用いられていた。
ウ　PHP(ピーエイチピー)は、Web アプリケーションのサーバーサイド・スクリプトに専門化し、データベースと連携した動的な Web ページを比較的容易に生成することを主目的としたプログラム言語。
エ　Ruby(ルビー)は、まつもとゆきひろ氏が開発したオブジェクト指向型のスクリプト言語。

問６　ア
〔解説〕イ　静的リンクのように本体プログラムに組み込んで使用する場合には本体プログラムもＧＰＬ扱いになる。したがってソースコードを公開する必要がある
ウ　ＧＰＬライセンスは、サービスの提供に料金を課してもよいし、また無料で行ってもよい
エ　複製の自由な再頒布を禁止してはいけない

問７　ア
〔解説〕Ｅｃｌｉｐｓｅ（エクリプス）は，ＩＢＭによって開発されたソフトウェアの統合開発環境ＯＳＳ（オープンソースソフトウェア）であり，Ｊａｖａをはじめとするいくつかの言語に対応している。

問８　ウ
〔解説〕標準入力はキーボード，標準出力はディスプレイとなっているが，リダイレクト機能により，“＜”記号で標準入力をファイルからの読込みに，“＞”記号で標準出力をファイルへの書込みに，“＞＞”記号で標準出力をファイルへの追加書込みにすることができる。

問 9　イ
〔解説〕ア　Ｔｏｍｃａｔ(トムキャット)は、Ｊａｖａサーブレット・ＪＳＰで処理を行うオープンソースのＷｅｂアプリケーションサーバ
ウ　ＧＣＣ(GNU C Compiler)は、ＧＮＵが開発・配布している様々なプログラム言語のコンパイラでＯＳＳの条件に従って自由に使用できる
エ　Ｌｉｎｕｘは、世界中のプログラマや企業により改良され、発展し、世界的に利用されているオープンソースのＯＳ

問 10　ア
〔解説〕パイプ(Pipe)は、ＵＮＩＸ系ＯＳにおいて、あるプログラムの標準出力を直接別のプログラムの標準入力に接続することで、複数のプログラムを連鎖的に実行する仕組み。
イ　ＵＮＩＸ系ＯＳにおいて、時間の掛かる処理を裏側で行わせておくことができる機能
ウ　ブレース(brace)とは中括弧"{}"の意味で、"{"と"}"で囲まれた複数の変数を様々に展開する機能
エ　標準入出力を切り替えることができる機能です。ＣＵＩ環境では標準入力＝キーボード、標準出力＝モニターですが、入力元や出力先をファイルなどに変更するときに使用

問 11　ア
〔解説〕イ　オープンソースの分散メッセージングシステム(Ａｐａｃｈｅ　Ｋａｆｋａ)
ウ　オープンソースの分散処理システム(Ａｐａｃｈｅ　Ｓｐａｒｋ)
エ　オープンソースのリアルタイム分散処理システム(Ａｐａｃｈｅ　Ｓｔｏｒｍ)

9-2　ジョブ管理〔解答・解説〕

問 1　エ
〔解説〕スプーリング：システム全体のスループットを高めるため、主記憶装置と低速の出力装置とのデータ転送を、高速の補助記憶装置を介して行う方式。
スループット：単位時間当たりのジョブの処理件数のこと。スプーリングはスループットの向上に役立つ。

問 2　イ

問 3　ア
〔解説〕イ　タスクの状態遷移に関する記述。
ウ　ジョブは OS から見た処理単位なので、バッチ処理でもオンライン処理でも同等な関係である。
エ　リーダ，イニシエータ，ターミネータ，ライタはジョブスケジューラの機能。

問 4　ウ
〔解説〕ジョブＡが終了するまでに，Ａ，Ｂ，Ｃを順に処理していくため１５分かかる
ジョブＢが終了するまでに，Ｂ，Ｃを順に処理していくためあと１０分かかる
ジョブＣが終了するまでに，あと５分かかる
したがって，ジョブＡ，Ｂ，Ｃが終了するまでにそれぞれ１５分，２５分，３０分かかる

問 5　イ
〔解説〕ア　入出力割込みの説明
ウ　デバイスコントローラの説明
エ　デバイスファイルの説明

問 6　ウ
〔解説〕ジョブ（実行時間２０分）を多重度3で実行するため，ジョブの平均処理時間は
　２０／3≒6.6（分)。
到着間隔（5分）よりも大きいため，実行待ちジョブが次第に増える。
また，印刷（１５分）を2台のプリンタで行うため，平均印刷時間は
　１５／２＝7.5（分)。
到着間隔（5分）よりも大きいため，印刷待ちジョブが次第に増える。

問 7　ウ
〔解説〕一時ファイルはジョブの開始時に作成され、直後のジョブが終了した時点で削除される。問題文の条件に従ってジョブの実行状況を追跡すると次のようになる。
１：ジョブAが生起され実行開始される。
　　→50Mバイトの一時ファイルを作成
２：ジョブAが終了する。一時ファイルは直後のジョブB、Cで参照するので削除しない。
３：ジョブB、ジョブCが生起される。多重度は2なのでどちらも実行開始される。
　　→50×2で100Mバイトの一時ファイルを作成
４：ジョブB、ジョブCが終了する。ジョブAの一時ファイルが削除される。
５：ジョブD、ジョブEが生起される。多重度は2なのでどちらも実行開始される。
　　→50×2で100Mバイトの一時ファイルを作成
６：ジョブD、ジョブEが終了する。ジョブB、ジョブCの一時ファイルが削除される。
７：ジョブFが生起され実行開始される。
　　→50Mバイトの一時ファイルを作成
８：ジョブFが終了する。ジョブD、ジョブEの一時ファイルが削除される。

一時ファイルの容量が最も多くなるのは、4つの一時ファイルが同時に存在するジョブD・E実行中で、その容量は200Mバイトである。したがって、一時ファイルを作成する磁気ディスクには少なくとも200Mバイトの容量が必要となる。

9-3　ﾀｽｸ管理〔解答・解説〕

問 1　ア
〔解説〕イはジョブ管理，ウ，エはデータ管理の機能

問 2　イ

問 3　ウ

問 4　ア
〔解説〕イ　遷移は起こらない
　　ウ　待ち状態から実行可能状態への遷移
　　エ　実行状態から待ち状態への遷移

問 5　エ
〔解説〕ア　データ管理の説明
　　イ　記憶管理の説明
　　ウ　入出力管理の説明

問 6　ウ
〔解説〕ノンプリエンプティブ方式とは，実行可能状態になったタスクから実行を行う方式であり，ＯＳによるＣＰＵ使用権割当てなどの管理は行われない。

問 7　ウ
〔解説〕ア　ページの入れ替えが頻繁になりすぎて，データ処理が進まなくなること
イ　プログラムの実行時に必要な資源を割り当てること
エ　同じ優先度のタスクが平等に実行されるよう，優先度を順に回転させる方式

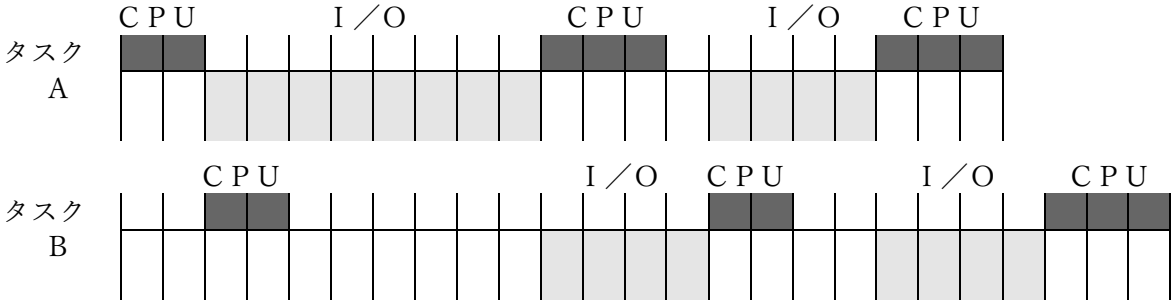
問 8　ウ
〔解説〕マルチプログラミングでは，主記憶上に複数のプログラム（タスク）を置き、それぞれの入出力動作時のＣＰＵの空き時間を利用して他のプログラムを実行させるので，ＣＰＵのスループットを向上させることができる。

問 9　ウ
〔解説〕タイムチャートを作成すると次のようになる。

タスクA	ＣＰＵ ２０	Ｉ／Ｏ ３０	ＣＰＵ ２０	待ち 10	Ｉ／Ｏ ４０	CPU 10	
タスクB	待ち ２０	CPU 10	待ち ２０	Ｉ／Ｏ ３０	ＣＰＵ ２０	待ち ２０	Ｉ／Ｏ ２０	ＣＰＵ ２０

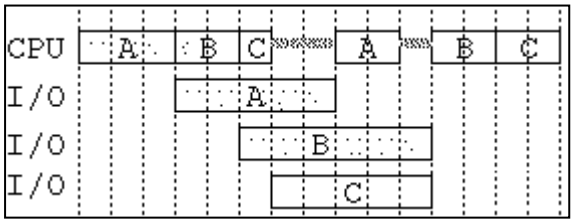
よって，二つのタスクの処理が完了するまでの時間は１６０ミリ秒

問 10　ウ
〔解説〕タイムチャートは次のように変化する。



両方のタスクが終了するまでの時間は２５，このうちＣＰＵを使用しているのは１５であるから，
ＣＰＵ使用率＝１５÷２５＝0.6→６０％

問 11　イ
〔解説〕優先度が高いタスクの順にをＡ、Ｂ、Ｃとすると図のように実行され、 ＣＰＵ の遊休時間は３ミリ秒となる。



問 12　イ
〔解説〕ア，ウ，エは内部割込み。

問 13　ウ
〔解説〕ア　内部割込みの一つページフォールトの説明
イ　内部割込みのＳＶＣ割込み(スーパーバイザコール)の説明
ウ　外部割込みに分類される機械チェックの割込みの説明(正解)
エ　内部割込みの一つであるプログラム割込みの説明

問 14　イ
〔解説〕ア　タスクスケジューリングの説明。
ウ　タスクの内部状態、置かれた状況、タスク ID 及び優先度などを保持する TCB(Task Control Block，タスク制御ブロック)の説明。
エ　マルチタスクまたはコンカレント処理の説明。

問 15　ウ
〔解説〕時間の経過と CPU の使用状況を表にしながら考えるとわかりやすい。

まず優先度が"高"であるタスクは、他のタスクの実行によって待ち状態になることはないため、まずはこのタスクの CPU 使用状況を表に描き入れる。

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
優先度"高"				I/O	I/O	I/O				I/O	I/O	I/O							

図　優先度"高"タスクのCPU使用状況

他の優先度"低"のタスクは優先度"高"のタスクが CPU を使用していない間だけ CPU を使用できることを踏まえて、それぞれのタスクと組み合わせた場合の CPU 使用状況を描き入れる。

※空白は待ち時間を示す

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
優先度"高"				I/O	I/O	I/O				I/O	I/O	I/O							
タスク(ア)						I/O	I/O	I/O	I/O	I/O			I/O	I/O					
優先度"高"				I/O	I/O	I/O				I/O	I/O	I/O							
タスク(イ)							I/O	I/O				I/O	I/O	I/O					
優先度"高"				I/O	I/O	I/O				I/O	I/O	I/O							
タスク(ウ)							I/O	I/O					I/O						
優先度"高"				I/O	I/O	I/O				I/O	I/O	I/O							
タスク(エ)							I/O	I/O	I/O	I/O			I/O	I/O	I/O	I/O	I/O		

図　各タスクと遊休時間の有無

下線が書かれている時間が CPU の遊休時間になる。
表にしてみると一目瞭然で、優先度"高"のタスクと「ウ」のタスクを組み合わせた場合に CPU の遊休時間が最も少ない 0 ミリ秒になることがわかる。

問 16　ウ
〔解説〕割込み処理では，特権モードに移行後，元の状態に戻すためのレジスタ類の退避を行い，実際の割込み処理を行う。

問17　ウ
〔解説〕プロセスＡを実行中に割込みが発生し，プロセスＢを実行するまでのＯＳの処理の組合せを答える。
1.プロセスＡの実行状態を退避する。
2.割込み処理を行うプロセスＢを選択する。
3.プロセスＢで実行状態を回復する。

問18　ア
〔解説〕イ　処理時間順方式の説明
ウ　イベントドリブンプリエンプション方式(リアルタイム方式)の説明
エ　静的優先順位方式の説明

問19　ア
〔解説〕イ　割込みの説明
ウ　タスク間同期制御の説明
エ　排他制御の説明

問20　ウ
〔解説〕以下のようにスケジューリングされる。
1．タスクA,B,Cが実行可能状態になり,優先度の最も高いタスクAが時間２までＣＰＵを使用する
2．タスクAが時間２で入出力処理に移行し,空いたＣＰＵをタスクB,Cのうち優先度の高いタスクBがＣＰＵの使用を開始する
3．タスクAは時間４で入出力処理を完了し,時間４まで経過したタスクBからＣＰＵを取り,時間６までＣＰＵを使用する。残り時間１のタスクBはその間は処理を中断する
4．タスクAは時間６で処理を完了し,ＣＰＵが空くので,タスクB,Cのうち優先度の高いタスクBが中断していた処理のためＣＰＵを時間７まで使用する
5．タスクBは時間7で入出力処理に移行し,ＣＰＵがあくのでタスクCがＣＰＵの使用を開始する

問21　ウ
〔解説〕ア　Aの実行が継続される。
イ　Aの実行が継続される。
エ　B は実行可能状態に移される。

9－4　実記憶管理〔解答・解説〕

問 1　ウ
〔解説〕オーバレイは、主記憶に格納できない大きいプログラムをいくつかのブロック(セグメント)に分割し、その時の処理に必要なブロックだけを主記憶にロードして実行する方式。
オーバレイの名の通り、実行に必要となったモジュールを不必要となったモジュールが存在したのと同じ領域に上書きするため、同じ領域にロードされるモジュール同士の参照はできないことになる。このことに注目すると参照が許されるのはロードされる領域が重なっていないモジュール間だけということになる。

したがって選択肢の参照が許されるモジュール同士は、領域が重なっていない「F→A」のみ。

問 2　ア

問 3　ア
〔解説〕A，B，Cをロードし終わった時点での空き領域は５０ｋバイト
Bを解放した領域にDがロードされ，空き領域は２０ｋバイト増加
Aを解放した領域にEがロードされ，空き領域は１１０ｋバイト増加
よって，空き領域は３か所となる。

問 4　イ
〔解説〕アはページング，ウはメモリコンパクション，エはオーバレイに関する説明。

問 5　イ
〔解説〕記憶媒体の有効利用やバックアップ，配布などの効率化を目的として，複数のファイルを一つにまとめる処理をアーカイブといい，アーカイブを実行するソフトウェアを，アーカイバという。

9－5　再配置可能プログラムとプログラムの4つの性質〔解答・解説〕

問 1　ウ

問 2　イ

問 3　イ
〔解説〕ア　オーバーライドは、スーパクラスで定義されたメソッドをサブクラスで再定義すること
ウ　カプセル化は、オブジェクト内の詳細な仕様や構造を外部から隠蔽すること
エ　汎化は、複数のクラスの共通する性質をまとめて，抽象化したクラスを作ること

問 4　ウ

問 5　エ
〔解説〕アは再配置可能プログラム，イは再帰的プログラム，ウはオーバレイの説明である。

問 6　ウ
〔解説〕設問の再帰関数 f(n) は以下のような処理を行う。
引数ｎが１以下のとき
１を返す
それ以外の場合
ｎ＋ｆ(ｎ－１)を返す
ｆ(ｎ)の部分を展開しながら地道に計算していくと次のようになる。

ｆ(５)
＝５＋ｆ(４)　//ｆ(５)=５＋ｆ(４)
＝５＋４＋ｆ(３)　//f(4)=4+f(3)
＝５＋４＋３＋ｆ(２)　//f(3)=3+f(2)
＝５＋４＋３＋２＋ｆ(１)　//f(2)=2+f(1)
＝５＋４＋３＋２＋１　//f(1)=1
＝１５

したがって、ｆ(５)の値は１５です。

問 7　イ
〔解説〕ア　(誤)ＦＩＦＯ→(正)ＬＩＦＯ
ウ　(誤)逐次→(正)同時
エ　(誤)再帰的→(正)再入可能

問 8　ウ
〔解説〕　Ack(1, 3)
Ack(1, 3)は、m＞0 かつ n＞0
＝Ack(0, Ack(1, 2))
Ack(1, 2)は、m＞0 かつ n＞0
＝Ack(0, Ack(0, Ack(1, 1)))
Ack(1, 1)は、m＞0 かつ n＞0
＝Ack(0, Ack(0, Ack(0, Ack(1, 0))))
Ack(1, 0)は、m＞0 かつ n＝0
＝Ack(0, Ack(0, Ack(0, Ack(0, 1))))
Ack(0, 1)は、m＝0
＝Ack(0, Ack(0, Ack(0, 2)))
Ack(0, 2)は、m＝0
＝Ack(0, Ack(0, 3))
Ack(0, 3)は、m＝0
＝Ack(0, 4)
Ack(0, 4)は、m＝0
＝5

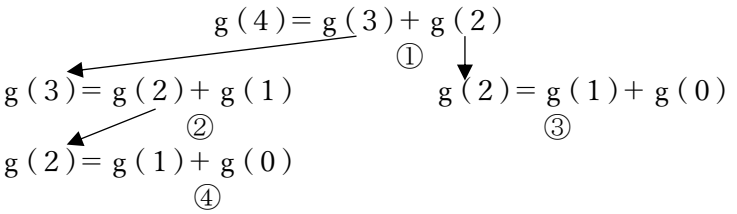
問 9　イ
〔解説〕n＝4 の場合，以下のように 4 回乗算を行う。
F(4) ＝4 × F(3)
＝4 × 3 × F(2)
＝4 × 3 × 2 × F(1)
＝4 × 3 × 2 × 1 × F(0)
＝4 × 3 × 2 × 1 × 1

問10　ウ
〔解説〕　fact(4)
＝4×fact(3)　　//n＞0
＝4×3×fact(2)　　//n＞0
＝4×3×2×fact(1)　　//n＞0
＝4×3×2×1×fact(0)　　//n＞0
＝4×3×2×1×1　　//n＝0
＝24
ア　0 になる
イ　エ　無限ループになる

問 11　イ
〔解説〕F(5) ＝5 × G(4)
＝5 ×(4 ＋ F(3))
＝5 ×(4 ＋(3 × G(2)))
＝5 ×(4 ＋(3 ×(2 ＋ F(1))))
＝5 ×(4 ＋(3 ×(2 ＋ 1)))
＝6 5

問 12　イ

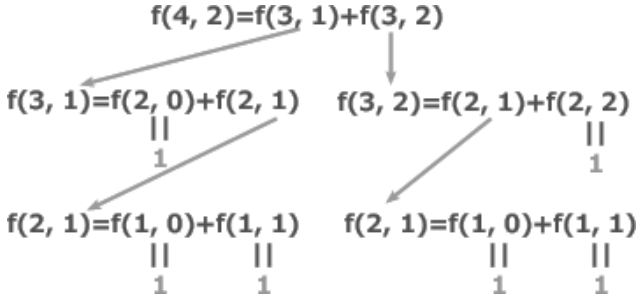
〔解説〕再帰関数を 1 つずつ展開していったものを下図に示す。g(1)と g(0)は整数 1 を返すので、これらの再帰部分は省略する。



必要となる加算の回数は 4 回。

問 13　エ

〔解説〕再帰関数を 1 つずつ展開していくと次のようになる。



結果が 1 となる部分をすべて足し合わせると最終的に関数が返す値は 6 であることがわかる。

問 14　イ

〔解説〕ハッシュ関数が ｍｏｄ(ａ$_1$＋ａ$_2$＋ａ$_3$＋ａ$_4$＋ａ$_5$、１３）なので、そのまま"５４３２１"を当てはめる。ｍｏｄ()は、第 1 引数を第 2 引数で割った余りを求めるので、

ｍｏｄ(５＋４＋３＋２＋１，１３)＝ｍｏｄ(１５，１３)＝２

したがって、データ"５４３２１"が格納されるのは配列の 2 番目の位置となる。

問 15　ア

〔解説〕問題のプログラムをトレースしていくと…

```
[comp("11", "101")]
  begin
    len(11) =2, len(101)=3 なので、次の処理へ移る。
    first(11) =1, first(101)=1
    first(11) =first(101)なので、comp("1", "01")を呼び出す。

      [comp("1", "01")]
      begin
        len(1) =, len(01)=2 なので、次の処理へ移る。
        first(1) =1, first(01)=0
        first(1)>first(01)なので、−1 を返す。
       end

    comp(1, 01)から返却された−1 を返す。
  end
```

したがって comp("11", "101")の結果は「−1」になります。

9－6　仮想記憶管理〔解答・解説〕

問 1　エ
〔解説〕デマンドページング：
　デマンド(demand)とは要求を意味する英語で、その名の通りアクセス要求があった時に要求があったページのみを主記憶に割り当てる方式。ＯＳで用いられている。

プリページング：
　ページにアクセス要求がある前に、参照されそうなページを主記憶に読み込んでおく方式。

ア、イ、ウ　プリページング方式の説明

問 2　イ
〔解説〕ページング方式では，主記憶装置と補助記憶装置間で固定長のページ単位での入替えを行うので，主記憶装置内にフラグメンテーションが発生せず，使用効率が高くなる。

問 3　ア
〔解説〕スラッシングとは，仮想記憶システムにおいて，ページの置換えが頻繁に発生し，システム性能が低下してしまうことである。

問 4　ア
〔解説〕イ　主記憶を効率的に利用するために、主記憶上で長い間待ち状態となっているプログラムを実行状態のまま補助記憶上の領域に退避させること
ウ　ディスク装置の記憶領域の中に使用されない領域の断片が存在した状態になりアクセス効率が低下する現象
エ　ページング方式において要求されたページが主記憶上に存在しないときに発生する割込み

問 5　ア
〔解説〕スラッシングとは，仮想記憶システムにおいて，ページの置換えが頻繁に発生し，システム性能が低下してしまう現象であり，プログラム多重度が大きいと発生しやすくなる。

問 6　ウ
〔解説〕

割当てステップ	参照する仮想ページ番号	実記憶ページの状態		
1	1	1	—	—
2	4	1	4	—
3	2	1	4	2
4	4	1	4	2
5	1	1	4	2
6	3	3	4	2

問 7　ア

問 8　ア

問 9　ウ

問 10　ウ

〔解説〕ＦＩＦＯ方式は，ページインしてから最も時間の経過したページをページアウトさせる方式であり，主記憶の内容は，

４ → ４，３ → ４，３，２ → ３，２，１ → ２，１，５

と変遷し，そのたびにページインが起こるので，５回となる。

問 11　ウ

〔解説〕ア　仮想記憶の管理はＯＳが行うため，アプリケーションは仮想記憶を利用するためのモジュールを組み込む必要はない。

イ　仮想記憶を利用するために，アプリケーションを磁気ディスクにインストールする必要はない。

エ　個々のアプリケーションにおいて，仮想記憶を使用するという設定は必要ない。

問 12　ウ

〔解説〕ページ枠の遷移を順を追って考えていく。下記のページ枠は、左から４０００，５０００，６０００，７０００番地とする。

(1)：最初の１から４までは設問の指示通りにページインする。

１　２　３　４

(2)：２は主記憶に存在するのでページアウトは発生しない。

１　２　３　４

(3)：５は主記憶に存在しないのでページ置換えが必要になる。この時点で最も昔に参照されたページは１なので、１をページアウトしその位置に５をページインする。

５　２　３　４

(4)：３は主記憶に存在するのでページアウトは発生しない。

５　２　３　４

(5)：１は主記憶に存在しないのでページ置換えが必要になる。この時点で最も昔に参照されたページは４なので、４をページアウトしその位置に１をページインする。

５　２　３　１

(6)：６は主記憶に存在しないのでページ置換えが必要になる。この時点で最も昔に参照されたページは２なので、２をページアウトしその位置に６をページインする。

５　６　３　１

(7)：５は主記憶に存在するのでページアウトは発生しない。

５　６　３　１

(8)：４は主記憶に存在しないのでページ置換えが必要になります。この時点で最も昔に参照されたページは３なので、３をページアウトしその位置に４をページインする。

５　６　４　１

操作終了時点でページ４は左から３番目、つまり６０００番地にページインしている。

よって「ウ」が正解となる。

問 13　ア

〔解説〕イ　ディスク装置の記憶領域の中に使用されない領域の断片が存在した状態になりアクセス効率が低下する現象

ウ　仮想記憶管理方式の一つで仮想アドレス空間と主記憶空間を「ページ」と呼ばれる固定長の区画に分割し、このページ単位で主記憶と補助記憶装置のアドレス変換を行う方式

エ　処理性能や通信性能の向上を阻む支障となっている要素のこと

問 14　ア

問 15　エ

〔解説〕ページインだけの処理の割合を"Ｐ"とすると、ページアウトを伴う処理の割合は"１－Ｐ"で表すことができる。

それぞれの処理時間と平均処理時間の関係を表す次の式を解くと

２０×Ｐ＋６０×(１－Ｐ)＝３０
２０Ｐ＋６０－６０Ｐ＝３０
－４０Ｐ＝－３０
Ｐ＝０.７５

ページインだけの処理の割合"Ｐ"は０.７５であるとわかる。

問 16　ア

〔解説〕ページフォールトとは、プログラムの実行に必要なページが主記憶に存在していないときに発生する割込み。ページフォールトが発生すると、ページアウトやページイン等のページ置換え処理が実行される。

問 17　　エ

〔解説〕ページフォールトは、必要なページが主記憶上に存在しないときに起こるので、必ずそれに伴いページインが発生する。(ページフォールト＝ページイン)

一方、ページフォールトが発生しても主記憶上に空きがある場合には、ページアウトを行わずにその空き領域に必要なページを移動すれば済む。すなわち、ページアウトの回数はページフォールトの回数よりも少なくなる可能性がある。(ページフォールト≧ページアウト)

以上の事からエ(ページフォールト＝ページイン≧ページアウト)が成立する。