FE

平成27年度　秋期
基本情報技術者試験
午後　問題

試験時間	13:00 ～ 15:30（2時間30分）

注意事項

1. 試験開始及び終了は，監督員の時計が基準です。監督員の指示に従ってください。
2. 試験開始の合図があるまで，問題冊子を開いて中を見てはいけません。
3. **答案用紙への受験番号などの記入は，試験開始の合図があってから始めてください。**
4. 問題は，次の表に従って解答してください。

問題番号	問1	問2～問7	問8	問9 ～ 問13
選択方法	必須	4問選択	必須	1問選択

5. 答案用紙の記入に当たっては，次の指示に従ってください。
   (1) 答案用紙は光学式読取り装置で読み取った上で採点しますので，B又はHBの黒鉛筆で答案用紙の**マークの記入方法**のとおりマークしてください。マークの濃度がうすいなど，**マークの記入方法**のとおり正しくマークされていない場合は読み取れません。特にシャープペンシルを使用する際には，マークの濃度に十分注意してください。訂正の場合は，あとが残らないように消しゴムできれいに消し，消しくずを残さないでください。
   (2) **受験番号欄**に**受験番号**を，**生年月日欄**に**受験票の生年月日**を記入及びマークしてください。答案用紙の**マークの記入方法**のとおり記入及びマークされていない場合は，採点されないことがあります。生年月日欄については，受験票の生年月日を訂正した場合でも，訂正前の生年月日を記入及びマークしてください。
   (3) **選択した問題**については，次の例に従って，**選択欄**の**問題番号**の(選)をマークしてください。答案用紙の**マークの記入方法**のとおりマークされていない場合は，採点されません。問2～問7について5問以上マークした場合は，はじめの4問を採点します。問9～問13について2問以上マークした場合は，はじめの1問を採点します。

〔問3，問4，問6，問7，問9を選択した場合の例〕

選択欄								選択欄				
問1	問2	問3	問4	問5	問6	問7	問8	問9	問10	問11	問12	問13
●	(選)	●	●	(選)	●	●	●	●	(選)	(選)	(選)	(選)

   (4) **解答**は，次の例題にならって，**解答欄**にマークしてください。答案用紙の**マークの記入方法**のとおりマークされていない場合は，採点されません。

〔例題〕　次の［　　］に入れる正しい答えを，解答群の中から選べ。

秋の情報処理技術者試験は，［ a ］月に実施される。

解答群　ア　8　　イ　9　　ウ　10　　エ　11

正しい答えは“ウ　10”ですから，次のようにマークしてください。

例題	a	(ア)	(イ)	●	(エ)	(オ)	(カ)	(キ)	(ク)	(ケ)	(コ)

裏表紙の注意事項も，必ず読んでください。

〔問題一覧〕

●問 1（必須問題）

問題番号	出題分野	テーマ
問 1	情報セキュリティ	ログ管理システム

●問 2～問 7（6 問中 4 問選択）

問題番号	出題分野	テーマ
問 2	ハードウェア	浮動小数点数
問 3	データベース	電子部品の出荷データを管理する関係データベースの運用
問 4	ネットワーク	Web サイトにおけるセッション管理
問 5	ソフトウェア設計	決定表を用いた注文機能の設計
問 6	プロジェクトマネジメント	プロジェクトの見積り
問 7	経営戦略・企業と法務	新システム稼働による業績改善

●問 8（必須問題）

問題番号	出題分野	テーマ
問 8	データ構造及びアルゴリズム	Boyer-Moore-Horspool 法を用いた文字列検索

●問 9～問 13（5 問中 1 問選択）

問題番号	出題分野	テーマ
問 9	ソフトウェア開発（C）	入退室状況の印字
問 10	ソフトウェア開発（COBOL）	アンケート結果の分析
問 11	ソフトウェア開発（Java）	ブロックのデータのキャッシュ管理
問 12	ソフトウェア開発（アセンブラ）	ビット列の挿入
問 13	ソフトウェア開発（表計算）	PC 販売店での購入金額の計算

共通に使用される擬似言語の記述形式

擬似言語を使用した問題では，各問題文中に注記がない限り，次の記述形式が適用されているものとする。

〔宣言，注釈及び処理〕

記述形式		説明
○		手続，変数などの名前，型などを宣言する。
/* 文 */		文に注釈を記述する。
処理	・変数 ← 式	変数に式の値を代入する。
	・手続(引数，…)	手続を呼び出し，引数を受け渡す。
	▲ 条件式 　処理 ▼	単岐選択処理を示す。 条件式が真のときは処理を実行する。
	▲ 条件式 　処理 1 ―― 　処理 2 ▼	双岐選択処理を示す。 条件式が真のときは処理 1 を実行し，偽のときは処理 2 を実行する。
	■ 条件式 　処理 ■	前判定繰返し処理を示す。 条件式が真の間，処理を繰り返し実行する。
	■ 　処理 ■ 条件式	後判定繰返し処理を示す。 処理を実行し，条件式が真の間，処理を繰り返し実行する。
	■ 変数：初期値，条件式，増分 　処理 ■	繰返し処理を示す。 開始時点で変数に初期値（式で与えられる）が格納され，条件式が真の間，処理を繰り返す。また，繰り返すごとに，変数に増分（式で与えられる）を加える。

〔演算子と優先順位〕

演算の種類	演算子	優先順位
単項演算	＋，－，not	高 ↑
乗除演算	×，÷，％	
加減演算	＋，－	
関係演算	＞，＜，≧，≦，＝，≠	
論理積	and	
論理和	or	↓ 低

注記　整数同士の除算では，整数の商を結果として返す。％演算子は，剰余算を表す。

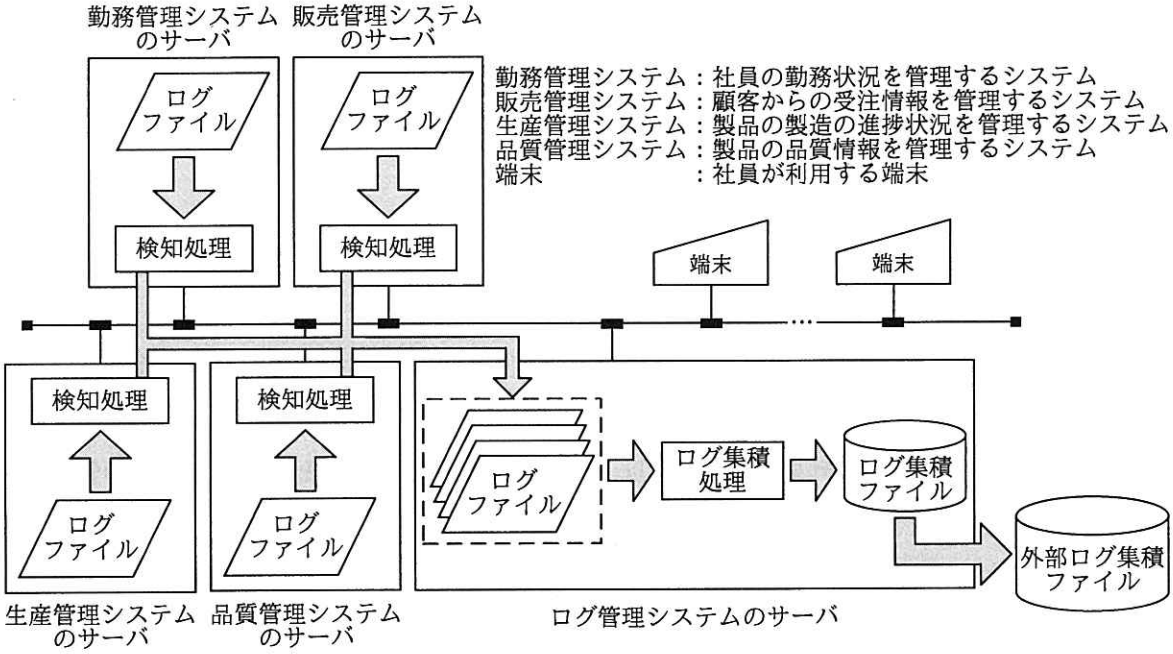
〔論理型の定数〕

true, false

次の問１は必須問題です。必ず解答してください。

問１　ログ管理システムに関する次の記述を読んで，設問１～５に答えよ。

中堅の製造業である B 社では，他社で発生した情報漏えい事件を受けて，社内の業務システムへの不正アクセスを早期に検知するための仕組みを強化することになった。B 社では，業務システムのアクセスログ（以下，ログという）を一元管理するために，ログ管理システムを構築することにした。ログ管理システムの対象になる業務システムは，図１のネットワーク構成図に示す，勤務管理システム，販売管理システム，生産管理システム及び品質管理システムの四つである。各管理システムには，１台のサーバが割り当てられている。



注記　⇨はログを収集する流れを示す。

図１　ネットワーク構成図

〔業務システムの利用とログの説明（抜粋）〕

B 社の社員は，固定の IP アドレスが設定されている端末から，一意に社員を特定

できる社員 ID で，業務システムのうちの一つにログインし，“参照”，“更新”，“ダウンロード”の操作を行う。社員が，業務システムにログインしたときに“参照”のログがログファイルに書き込まれる。また，ダウンロードの都度，そのデータ量を記録したログがログファイルに書き込まれる。一人の社員が，同時に複数の業務システムを使わないこと，及び，業務システム全体からデータを1日に5Mバイトを超えてダウンロードしないことを業務システムの利用規程で定めている。

〔ログ管理システムの概要（抜粋）〕

業務システムの各サーバ上のログファイルにログが書き込まれると，各業務システムに組み込まれている検知処理が，ログの書込みを検知し，そのログをログ管理システムのサーバ上の業務システム別のログファイルに書き込む。書き込まれたログは，ログ管理システムのログ集積処理が，各業務システムのログを一元管理するログ集積ファイルに書き込む。ログには，業務システムを識別するための業務 ID や，社員が実施した操作を示す，“参照”，“更新”，“ダウンロード”の操作種別などが含まれている。

〔ログ管理システムの要件（抜粋）〕

(1) ログ集積ファイルを基に，いつ，誰が，どの端末からどの業務システムをどのように操作したかが追跡できる。

(2) ログ管理システムのサーバ上のログファイルに書き込む処理は，ログ管理システムへのログインを必要とする。

(3) ログ管理システムの管理者（以下，ログ管理者という）と業務システムの管理者（以下，業務システム管理者という）だけが，ログ集積ファイルを参照できる。

(4) ログ管理者は，ログ集積ファイルをログ管理システムから外部の機器に出力することができる。

(5) ログ管理システムから外部の機器に出力される外部ログ集積ファイルには，改ざんと漏えいを防止する対策を講じる。

(6) 各サーバ間の通信には，公開鍵暗号方式を利用する。

(7) ①ログ集積ファイルに書き込まれたログが一定条件を満たした際には，電子メールでログ管理者に通報する。

〔ログ管理システムの概要（抜粋）〕及び〔ログ管理システムの要件（抜粋）〕を基に，表1のログ管理システムの仕組み（抜粋）と，表2のログ管理システムへのアクセス権限表（抜粋）を作成した。

表1　ログ管理システムの仕組み（抜粋）

No.	要件	仕組み
1	ログ管理システムのログ集積ファイルを基に，いつ，誰が，どの端末からどの業務システムをどのように操作したかが追跡できる。	・業務システムに組み込まれた検知処理が，ログ管理システムのサーバ上のログファイルに書き込む。 ・ログファイルのログをログ集積ファイルに書き込む。 ・［ a ］。
2	ログ管理システムから外部の機器に出力される外部ログ集積ファイルには，改ざんと漏えいを防止する対策を講じる。	・［ b ］。 ・［ c ］。

表2　ログ管理システムへのアクセス権限表（抜粋）

	ログ管理システムへのログイン	ログファイルへのアクセス	ログ集積ファイルへのアクセス
ログ管理者	可		RE
業務システム管理者	可		R
検知処理	［ d1 ］	［ d2 ］	

注記　網掛けの部分は表示していない。
　　Rは参照，Eは外部へ出力，Wは書込みを示す。

設問1　表1中の［　　］に入れる要件を満たす仕組みとして適切な答えを，解答群の中から選べ。

aに関する解答群

ア　各業務システムの稼働状況を監視する

イ　各業務システムの時刻を同期させる

ウ　検知処理のログ管理システムへのアクセスを監視する

エ　ログ集積ファイルへのアクセスを監視する

オ　ログ集積ファイルを圧縮する

b，cに関する解答群

ア　同一内容の複数個のログ集積ファイルを出力する

イ　ログ集積ファイルに電子署名を付加する

ウ　ログ集積ファイルの出力に当たっては，推測しにくい名称を付ける

エ　ログ集積ファイルのログ中の個人情報を削除する

オ　ログ集積ファイルを圧縮する

カ　ログ集積ファイルを暗号化する

設問2　ログ管理システムの要件を満たすために，日時，操作種別以外で全てのログに共通して含むべき項目を全て挙げた適切な答えを，解答群の中から選べ。

解答群

ア　業務ID，社員ID

イ　業務システムのサーバのIPアドレス，業務ID

ウ　業務システムのサーバのIPアドレス，業務ID，社員ID

エ　端末のIPアドレス，業務ID，社員ID

オ　端末のIPアドレス，社員ID

設問3　表2中の［　　］に入れる適切な答えを，解答群の中から選べ。ここで，d1とd2に入れる答えは，解答群の中から組合せとして適切なものを選ぶものとする。

解答群

	d1	d2
ア	可	RE
イ	可	W
ウ	不可	E
エ	不可	RE
オ	不可	RW
カ	不可	W

設問4　業務システムの検知処理はログ管理システムのサーバ上のログファイルへ書き込む。この通信を暗号化するために最低限必要な公開鍵の数として適切な答えを，解答群の中から選べ。

解答群

ア　1　　イ　4　　ウ　8　　エ　12

設問5　ログ集積ファイルを基に，業務システムへの不正アクセスを早期に検知するために，〔ログ管理システムの要件（抜粋）〕の下線①で言及している一定条件として適切な答えを，解答群の中から二つ選べ。ここで，解答群は，同じ社員 ID のログに対する条件とする。

解答群

ア　1日中“参照”のログだけが書き込まれたとき

イ　1日の間に“更新”のログが1回以上，書き込まれたとき

ウ　ある業務システムの連続した“更新”のログの間に，別の業務システムのログが書き込まれたとき

エ　同じ業務システムの“参照”と“更新”のログが連続して書き込まれたとき

オ　業務システムからダウンロードされたデータ量が1日で 5M バイトを超えたとき

カ　特定の業務システムの“参照”のログが 15 分間，書き込まれていないとき

次の問2から問7までの6問については，この中から4問を選択し，選択した問題については，答案用紙の選択欄の(選)をマークして解答してください。

なお，5問以上マークした場合には，はじめの4問について採点します。

問2　浮動小数点数に関する次の記述を読んで，設問1～4に答えよ。

$\alpha = 0$，又は $1 \leqq |\alpha| < 2$ を満たす α，及び $-126 \leqq \beta \leqq 127$ を満たす β を用いて $\alpha \times 2^{\beta}$ の形で表記される浮動小数点数を，図1に示す32ビット単精度浮動小数点形式の表現（以下，単精度表現という）で近似する。

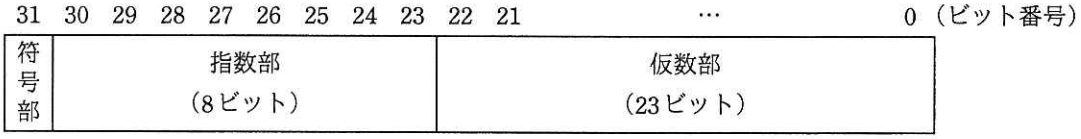


図1　32ビット単精度浮動小数点形式

(1)　符号部（ビット番号31）

α の値が正のとき0，負のとき1が入る。

(2)　指数部（ビット番号30～23）

β の値に127を加えた値が2進数で入る。

(3)　仮数部（ビット番号22～0）

$|\alpha|$ の整数部分1を省略し，残りの小数部分が，ビット番号22に小数第1位が来るような2進数で入る。このとき，仮数部に格納できない部分については切り捨てる。

(4)　α の値が0の場合，符号部，指数部，仮数部ともに0とする。

なお，値の記述として，単に α と記述した場合は，α は10進数表記であり，$(\alpha)_n$ と記述した場合は α がn進数表記であることを示す。例えば，$(0.101)_2$ は0.625と同じ値を表す。また，00…0という表記は，0が連続していることを表す。

設問１　0.625 を単精度表現したときに指数部に入る値として正しい答えを，解答群の中から選べ。

解答群

ア　$(00)_{16}$　　イ　$(7E)_{16}$　　ウ　$(7F)_{16}$　　エ　$(FE)_{16}$

オ　$(FF)_{16}$

設問２　次の単精度表現された数値として正しい答えを，解答群の中から選べ。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	…	0
0	0	1	1	1	1	1	1	0	1	0	0	0	0	0	…	0

解答群

ア　0.125　　イ　0.25　　ウ　0.375　　エ　0.5

オ　0.75　　カ　1.5

設問３　次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。

二つの浮動小数点数 A と B の加算を行う。

A の単精度表現

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	…	0
0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	…	0

B の単精度表現

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	…	0
1	1	0	0	0	0	0	1	1	1	0	0	0	0	0	…	0

A と B の加算を，次の①，②の手順で行う。

① 指数部の値を大きい方に合わせる。A が $(1.1)_2 \times 2^5$ であることから，B を $(-([\ \text{a}\ ])_2) \times 2^5$ とする。

② 加算を行う。

$((1.1)_2 + (-([\ \text{a}\ ])_2)) \times 2^5 = (1.1)_2 \times 2^{[\ \text{b}\ ]}$

aに関する解答群

ア 0.001	イ 0.01	ウ 0.011	エ 0.1
オ 0.11	カ 1.1		

bに関する解答群

ア 3	イ 4	ウ 5	エ 6
オ 130	カ 131	キ 132	

設問4　次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。

設問3のAについてA×10の値は，次の①～③の手順で求めることができる。

① A×8の値を求める。

$A \times 8 = (1.1)_2 \times 2^5 \times 8 = (1.1)_2 \times 2^5 \times 2^3 = (1.1)_2 \times 2^8$

② A×2の値を同様に求める。

③ ①と②の結果を加算する。

加算結果を単精度表現すると，［ c ］になる。

cに関する解答群

ア

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	…	0
0	0	0	0	0	1	0	0	0	1	1	1	0	0	0	…	0

イ

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	…	0
0	0	0	0	0	1	0	0	0	1	1	1	1	0	0	…	0

ウ

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	…	0
0	1	0	0	0	0	1	1	1	1	1	1	0	0	0	…	0

エ

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	…	0
0	1	0	0	0	0	1	1	1	1	1	1	1	0	0	…	0

オ

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	…	0
0	1	0	0	0	1	0	0	0	1	1	1	0	0	0	…	0

カ

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	…	0
0	1	0	0	0	1	0	0	0	1	1	1	1	0	0	…	0

問3　電子部品の出荷データを管理する関係データベースの運用に関する次の記述を読んで，設問1～4に答えよ。

C社は，電子部品を製造販売する会社である。

ある期間に出荷した特定の電子部品について，製造装置の設定ミスによる不具合が発生しているおそれがあるので，顧客への連絡と出荷済みの電子部品の無償交換（回収及び再出荷。以下，リコールという）を実施することにした。

出荷情報は，図1に示す表で管理されている。下線付きの項目は主キーを表す。

顧客表

顧客番号	顧客名	住所	代表電話
0181	情報電機株式会社	東京都文京区本駒込○○－△	99-9999-9999

部品表

部品番号	部品名	単価
007551	スイッチ	80

出荷表

出荷番号	顧客番号	部品番号	出荷数	出荷金額	出荷日
150412	0181	007551	400	32000	20150115
150413	0059	000890	100	48000	20150115

図1　表の構成とデータの格納例

設問1　リコールの対象となる電子部品の出荷先の顧客番号，顧客名，出荷番号，出荷日，出荷数を，顧客番号の昇順に表示する。リコールの対象となる電子部品の部品番号は“007551”で，出荷日は2015年1月10日から2015年1月20日までである。次のSQL文の［ a ］に入れる正しい答えを，解答群の中から選べ。

```
SELECT 顧客表.顧客番号, 顧客表.顧客名,
       出荷表.出荷番号, 出荷表.出荷日, 出荷表.出荷数
  FROM 顧客表, 出荷表
  WHERE 出荷表.顧客番号 = 顧客表.顧客番号 AND
        出荷表.部品番号 = '007551' AND
        [ a ]
  ORDER BY 顧客表.顧客番号
```

解答群

ア　出荷表.出荷日 = '20150110' OR 出荷表.出荷日 = '20150120'

イ　出荷表.出荷日 = ANY ('20150110', '20150120')

ウ　出荷表.出荷日 BETWEEN '20150110' AND '20150120'

エ　出荷表.出荷日 IN ('20150110', '20150120')

設問2　C社では，電子部品を単品で出荷するだけでなく，複数の電子部品を同梱(こん)したパッケージも出荷している。このパッケージにも一意の部品番号が割り振られている。パッケージの同梱部品の情報は，図2に示すパッケージ表で管理されている。

リコールの対象となる電子部品がパッケージにも含まれていることが判明したので，該当するパッケージの出荷情報も含めて表示するよう設問1のSQL文を変更する。次のSQL文の［　b　］に入れる正しい答えを，解答群の中から選べ。ここで，［　a　］には設問1の正しい答えが入っているものとする。また，パッケージの同梱部品にパッケージが含まれることはない。

部品表

部品番号	部品名	単価
009220	スイッチモジュール	400

パッケージ表

部品番号	同梱部品	同梱点数
009220	000058	1
009220	007551	3

図2　パッケージ表の構成とデータの格納例

```
SELECT 顧客表.顧客番号, 顧客表.顧客名,
       出荷表.出荷番号, 出荷表.出荷日, 出荷表.出荷数
  FROM 顧客表, 出荷表
  WHERE 出荷表.顧客番号 = 顧客表.顧客番号 AND
        [ b ] AND
        [ a ]
  ORDER BY 顧客表.顧客番号
```

解答群

ア (出荷表.部品番号 = '007551'
AND 出荷表.部品番号 = ANY
(SELECT パッケージ表.同梱部品 FROM パッケージ表))

イ (出荷表.部品番号 = '007551'
AND 出荷表.部品番号 = ANY
(SELECT パッケージ表.部品番号 FROM パッケージ表))

ウ (出荷表.部品番号 = '007551'
OR 出荷表.部品番号 = ANY
(SELECT パッケージ表.同梱部品 FROM パッケージ表
WHERE パッケージ表.部品番号 = '007551'))

エ (出荷表.部品番号 = '007551'
OR 出荷表.部品番号 = ANY
(SELECT パッケージ表.部品番号 FROM パッケージ表
WHERE パッケージ表.同梱部品 = '007551'))

設問3　今回のリコールの対象となる電子部品の出荷金額の合計を表示する。次のSQL 文の［ c ］に入れる正しい答えを，解答群の中から選べ。ここで，［ a ］と［ b ］には設問1及び設問2の正しい答えが入っているものとする。

SELECT ［ c ］ AS 合計出荷金額
FROM 出荷表
WHERE ［ b ］ AND
［ a ］

解答群

ア AVG(出荷表.出荷金額)

イ COUNT(出荷表.出荷金額)

ウ MAX(出荷表.出荷金額)

エ SUM(出荷表.出荷金額)

設問4　回収の対象となった出荷の情報は残したまま，再出荷に関する情報を管理することができるように，表の構成を変更する。次の記述中の［　　］に入れる適切な答えを，解答群の中から選べ。

図 1 及び図 2 の表に回収及び再出荷の情報を追加する場合，［　d　］に回収日と再出荷番号の項目を追加し，初期値には NULL を設定しておき，回収対象の場合には回収した日と再出荷時の出荷番号を設定すればよい。ただし，この方法では既存データへの影響が大きく，また，リコールの頻度が低い場合は効率が悪い。

そこで，既存データに影響を与えない方法として，新たに回収表を作成して，一意に割り振った回収番号，回収対象となった出荷の出荷番号，回収日，再出荷時の出荷番号を格納する方法を考えた。この方法では，例えば，ある月の出荷金額の合計を求めるとき，回収対象となった出荷の出荷金額を除いて求めたい場合は，［　e　］から集計できる。

dに関する解答群

ア　顧客表　　イ　出荷表　　ウ　パッケージ表　　エ　部品表

eに関する解答群

ア　出荷表と回収表　　イ　パッケージ表と回収表

ウ　部品表と回収表　　エ　部品表と出荷表とパッケージ表

オ　部品表とパッケージ表と回収表

問4　Web サイトにおけるセッション管理に関する次の記述を読んで，設問1～4に答えよ。

セッションとは，一連の処理の始まりから終わりまでを表す概念である。例えば，あるショッピングサイトでは，会員がログインし，その後，商品の選択，注文，決済など，何度も Web サイトへのアクセスを繰り返しながら商品を購入し，最後にログアウトする。このときの，ログインからログアウトまでが同じ一つのセッションである。

Web サイトへのアクセスに使うプロトコルである HTTP ではセッションの扱いについての規定はないが，Web サイトと Web ブラウザの間でセッション ID を送受信することで，一連の通信を一つのセッションとして管理できる。ここで，セッション ID とはセッションごとに割り振られた一意の文字列である。

Web サイトは，セッションの開始とともにセッション ID を生成し，Web ブラウザに送る。Web ブラウザは，Web サイトから受信したセッション ID を含めた HTTP リクエストを Web サイトに送る。Web サイトは，同じセッション ID をもつ HTTP リクエストを，同一セッションの一連の HTTP リクエストとみなす。一般に，会員 ID や選択された商品の情報などのセッションに関係する情報は，Web サイト側がセッション ID に関連付けて管理する。

セッション ID は注意して取り扱う必要がある。例えば，セッション ID の有効期間をできるだけ短くしたり，セッション ID を推測しにくい文字列にしたり，セッション ID の送受信を暗号化されている通信路で行ったりするなど，セキュリティ上のリスクを抑える工夫をする。

セッション ID の送受信には，主に次に挙げる方法が用いられている。

(1) HTTP リクエストの拡張ヘッダや Web サイトが Web ブラウザに送信する HTTP レスポンスの拡張ヘッダに，クッキーの値としてセッション ID を記載する。このとき，Web ブラウザでクッキーの管理が有効になっている必要がある。

(2) HTML 中のリンク先やフォームの送信先を示す URL の中にセッション ID を埋め込む（次の例では下線の箇所）。

例：top

(3) HTML 中のフォームでフィールド hidden にセッション ID を埋め込む（次の例では下線の箇所）。

例：<input type="hidden" name="sid" value="セッション ID">

設問1　次の記述中の［　　］に入れる適切な答えを，解答群の中から選べ。

ショッピングサイト A での商品購入の流れは図 1 のとおりである。注文と決済に必要な情報をセッション ID に関連付けて管理するため，ショッピングサイト A では，閲覧者が［ a ］にセッション ID を生成し，ログアウト時に破棄することとした。

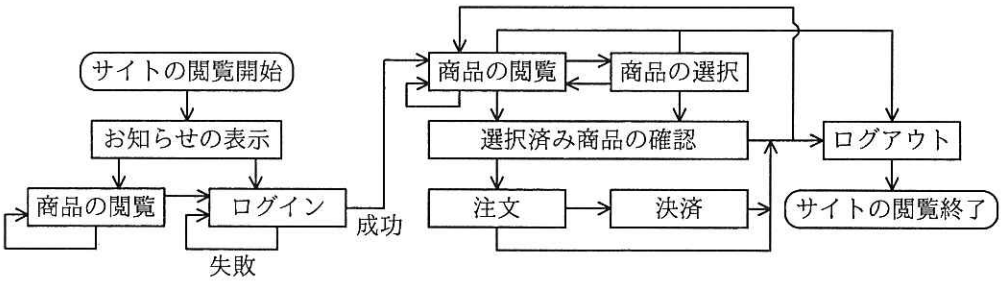


図 1　ショッピングサイト A での商品購入の流れ

a に関する解答群

ア　サイトの閲覧を開始したとき　　イ　商品を閲覧するたび

ウ　商品を選択するたび　　エ　ログインに成功したとき

設問2　セッション ID として使う文字列として適切な答えを，解答群の中から選べ。

解答群

ア　会員 ID と同じ文字列

イ　会員 ID と通し番号を連結した文字列

ウ　十分に長いランダムな文字列

エ　通し番号を示す文字列

設問3　次に示す表は，(A)～(C)の特徴と，本文中のセッションIDを送受信する(1)～(3)の方法の組合せを示したものである。表中の［　　］に入れる適切な答えを，解答群の中から選べ。

	特徴	方法
(A)	Webブラウザのアドレスバーに表示されるURLの中にセッションIDが含まれる。	b
(B)	Webブラウザの設定次第で利用できないことがある。	
(C)	タグ<a>で指定されたリンクのクリックではセッションIDが送信されない。	

bに関する解答群

ア	イ	ウ	エ	オ	カ
(1)	(1)	(2)	(2)	(3)	(3)
(2)	(3)	(1)	(3)	(1)	(2)
(3)	(2)	(3)	(1)	(2)	(1)

設問4　次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。

クッキーは名前と値の組であり，Webサイトと Webブラウザでそれぞれ管理される。Webサイトは，Webブラウザに管理させたいクッキーを，Webブラウザに送信するHTTPレスポンスの拡張ヘッダに記載する。Webブラウザは，Webサイトから受信したクッキーを，その Webサイトに送信するHTTPリクエストの拡張ヘッダに記載する。

クッキーの値としてセッションIDを記載することで，WebサイトとWebブラウザの間で，セッションIDを送受信することができる。

Webサイトは，HTTPレスポンスに記載するクッキーに，付加情報として送信先ドメイン名の指示を加えることができる。これは，Webブラウザに対し，当該クッキーをどのドメイン又はホスト宛ての HTTPリクエストに記載すべきかを指示するもので，HTTPレスポンスの送信元ホスト名か，より上位のドメイン名が指示されているクッキーだけが有効である。例えば，ホスト www.example.com の上位のドメインは example.com と com である。ただし，多くの組織の上位ドメインとなる com や org などは有効とはみなさない。Webブ

ラウザは，有効でないドメイン名が指示されたクッキーを管理しない。

Web ブラウザは，管理しているクッキーを，指示されたドメイン名と等しいホストか，より下位ドメインのホスト宛てに送信する HTTP リクエストに記載する。例えば，送信先ドメイン名として，example.com が指示されているクッキーは，example.com の下位ドメインのホストである，www.example.com や www.foo.example.com 宛てに送信する HTTP リクエストに記載される。

送信先ドメイン名として example.com が指示されたクッキーを www.example.com から受け取った Web ブラウザは，www.example.com の他，www2.example.com などの example.com の下位ドメインのホストへ送信する HTTP リクエストにも，当該クッキーを記載する。

クッキーを受け入れる設定であって，かつ，クッキーを一つも管理していない Web ブラウザから http://www.foo.example.com/index.html にアクセスし，その応答として受け取った HTTP レスポンスには，表 1 に挙げる名前をもつクッキーが含まれており，それぞれに，送信先ドメイン名の指示が付加されていた。Web ブラウザは，このうちの［ c ］個のクッキーを管理する。この直後に，同じ Web ブラウザから http://www.bar.example.com/index.html にアクセスするときに HTTP リクエストに記載されるクッキーは，［ d ］である。

表 1　クッキーの名前と送信先ドメイン名の指示

名前	送信先ドメイン名の指示
c1	example.com
c2	foo.example.com
c3	www.foo.example.com
c4	www.example.com
c5	www.bar.example.com

c に関する解答群

ア　1　　イ　2　　ウ　3　　エ　4　　オ　5

d に関する解答群

ア　c1　　イ　c1 と c2　　ウ　c1 と c5　　エ　c4 と c5　　オ　c5

問5　決定表を用いた注文機能の設計に関する次の記述を読んで，設問1，2に答えよ。

T社は，スポーツ用品の小売業者である。このたび，店舗での販売だけでなく，会員制のWebサイトを構築し，インターネット販売を開始することにした。

情報システム部に所属するAさんは，注文機能を構成する処理の一つである，注文確定をさせる前の処理（以下，注文確定前処理という）の設計を担当している。注文機能の流れを，図1に示す。

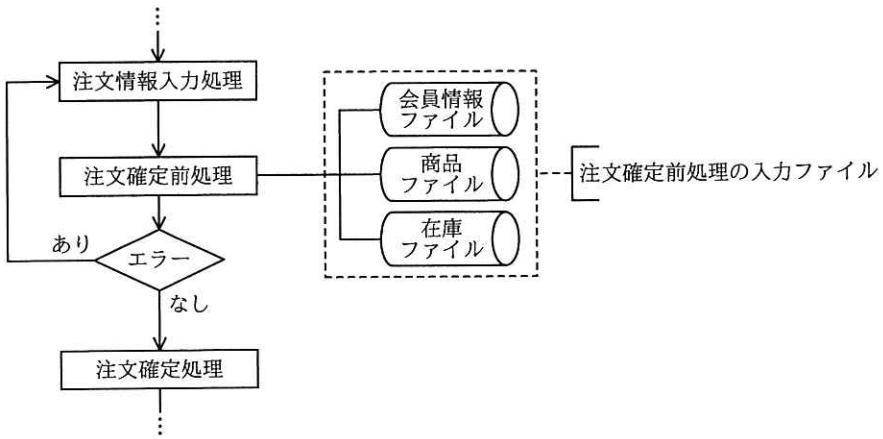


図1　注文機能の流れ

注文確定前処理は，注文情報入力処理（以下，入力処理という）で注文情報の入力が完了した後に呼び出される。注文情報と，入力ファイルの情報を参照して，注文確定可否のチェック，注文額の計算などを行う。注文確定前処理において，エラーがない場合は注文確定処理（以下，確定処理という）へ進み，エラーがある場合は入力処理に戻る。

注文情報を表1に，注文確定前処理が参照する入力ファイルの主な項目を表2に示す。

E
後

表1　注文情報

項目	項目の説明
希望納期	会員が配達を希望する日付
届け先情報[1)]	商品の届け先情報（宛名，郵便番号，住所，電話番号）
請求先区分[1)]	請求先の選択項目（“届け先と同じ”，“指定する”）
請求先情報	請求書の送付先情報（宛名，郵便番号，住所，電話番号）。請求先区分が“指定する”の場合は入力項目であり，“届け先と同じ”の場合は入力不可項目となる。
購入対象商品[1)]	会員が，購入対象として選んだ商品の商品コードと，注文した数。1回の注文で，1～10件が指定できる。

注[1)]　必須入力項目である。

表2　注文確定前処理が参照する入力ファイルの主な項目

ファイル	主な項目
会員情報ファイル	会員番号，会員名，郵便番号，住所，電話番号，Eメールアドレス，累計購入額，会員区分
商品ファイル	商品コード，商品名，単価
在庫ファイル	商品コード，倉庫コード，在庫数

注記　下線付きの項目は主キーを表す。

〔注文確定前処理の概要〕

次の(1)～(8)を順に処理する。ただし，処理中にエラーを検出した場合は，該当するエラーメッセージを表示して入力処理に戻る。エラーを検出しなかった場合は，注文確定情報として確定処理に受け渡す。

(1)　必須入力項目の入力チェックを行い，未入力の項目がある場合は，必須項目未入力エラーとする。

(2)　希望納期が未入力の場合は，注文確定前処理を実行している日（以下，処理日という）の1週間後の日付を希望納期として設定する。

(3)　希望納期が処理日以前の場合は，項目関連エラーとする。

(4)　請求先区分が“指定する”であって，請求先情報が未入力である場合は，項目関連エラーとする。

(5)　請求先区分が“届け先と同じ”である場合は，請求先情報に，届け先情報と同じ値を設定する。

(6)　購入対象商品の全てについて，在庫ファイルから在庫数を取得する。

(7)　在庫数が，注文を受けた数（以下，注文数という）よりも少ない商品が存在す

る場合は，在庫不足エラーとする。

(8) 購入対象商品の全てについて，在庫ファイルの在庫数を更新し，注文額及び①割引額を計算する。

A さんは，注文確定前処理の設計に当たって，決定表を用いて仕様を整理した。注文確定前処理の決定表の一部を表 3 と表 4 に示す。

表 3　注文確定前処理の決定表（一部）

注文確定前処理（入力情報整合性チェック）									
条件部	全ての必須入力項目が入力されている	Y	Y	Y	Y	Y	Y	Y	N
	a	Y	Y	Y	Y	N	N	N	−
	b	Y	Y	Y	N	−	−	−	−
	c	Y	Y	N	−	Y	Y	N	−
	d	Y	N	−	−	Y	N	−	−
動作部	必須項目未入力エラーメッセージを表示する	−	−	−	−	−	−	−	X
	希望納期に処理日の 1 週間後の日付を設定する	−	−	−	−	X	X	X	−
	項目関連エラーメッセージを表示する	−	X	−	X	e	X	−	−
	請求先情報に届け先情報と同じ値を設定する	−	−	X	−		−	X	−
	在庫数を取得して表 4 を処理する	X	−	X	−		−	X	−

注記　決定表の条件部の条件の組合せと，それに対応する動作部の組みを，規則という。条件部に記載した条件の間の関係は，論理積（AND）であり，上から順に条件が評価される。“Y”は条件が真，“N”は条件が偽，“−”は真偽に関係ないことを表す。動作部の“X”は条件が全て満たされたとき記述された動作を実行することを，“−”は実行しないことを表す。動作部は上から順に動作を実行する。

表 4　注文確定前処理の決定表（一部）

注文確定前処理（在庫確認，更新及び注文額・割引額の計算）			
条件部	在庫数＜注文数	Y	N
動作部	在庫不足エラーメッセージを表示する	X	−
	在庫ファイルを更新し，注文額及び割引額の計算を行う	−	X

設問1　表 3 の注文確定前処理の決定表（一部）中の［　　］に入れる正しい答えを，解答群の中から選べ。

a～dに関する解答群

ア　希望納期＞処理日　　イ　希望納期≦処理日

ウ　希望納期が入力されている　　エ　希望納期が未入力である

オ　請求先区分が“指定する”である　　カ　請求先区分が“届け先と同じ”である

キ　請求先情報が入力されている　　ク　請求先情報が未入力である

eに関する解答群

ア	イ	ウ	エ	オ	カ
－	－	－	X	X	X
－	－	X	－	－	X
－	X	X	－	X	－

設問2　〔注文確定前処理の概要〕の下線①の割引額について説明する。T 社では，会員を無料会員と有料会員に分類（以下，会員区分という）している。年会費 5,000 円を支払って有料会員になると，無料会員よりも優遇された割引サービスを受けることができる。割引サービスとは，直近 1 年間の累計購入額（以下，累計購入額という）に応じて，注文時の注文額に一定の割引率を適用して割り引いたものを購入額としたり，送料を無料としたりするサービスである。

なお，累計購入額の算出に当たっては，過去の購入実績だけを対象とする。T 社の割引サービスを表 5 に示す。

表5　T社の割引サービス

累計購入額 / 会員区分	10万円未満	10万円以上20万円未満	20万円以上
無料会員	・なし	・注文額から10%を割り引く	・注文額から15%を割り引く ・送料無料
有料会員	・注文額から10%を割り引く	・注文額から15%を割り引く ・送料無料	・注文額から20%を割り引く ・送料無料

表6は，表5で示した割引サービスの要件を整理するために作成中の決定表であり，動作部は未記入である。Aさんは，この決定表から発生し得ない無効な条件の組合せを抽出し，決定表から削除することによって，テストケースの設計にも活用したいと考えている。表6の決定表において，削除できる規則は幾つあるか，正しい答えを，解答群の中から選べ。

表6　割引サービスの要件を整理するために作成中の決定表

割引サービスの選択																
条件部	無料会員である	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	N	N	N
	累計購入額＜10万円	Y	Y	Y	Y	N	N	N	N	Y	Y	Y	Y	N	N	N	N
	10万円≦累計購入額＜20万円	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N	Y	Y	N	N
	累計購入額≧20万円	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N
動作部	注文額から10%を割り引く																
	注文額から15%を割り引く																
	注文額から20%を割り引く																
	送料無料																

解答群

ア　6　　イ　7　　ウ　8　　エ　9　　オ　10

カ　11　　キ　12

問6　プロジェクトの見積りに関する次の説明を読んで，設問1～3に答えよ。

P社は，得意先であるQ社の業務システム開発を受託した。今回は，前回開発したシステムの機能拡張であり，前回と同じメンバによるプロジェクトチーム（以下，チームという）が開発を担当することになった。

〔プロジェクトの概要〕

(1)　このチームの工程別の生産性基準値は，表1のとおりである。

表1　工程別の生産性基準値

工程	設計	プログラミング	テスト
生産性（kステップ／人時）	0.05	0.1	0.1

(2)　メンバの総数は5人である。

(3)　各メンバの1日の作業時間は8時間，1週間の作業日数は5日である。また，メンバの生産性は，全員等しいとする。

(4)　各工程の途中段階で開発規模の再見積りを行い，各工程の分担を見直すことにする。

設問1　設計工程及び設計工程での再見積り後の予測に関する次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。

設計工程に着手する前の時点（以下，当初という）に見積もった開発規模は，100kステップであった。当初の見積り規模及び表1に示す生産性基準値から算出した設計工程の所要工数は，［　a　］人時である。

設計工程では，当初の見積り規模を基に，メンバ全員が設計を均等に分担して，全員が同時に設計を開始した。第1週から第8週まで，全員が毎週40時間を消費した。設計が進んだ結果，各メンバが分担している開発規模が均等でないことが判明した。

現在は設計工程の第8週末の時点であり，現在の設計進捗率を基に開発規模の再見積りを行った。再見積り後の開発規模は［　b　］kステップであり，設計

～テスト工程の総工数は［ c ］人時となる。第9週以降もこのままの分担で設計を続けた場合，各メンバの設計終了までに要する予測時間は，表2のとおりである。

表2　第8週末の進捗率と第9週以降の各メンバの予測時間（週単位）

メンバ名	進捗率（%）	各週の予測時間（時）			
		9週	10週	11週	12週
A	80	40	40		
B	約73	40	40	40	
C	約89	40			
D	約70	40	40	40	20
E	約76	40	40	20	
全体	約77	200	160	100	20

aに関する解答群

ア　1,000　　イ　1,500　　ウ　2,000　　エ　2,500

bに関する解答群

ア　102　　イ　104　　ウ　106　　エ　108

cに関する解答群

ア　4,080　　イ　4,160　　ウ　4,240　　エ　4,360

設問2　第8週末時点での進捗に関する次の記述中の［　　］に入れる適切な答えを，解答群の中から選べ。

表2から，［ d ］の設計工数が当初の計画値を超えている。このままの分担で継続すると納期に遅延が生じるので，第9週以降の作業分担について見直しを行うことにした。

dに関する解答群

ア　B及びDの2人　　イ　B，D及びEの3人

ウ　C以外の4人　　エ　Dのただ1人

設問３　各メンバの開発に掛かる工数が均等になるように，第９週以降の作業分担の見直し案として次の二つの案を考えた。残りの設計の途中で設計変更が発生するリスクの影響を考慮して，どちらの案を採用するかを検討したい。ここで，設計変更が発生しても開発規模は変わらないものとする。評価として**誤っている**答えを，解答群の中から選べ。

第１案は，残りの設計を表２に示した当初の分担のまま実行し，各メンバが設計を終了し次第，プログラミングに着手する。このとき，再見積り後の開発規模を基に全員のプログラミングの終了日がそろうようにプログラミングの分担を割り振る。かつ，テストの分担は全員が均等になるように割り振る。

第２案は，全員の設計の終了日がそろうように残りの設計の分担を割り振る。かつ，プログラミングとテストの分担もそれぞれ全員が均等になるように割り振り，全員のプログラミングの開始日及び終了日をそろえる。

解答群

ア　設計変更が発生しなかった場合，プログラミング工程の終了日は第１案の方が第２案よりも早くなる。

イ　設計変更が発生した場合，第１案では，設計工程のコスト増及びスケジュール遅延だけでなく，その影響で，既に着手していたプログラミングの手戻りなどが発生して，プログラミング工程のコスト増が発生する可能性がある。

ウ　設計変更が発生した場合，第２案では，設計工程のコスト増及びスケジュール遅延は発生するが，その影響で，プログラミング工程のコスト増は発生しない。

問7　新システム稼働による業績改善に関する次の記述を読んで，設問1，2に答えよ。

消費財メーカのZ社は，営業支援とコスト管理のための新システムを開発している。Z社には五つの事業部があり，各事業部の2015年度の売上高と営業利益の見込みは表1のとおりである。各事業部は，2016年度初日からの新システム稼働によって，2016年度に表2の業績改善を期待している。ここで，営業利益率は売上高に対する営業利益の比率である。

Z社は，表1，2を基に，各事業部の2016年度の業績について予想することにした。ここで，2016年度の売上高と営業利益が2015年度から変動する要因は，新システム稼働による業績改善だけとする。

表1　各事業部の2015年度の売上高と営業利益の見込み

単位　億円

事業部	売上高	営業利益
P	180	14
Q	100	12
R	60	1
S	50	4
T	10	−1
合計	400	30

表2　各事業部の2016年度に期待する業績改善（対2015年度）

事業部	売上高	利益の改善
P	影響なし	営業利益を10%増加
Q	5%増加	営業利益率を維持
R	10%増加	営業利益を20%増加
S	影響なし	営業利益率を10%に引上げ
T	50%増加	営業利益を3億円増加

E
後

設問1　2016 年度の業績の予想に関する次の記述中の［　　］に入れる適切な答えを，解答群の中から選べ。

表 1，2 を基に各事業部の 2016 年度の売上高と営業利益を予想した結果，及び売上高の事業部構成比と各事業部の営業利益率を表 3 に示す。

表 3　各事業部の売上高と営業利益

事業部	2015 年度				2016 年度			
	売上高（億円）	構成比（%）	営業利益（億円）	営業利益率（%）	売上高（億円）	構成比（%）	営業利益（億円）	営業利益率（%）
P	180	45.0	14.0	7.8	180	43.3		
Q	100	25.0	12.0	12.0			12.6	12.0
R	60	15.0	1.0	1.7			1.2	1.8
S	50	12.5	4.0	8.0	50	12.0		
T	10	2.5	−1.0	−10.0			2.0	13.3
合計	400	100.0	30.0	7.5	416	100.0	36.2	8.7

注記　網掛け部分は表示していない。
営業利益率は小数第 2 位を四捨五入している。

表 3 から，新システム稼働による売上高への効果は，16 億円を期待できる。また，2015 年度から 2016 年度に掛けて売上高の増加額が最も大きいのは［ a ］事業部である。2015 年度と 2016 年度それぞれの売上高の事業部構成比を多重円グラフに表すと，図 1 のとおりになる。ここで，多重円グラフの内側が 2015 年度の構成比，外側が 2016 年度の構成比である。

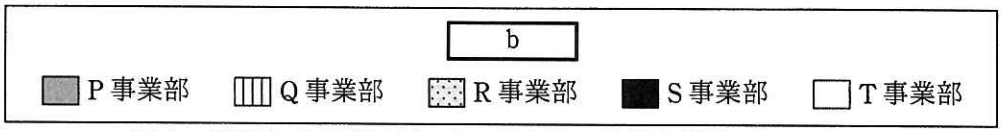


図 1　2015 年度と 2016 年度の売上高の事業部構成比

表 3 から，2016 年度の期待する営業利益率が最も大きいのは，［ c ］事業部である。また，2016 年度の各事業部の期待する営業利益をパレート図に表すと，図 2 のとおりになる。

d

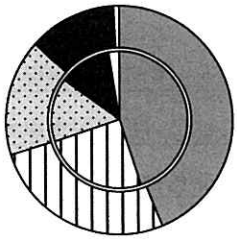
図2　2016年度の各事業部の営業利益（パレート図）

a，cに関する解答群

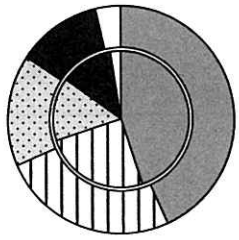
ア　P　　イ　Q　　ウ　R　　エ　S　　オ　T

bに関する解答群

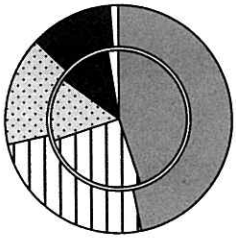
ア



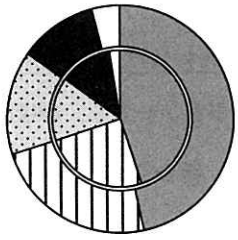
イ



ウ

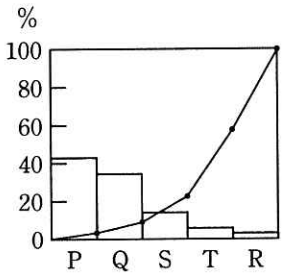


エ

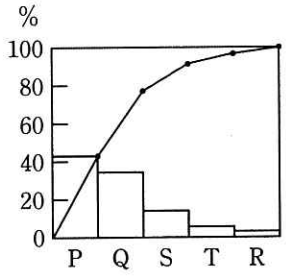


dに関する解答群

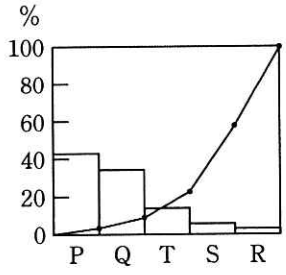
ア



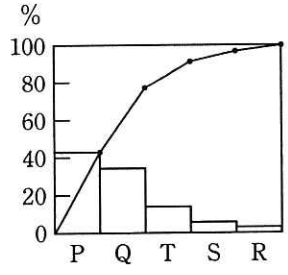
イ



ウ

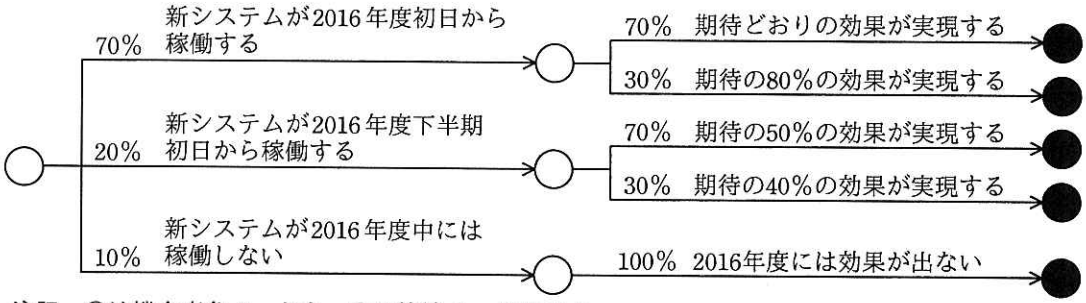


エ



設問２　Ｚ社では，現在開発している新システムの稼働開始が遅延するリスクと，期待している効果が見込みよりも小さくなるリスクを考慮して，2016 年度の業績を予想することにした。確率を考慮した業績の予想に関する次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。

Ｚ社が想定した，新システムが稼働する時期と効果の実現度合いは，図３に示す決定木のとおりである。



注記　○は機会事象ノードを，●は終端ノードを示す。

図３　新システムが稼働する時期と効果の実現度合いに関する決定木

図３から，新システムが予定どおり 2016 年度初日から稼働して，期待どおりの効果を実現する確率は，［　e　］。

同様に，新システムが稼働する時期と効果の実現度合いそれぞれの確率を考慮すると，2016 年度の事業部の売上高合計の期待値を千万円の単位で四捨五入した額は，［　f　］億円になる。

eに関する解答群

ア　50%を上回る

イ　70%以上である

ウ　期待どおりの効果が実現できない確率よりも低い

エ　期待の40%以下の効果しか実現しない確率よりも低い

オ　期待の50%以下の効果しか実現しない確率の2倍以上である

fに関する解答群

ア　300　　イ　312　　ウ　404　　エ　408　　オ　412

次の問８は必須問題です。必ず解答してください。

問８　次のプログラムの説明及びプログラムを読んで，設問１～３に答えよ。

〔プログラムの説明〕

関数 BMMatch は，Boyer-Moore-Horspool 法（以下，BM 法という）を用いて，文字列検索を行うプログラムである。BM 法は，検索文字列の末尾の文字から先頭に向かって，検索対象の文字列（以下，対象文字列）と 1 文字ずつ順に比較していくことで照合を行う。比較した文字が一致せず，照合が失敗した際には，検索文字列中の文字の情報を利用して，次に照合を開始する対象文字列の位置を決定する。このようにして明らかに不一致となる照合を省き，高速に検索できる特徴がある。

(1) 対象文字列を Text[]，検索文字列を Pat[] とする。ここで，配列の添字は 1 から始まり，文字列 Text[] の i 番目の文字は Text[i] と表記される。Pat[] についても同様に i 番目の文字は Pat[i] と表記される。また，対象文字列と検索文字列は，英大文字から構成される。

例えば，対象文字列 Text[] が“ACBBMACABABC”，検索文字列 Pat[] が“ACAB”の場合の例を図１に示す。

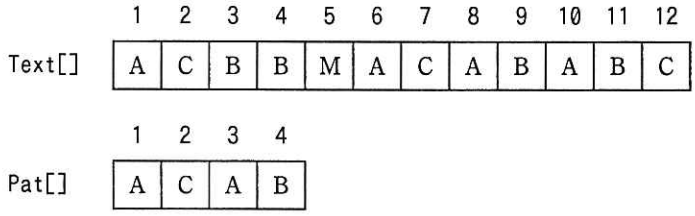


図１　対象文字列と検索文字列の格納例

(2) 関数 BMMatch では，照合が失敗すると，次に照合を開始する位置まで検索文字列を移動するが，その移動量を格納した要素数 26 の配列 Skip[] をあらかじめ作成しておく。Skip[1] に文字“A”に対応する移動量を，Skip[2] に文字“B”に対応する移動量を格納する。このように，Skip[1] ～ Skip[26] に文字“A”～“Z”

に対応する移動量を格納する。ここで，検索文字列の長さを PatLen とすると，移動量は次のようになる。

① 検索文字列の末尾の文字 Pat[PatLen] にだけ現れる文字と，検索文字列に現れない文字に対応する移動量は，PatLen である。

② 検索文字列の Pat[1] から Pat[PatLen − 1] に現れる文字に対応する移動量は，その文字が，検索文字列の末尾から何文字目に現れるかを数えた文字数から 1 を引いた値とする。ただし，複数回現れる場合は，最も末尾に近い文字に対応する移動量とする。

(3) 図 1 で示した Pat[] の例の場合，次の①～④に示すように，Skip[] は図 2 のとおりになる。

① 文字“A”は検索文字列の末尾から 2 文字目（Pat[3]）と 4 文字目（Pat[1]）に現れるので，末尾に近い Pat[3]に対応する移動量の 1（＝2－1）となる。

② 文字“B”は検索文字列の末尾の文字にだけ現れるので，移動量は PatLen（＝4）となる。

③ 文字“C”は検索文字列の末尾から 3 文字目（Pat[2]）に現れるので，移動量は 2（＝3－1）となる。

④ “A”，“B”及び“C”以外の文字については検索文字列に現れないので，移動量は PatLen（＝4）となる。

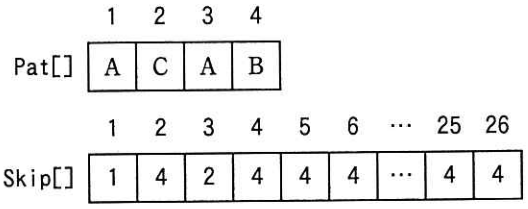


図 2　図 1 の格納例における Skip[] の値

(4) 図1の例で照合する場合の手順は，次の①～⑨となり，その流れを図3に示す。
この例では，PatLen＝4なので，検索文字列の末尾の文字はPat[4]である。

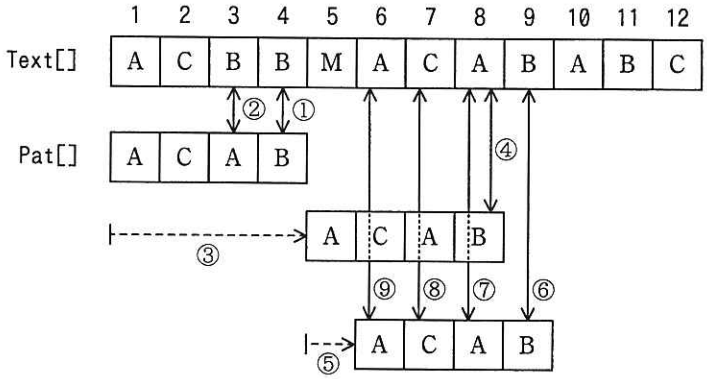


図3　図1の場合の照合手順

① Text[4]とPat[4]を比較する。Text[4]とPat[4]は同じ文字“B”である。

② Text[3]とPat[3]を比較する。Text[3]の“B”とPat[3]の“A”は異なる文字である。

③ ①で検索文字列の末尾の文字Pat[4]と比較したText[4]を基準に，Text[4]の文字“B”に対応する移動量であるSkip[2]の値4だけPat[]を右側に移動し，Text[8]とPat[4]の比較に移る。

④ Text[8]とPat[4]を比較する。Text[8]の“A”とPat[4]の“B”は異なる文字である。

⑤ ④で検索文字列の末尾の文字Pat[4]と比較したText[8]を基準に，Text[8]の文字“A”に対応する移動量であるSkip[1]の値1だけPat[]を右側に移動し，Text[9]とPat[4]の比較に移る。

⑥ Text[9]とPat[4]を比較する。Text[9]とPat[4]は同じ文字“B”である。

⑦ Text[8]とPat[3]を比較する。Text[8]とPat[3]は同じ文字“A”である。

⑧ Text[7]とPat[2]を比較する。Text[7]とPat[2]は同じ文字“C”である。

⑨ Text[6]とPat[1]を比較する。Text[6]とPat[1]は同じ文字“A”である。

⑥～⑨の比較で，対象文字列 Text[]の連続した一部分が検索文字列 Pat[]に完全に一致したので，検索は終了する。

〔関数 BMMatch の引数と返却値〕

関数 BMMatch の引数と返却値の仕様は，次のとおりである。

引数名／返却値	データ型	意味
Text[]	文字型	対象文字列が格納されている１次元配列
TextLen	整数型	対象文字列の長さ（１以上）
Pat[]	文字型	検索文字列が格納されている１次元配列
PatLen	整数型	検索文字列の長さ（１以上）
返却値	整数型	対象文字列中に検索文字列が見つかった場合は，１以上の値を返す。 検索文字列が見つからなかった場合は，−1 を返す。

関数 BMMatch では，次の関数 Index を使用する。

〔関数 Index の仕様〕

引数にアルファベット順で n 番目の英大文字を与えると，整数 n(1≦n≦26)を返却値とする。

〔プログラム〕

```
○整数型関数: BMMatch(文字型: Text[], 整数型: TextLen,
                      文字型: Pat[], 整数型: PatLen)
○整数型: Skip[26], PText, PPat, PLast, I

■ I: 1, I ≦ 26, 1
  • Skip[I] ← [ a ]
■
■ I: 1, I ≦ PatLen − 1, 1 ←――――――――――― γ
  • Skip[Index(Pat[I])] ← [ b ]
■
• PLast ← PatLen
■ PLast ≦ TextLen
  • PText ← PLast ←――――――――――――――― α
  • PPat ← PatLen
  ■ Text[PText] = Pat[PPat]
    ▲ PPat = 1 ←――――――――――――――― β
    ▼ • return (PText)
    • PText ← PText − 1
    • PPat ← PPat − 1
  ■
  • PLast ← PLast + Skip[Index(Text[PLast])]
■
• return (−1)
```

E 後

設問1　プログラム中の［　　］に入れる正しい答えを，解答群の中から選べ。

a，bに関する解答群

ア　0　　イ　1　　ウ　I － PatLen

エ　PatLen　　オ　PatLen － 1　　カ　PatLen － I

設問2　次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。

図4のように，Text[]に“ABCXBBACABACADEC”，TextLenに16，Pat[]に“ABAC”，PatLenに4を格納し，BMMatch (Text[], TextLen, Pat[], PatLen)を呼び出した。プログラムが終了するまでにαは［　c　］回実行され，βは［　d　］回実行される。またこの場合，関数BMMatchの返却値は［　e　］である。

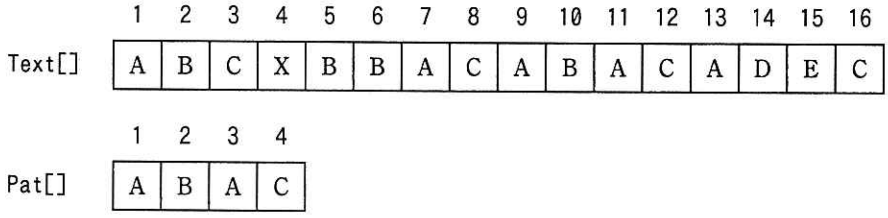


図4　対象文字列と検索文字列

c～eに関する解答群

ア　3　　イ　4　　ウ　5　　エ　6　　オ　7

カ　8　　キ　9　　ク　10　　ケ　11　　コ　12

設問３　次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。ここで，プログラム中の［ a ］と［ b ］には正しい答えが入っているものとする。

関数 BMMatch 中のγの処理を

I: PatLen − 1, I ≧ 1, −1

に変更した場合，関数 BMMatch は［ f ］。

fに関する解答群

ア　対象文字列中に，検索文字列が含まれていないのに，1 以上の値を返す場合がある

イ　対象文字列中に，検索文字列が含まれているのに，−1 を返す場合がある

ウ　正しい値を返す

次の問9から問13までの5問については，この中から1問を選択し，選択した問題については，答案用紙の選択欄の(選)をマークして解答してください。

なお，2問以上マークした場合には，はじめの1問について採点します。

問9　次のCプログラムの説明及びプログラムを読んで，設問1，2に答えよ。

C

〔プログラムの説明〕

入退室が制限されているエリア（以下，制限エリアという）の入退室の状況を記録したファイルを読み込んで，入退室状況を印字するプログラムである。

X社では，ICカードを用いた入退室システムを導入して，制限エリアの入退室を管理している。図1に，X社の制限エリアのレイアウトを示す。ドアは5か所（ドア番号 11，12，21，22，31）あり，各ドアの内外にカードリーダが設置されている。入室時と退室時には，カードリーダにICカードをかざしてドアを開錠する。各ドアのカードリーダにICカードをかざす都度，その入退室の状況を記録した1行の入退室レコードが入退室ログに書き込まれる。

制限エリアには，機密度に応じたレベルが設定されている。レベルは，外部からその部屋へ入るまでに開錠して通過する必要があるドアの数で表す。ドア番号は，10の位がレベルを表す。入退室できるレベルは，ICカードごとに設定されている。

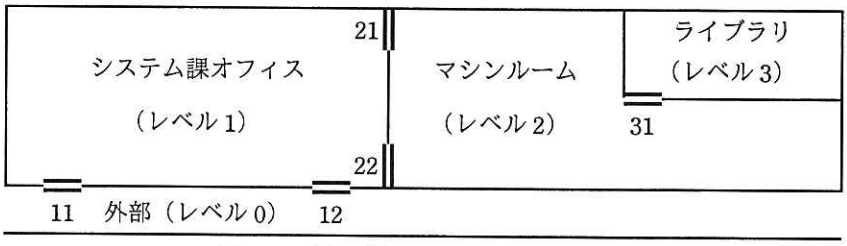


図1　X社の制限エリアのレイアウト

(1)　入退室レコードは33桁の固定長の文字列で，次に示す項目から成る。

項目名	カードID	日付			時刻			ドア番号	入退	可否	名前	改行
桁数	4桁	4桁	2桁	2桁	2桁	2桁	2桁	2桁	1桁	1桁	10桁	1桁
データ例	S001	2015	10	18	09	00	00	11	I	A	Sys Op Lib	\n

① カードIDは，ICカードを識別するIDであり，英数字から成る。

② 日付と時刻は，カードリーダにICカードをかざした日付と時刻である。

③ 入退は，“I”（入室）又は“O”（退室）である。

④ 可否は，“A”（開錠）又は“R”（拒否）である。

⑤ 名前は，ICカードの使用者を識別する名前であり，英数字と空白から成る。

⑥ レコードの終端には，改行文字が付いている。

(2) 入退室ログから印字したい日付・時間帯の入退室レコードを抽出し，カード ID，日付及び時刻（先頭の18桁）で昇順に整列したファイルを Access.Log とする。図2に，Access.Log のレコードの例を示す。

```
S001,20151018,090000,11,I,A,Sys Op Lib
S001,20151018,091004,21,I,A,Sys Op Lib
S001,20151018,092008,31,I,A,Sys Op Lib
S001,20151018,093012,31,O,A,Sys Op Lib
S001,20151018,094016,21,O,A,Sys Op Lib
S001,20151018,095020,11,O,A,Sys Op Lib
V001,20151018,110048,12,I,A,Visitor 01
V001,20151018,111052,22,I,R,Visitor 01
V001,20151018,112056,12,O,A,Visitor 01
```

注記 破線は項目の区切りを示す。
改行文字は表示していない。

図2 Access.Log のレコードの例

(3) Access.Log から読み込んだ1レコードごとに，カード ID，名前，日時，ドア番号，入退，可否から成る1行を，図3に示す様式で印字する。日時以降の項目は，ICカードをかざしたドアのレベルに応じて19，39，59桁目のいずれかから印字する。

```
1 …                19 …                39 …                59 …
カード ID 名前      レベル 1            レベル 2            レベル 3

S001  Sys Op Lib  10-18 09:00 11 IN
                                       10-18 09:10 21 IN
                                                           10-18 09:20 31 IN
                                                           10-18 09:30 31 OUT
                                       10-18 09:40 21 OUT
                  10-18 09:50 11 OUT

V001  Visitor 01  10-18 11:00 12 IN
                                       10-18 11:10 22 (R)IN
                  10-18 11:20 12 OUT
```

注記 見出し行は，あらかじめ印字されているものとする。

図3 図2のレコードの例を用いた印字結果

① カード ID 及び名前は，同一カード ID が続くとき，先頭の行だけに印字する。

② 日時は，“月 - 日 時 : 分”の形式で印字する。

③ 入退は，“I”なら“IN”と，“O”なら“OUT”と印字する。

④ 可否は，“A”なら何も印字せず，“R”なら“IN”又は“OUT”の直前に“(R)”を印字する。

(4) ライブラリ関数 strcmp(s1, s2) は，文字列 s1 と s2 を比較し，s1 < s2 のとき負の値を，s1 = s2 のとき 0 を，s1 > s2 のとき正の値を，それぞれ返す。また，ライブラリ関数 strcpy(s1, s2)は，文字列 s2 を s1 に複写する。

〔プログラム〕

```
      #include <stdio.h>
      #include <string.h>

      FILE *logFile;
      int  logEOF = 0;
      char cardID[5] = "----", date[9] = "--------", time[7] = "------",
           door[3] = "--", dir[2] = "-", act[2] = "-",
           name[11] = "----------";
① →   char lastID[5] = "----";

      void getRecord();
② →   void putRecord();

      void main() {
         logFile = fopen("Access.Log", "r");
         getRecord();
         while (logEOF != EOF) {
③ →         putRecord();
④ →         getRecord();
⑤ →      }
         fclose(logFile);
      }

      void getRecord() {
         if (fscanf(logFile, "%4c%8c%6c%2c%1c%1c%10c\n",
                    cardID, date, time, door, dir, act, name) == EOF)
            [    a    ] ;
         /* ファイルの終わりに達したとき fscanf は EOF（負の整数定数）を返す */
      }
```

```
    void putRecord() {
       int  putSpace;

       if (strcmp(cardID, lastID) == 0)
          printf("%18s", " ");
       else {
          printf("\n%4s  %10s  ", cardID, name);
⑥ →       [ b ] ;
⑦ →    }
       putSpace = [ c ] ;
       while (0 < putSpace--) {
          printf("%20s", " ");
⑧ →    }
       printf("%.2s-%.2s %.2s:%.2s %2s ",
             date+4, date+6, time, time+2, door);
       if (strcmp(act, "R") == 0)  printf("%s",   "(R)");
       if (strcmp(dir, "I") == 0)  printf("%s\n", "IN" );
       else                        printf("%s\n", "OUT");
    }
```

設問1　プログラム中の [　　] に入れる正しい答えを，解答群の中から選べ。

a，bに関する解答群

ア	cardID = lastID	イ	lastID = cardID
ウ	logEOF = 0	エ	logEOF = EOF
オ	strcpy(cardID, "　　")	カ	strcpy(cardID, lastID)
キ	strcpy(lastID, "　　")	ク	strcpy(lastID, cardID)

cに関する解答群

ア	door[0] - '0' - 1	イ	door[0] - '0'
ウ	door[0] - '0' + 1	エ	door[0] - '3'

設問2　次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。

プログラムを変更して，入退室順序の整合性を検査する処理を追加する。

各入退室者は，最初は外部（レベル 0）にいて，最後は外部に出るものとする。正しく入退室していれば，制限エリアのレベルは0から始まり，ドアを通過する都度，レベルが1ずつ変化し，最後は0に戻るはずである。しかし，抽出した日付・時間帯よりも前（後）に入室（退室）している場合や，IC カードをかざさずに人の後についてドアを通過した場合などには，順序の不整合が起きる。

なお，カードリーダに IC カードをかざしてドアが開錠した場合は，必ず入退室するものとする。

(1)　順序の不整合が起きた場合は，本来入退室の記録があるべき行のレベル 1 の印字位置に“***** Level *x*-->*y*”と印字する。これは，入退室者がこの前後にレベル*x*から*y*へ移動したはずであるが，その記録がないことを意味する。

(2)　図 4 に，不整合がある入力レコードの例を示す。図 4 のレコードを用いて変更後のプログラムを実行すると，印字結果は，図5のようになる。

```
T001,20151018,100024,11,I,A,Test Use 1
T001,20151018,102028,31,I,A,Test Use 1
T001,20151018,104032,21,O,A,Test Use 1
```

注記　破線は項目の区切りを示す。
　　改行文字は表示していない。

図4　不整合がある入力レコードの例

```
1 …                19 …                39 …                59 …
カード ID 名前      レベル 1            レベル 2            レベル 3

T001  Test Use 1  10-18 10:00 11 IN
                  ***** Level 1-->2
                                                          10-18 10:20 31 IN
                  ***** Level 3-->2
                                      10-18 10:40 21 OUT
                  ***** Level 1-->0
```

注記　見出し行は，あらかじめ印字されているものとする。

図5　不整合がある入力レコードの例（図4）のときの印字結果

プログラムを，次のように変更する。

(1) プログラムに，次の文を追加する。

追加位置	追加する文
行①の直後	int level = 0;
行②の直後	void checkLevel(); void clearLevel();
[d] の直後	checkLevel();
[e] の直後	clearLevel();

注記 level は，入退室者が現在いる制限エリアのレベルを表す変数である。

(2) プログラムの末尾に，次の二つの関数を追加する。

```
void checkLevel() {
   int  afterLevel, beforeLevel, doorLevel;

   doorLevel = door[0] - '0';
   beforeLevel = doorLevel;
   afterLevel  = doorLevel;
   if (strcmp(dir, "I") == 0)
      beforeLevel = doorLevel - 1;
   else
      afterLevel = doorLevel - 1;
   if (strcmp(act, "R") == 0)
      afterLevel = beforeLevel;
   if (level != beforeLevel)
      printf("***** Level %d-->%d\n%18s",
             [ f ], [ g ], " ");
   level = afterLevel;
}

void clearLevel() {
   if (logEOF == EOF || (strcmp(cardID, lastID) != 0))
      if (level > 0) {
         printf("%18s***** Level %d-->0\n", " ", level);
         level = 0;
      }
}
```

d, eに関する解答群

ア　行③	イ　行④	ウ　行⑤
エ　行⑥	オ　行⑦	カ　行⑧

f, gに関する解答群

ア　afterLevel	イ　beforeLevel
ウ　doorLevel	エ　level

C

問10　次のCOBOLプログラムの説明及びプログラムを読んで，設問1，2に答えよ。

〔プログラムの説明〕

D社では，ある製品の満足度を調べるために，アンケート調査を実施し，得られた1,000件の回答をアンケート結果ファイルに登録した。このプログラムは，アンケート結果ファイルを読み込み，集計結果をグラフで表示する。

(1)　アンケート結果ファイルは，図1に示すレコード様式の順ファイルである。

回答ID 4桁	価格満足度 1桁	機能満足度 1桁	デザイン満足度 1桁

図1　アンケート結果ファイルのレコード様式

①　回答IDは，一意に割り振られた0001～1000の番号である。

②　価格満足度，機能満足度及びデザイン満足度は，それぞれの満足度の高から低を5～1の5段階で評価したものである。

(2)　集計した満足度の分布をヒストグラムで表示する。図2に集計結果の表示例を示す。

①　価格（Price），機能（Function）及びデザイン（Design）の各項目のヒストグラムを並べて表示する。

②　各ヒストグラムは，項目名の行と満足度5～1の行の6行から成る。

③　各満足度の行は，満足度の値，その満足度の選択数，選択数を長さで表す横棒の順に表示する。

④　選択数は，値が0の場合以外は先行する数字0を除いて，値が0の場合は1桁の数字0を，右寄せで表示する。

⑤　横棒は，選択数20ごとに記号“*”を一つ用いて表示する。選択数が20に満たない端数は切り捨てる。

```
Price
5   69 ***
4  255 ***********
3  378 ******************
2  229 **********
1   69 ***
Function
5   59 **
4  238 **********
3  381 ******************
2  258 ***********
1   64 ***
Design
5   81 ****
4  244 ***********
3  362 ******************
2  245 ***********
1   68 ***
```

図2　集計結果の表示例

〔プログラム〕

（行番号）

```
 1  DATA DIVISION.
 2  FILE SECTION.
 3  FD ANS-FILE.
 4  01 ANS-REC.
 5     02 ANS-ID       PIC 9(4).
 6     02 ANS-SATIS    OCCURS 3.
 7        03 ANS-VAL   PIC 9(1).
 8  WORKING-STORAGE SECTION.
 9  77 S-PRICE      PIC 9(1) VALUE 1.
10  77 S-FUNCTION   PIC 9(1) VALUE 2.
11  77 S-DESIGN     PIC 9(1) VALUE 3.
12  77 ANS-FLAG     PIC 9(1) VALUE 0.
13     88 ANS-EOF   VALUE 1.
14  01 HIST-REC.
15     02 HIST-VAL    PIC 9(1).
16     02            PIC X(1)  VALUE SPACE.
17     02 HIST-NUM    PIC [   a   ].
18     02            PIC X(1)  VALUE SPACE.
19     02 HIST-PLOT   PIC X(50).
20  01 CNT-REC.
21     02 CNT-SATIS       OCCURS 3.
22        03 CNT-VAL      OCCURS 5.
23           04 CNT-NUM   PIC 9(4) VALUE 0.
```

```
77 PLOT-NUM  PIC 9(2).
77 I         PIC 9(1).
77 J         PIC 9(1).
PROCEDURE DIVISION.
MAIN-PROC.
    OPEN INPUT ANS-FILE.
    PERFORM READ-PROC.
    PERFORM DISP-PROC.
    CLOSE ANS-FILE.
    STOP RUN.
READ-PROC.
    PERFORM UNTIL ANS-EOF
       READ ANS-FILE
          AT END
             SET ANS-EOF TO TRUE
          NOT AT END
             PERFORM VARYING I FROM 1 BY 1 UNTIL I > 3
                MOVE ANS-VAL(I) TO J
                ADD 1 TO CNT-NUM(I, J)
             END-PERFORM
       END-READ
    END-PERFORM.
DISP-PROC.
    PERFORM VARYING I FROM 1 BY 1 UNTIL I > 3
       EVALUATE I
          WHEN S-PRICE
             DISPLAY "Price"
          WHEN S-FUNCTION
             DISPLAY "Function"
          WHEN S-DESIGN
             DISPLAY "Design"
       END-EVALUATE
       PERFORM VARYING J FROM 5 BY -1 UNTIL J = 0
          MOVE J TO HIST-VAL
          MOVE [ b ] TO HIST-NUM
          DIVIDE [ b ] BY 20 GIVING PLOT-NUM
          MOVE SPACE TO HIST-PLOT
          IF PLOT-NUM > 0 THEN
             MOVE ALL "*" TO HIST-PLOT( [ c ] )
          END-IF
          DISPLAY [ d ]
       END-PERFORM
    END-PERFORM.
```

設問1　プログラム中の［　　］に入れる正しい答えを，解答群の中から選べ。

aに関する解答群

ア　9(4)　　イ　B(3)9　　ウ　Z(3)9　　エ　Z(4)

bに関する解答群

ア　CNT-NUM(6 - I, J)　　イ　CNT-NUM(I, 5 - J)

ウ　CNT-NUM(I, 6 - J)　　エ　CNT-NUM(I, J)

cに関する解答群

ア　1:PLOT-NUM　　イ　1:PLOT-NUM + 1

ウ　PLOT-NUM　　エ　PLOT-NUM + 1

dに関する解答群

ア　HIST-NUM　　イ　HIST-PLOT　　ウ　HIST-REC　　エ　HIST-VAL

設問2　各回答の3項目の満足度を加算した値（以下，総合満足度という）を求め，総合満足度ごとの回答IDの一覧を追加して表示するよう，プログラムを変更する。追加した部分の表示例を図3に示す。表1中の［　　］に入れる正しい答えを，解答群の中から選べ。

① 総合満足度の昇順に，その値と該当する全ての回答IDを表示する。

② 回答IDは，総合満足度ごとに昇順に表示する。

③ 回答IDは，10件ごとに改行して表示する。

④ 回答が1件もない総合満足度の値に対しては，表示を省略する。

```
Total
 3 0774 0850
 5 0067 0084 0220 0230 0232 0269 0285 0297 0651 0682
   0709 0712 0748 0762 0782 0797 0810 0823 0853 0963
 6 0021 0044 0068 0106 0107 0124 0134 0139 0183 0193
   0194 0203 0208 0239 0241 0259 0268 0306 0334 0379
   0390 0409 0443 0462 0528 0529 0533 0539 0540 0574
   0594 0596 0601 0610 0647 0656 0661 0666 0680 0688
   0694 0726 0744 0775 0807 0812 0821 0888 0894 0896
   0899 0904 0912 0913 0938 0939 0945 0973 0974 0975
   0986
 (中略)
14 0839 0947
```

図3　総合満足度ごとの回答 ID 一覧の表示例

表1　プログラムの変更内容

処置	変更内容
行番号 7 と 8 の間に追加	SD SRT-FILE. 01 SRT-REC. 　02 SRT-SUM　PIC 9(2). 　02 SRT-ID　PIC 9(4).
行番号 26 と 27 の間に追加	77 SRT-FLAG　PIC 9(1) VALUE 0. 　88 SRT-EOF　VALUE 1. 01 T-SATIS-REC. 　02 T-SATIS-SUM　PIC Z(2). 　02 T-SATIS-IDS　OCCURS 10. 　　03　PIC X(1)　VALUE SPACE. 　　03 T-SATIS-ID　PIC X(4). 77 LAST-SUM　PIC 9(2). 77 POS　PIC 9(2).
行番号 30，31 を変更	SORT SRT-FILE ASCENDING KEY [　e　] 　INPUT　PROCEDURE IS READ-PROC 　OUTPUT PROCEDURE IS DISP-PROC.
行番号 40 ～ 43 を変更	MOVE 0 TO SRT-SUM PERFORM VARYING I FROM 1 BY 1 UNTIL I > 3 　MOVE ANS-VAL(I) TO J 　ADD 1 TO CNT-NUM(I, J) 　ADD ANS-VAL(I) TO SRT-SUM END-PERFORM MOVE ANS-ID TO SRT-ID RELEASE SRT-REC

行番号 66 の後ろに追加	DISPLAY "Total".
	MOVE 1 TO LAST-SUM POS.
	PERFORM UNTIL SRT-EOF
	　RETURN SRT-FILE
	　　AT END
	　　SET SRT-EOF TO TRUE
	　　NOT AT END
	　　IF [f] THEN
	　　IF POS > 1 THEN
	　　DISPLAY T-SATIS-REC
	　　END-IF
	　　MOVE SPACE TO T-SATIS-REC
	　　IF [g] THEN
	　　MOVE SRT-SUM TO T-SATIS-SUM LAST-SUM
	　　END-IF
	　　MOVE 1 TO POS
	　　END-IF
	　　MOVE SRT-ID TO T-SATIS-ID(POS)
	　　ADD 1 TO POS
	　END-RETURN
	END-PERFORM.
	DISPLAY T-SATIS-REC.

eに関する解答群

ア　SRT-ID　　イ　SRT-REC　　ウ　SRT-SUM

f，gに関する解答群

ア　POS > 1

イ　POS > 10

ウ　SRT-SUM NOT = LAST-SUM

エ　SRT-SUM NOT = LAST-SUM AND POS > 1

オ　SRT-SUM NOT = LAST-SUM AND POS > 10

カ　SRT-SUM NOT = LAST-SUM OR POS > 1

キ　SRT-SUM NOT = LAST-SUM OR POS > 10

問11　次のJavaプログラムの説明及びプログラムを読んで，設問に答えよ。
（Javaプログラムで使用するAPIの説明は，この冊子の末尾を参照してください。）

〔プログラムの説明〕

固定バイト長のブロック（以下，ブロックという）を単位としてデータを保管している装置（以下，ブロックデバイスという）に対し，ブロックのデータ（以下，ブロックデータという）へのアクセスを管理するプログラムである。図1に，プログラム構成の概要を示す。図中の四角はプログラムの構成要素を，矢印はブロックデータの流れを示す。ここで，このプログラムでは，ブロックデバイスからの読込みだけを考える。

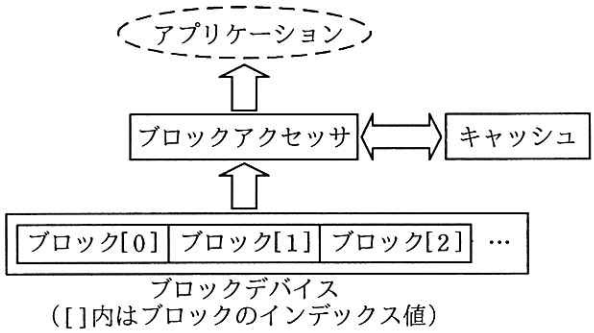


図1　プログラム構成の概要

ブロックデバイス内の各ブロックは，インデックス値で指定する。最初のブロックのインデックス値は，0である。

ブロックアクセッサは，アプリケーションから要求されたブロックデータをブロックデバイスから取得して，アプリケーションに渡す役割をする。ブロックアクセッサはキャッシュを使用し，ブロックデバイスから取得したブロックデータをキャッシュする。アプリケーションから要求されたブロックデータがキャッシュされていれば，キャッシュから取得したブロックデータをアプリケーションに渡す。

キャッシュは，FIFO（First In，First Out）とLRU（Least Recently Used）の2種類の管理方針に基づく実装が用意されていて，アプリケーションはどの実装を使用するかを指定する。

(1) クラス BlockAccessor は，ブロックアクセッサを表す。

① コンストラクタは，ブロックデバイスをオープンし，引数で指定されたキャッシュの管理方針に基づくキャッシュを生成する。

② メソッド readBlock は，引数で指定されたインデックス値のブロックデータを返す。ブロックデータがキャッシュにあれば，それを返す。なければ，ブロックデバイスから取得し，キャッシュした後，そのブロックデータを返す。

(2) クラス BlockDevice は，読取り専用のブロックデバイスを表す。実装は，バイト配列の配列であるフィールド blocks を使用し，ブロックデバイスを仮想的に表す。ただし，ブロックデータの値は全て 0 である。

① 静的メソッド open は，ブロックデバイスのインスタンスを返す。

② メソッド getBlockSize は，ブロックのサイズをバイト数で返す。

③ メソッド readBlock は，引数で指定されたインデックス値のブロックデータを引数で指定されたバッファに読み込む。

(3) 抽象クラス Cache は，ブロックアクセッサからキャッシュ機能を使用するためのクラスを表す。

① キャッシュの管理方針（FIFO 及び LRU）を表す入れ子列挙 Cache.Policy を定義する。

② 静的メソッド createCache は，引数で指定されたキャッシュの管理方針と一致する実装クラスのインスタンスを生成して返す。

③ 抽象メソッド getCachedBlockData は，引数で指定されたインデックス値のブロックデータがキャッシュされていれば，それを返す。キャッシュされていなければ，null を返す。

④ 抽象メソッド cacheBlockData は，引数で指定されたインデックス値及びその値に対応するブロックデータをキャッシュする。キャッシュできるブロックデータ数が上限に達している場合は，管理方針に従ってキャッシュされているブロックデータを削除してから，指定されたブロックデータをキャッシュする。

(4) 抽象クラス ListBasedCache は，リスト構造を使用して抽象クラス Cache の一部を実装する。キャッシュできるブロックデータ数の上限値は，フィールド CACHE_SIZE で表す。

① メソッド getCachedBlockData 及び cacheBlockData は，抽象クラス Cache で定

義されている同名のメソッドを実装する。

② 抽象メソッド hit は，要求されたブロックデータがキャッシュ中に見つかったときに呼び出される。引数は，見つかったブロックデータとそのインデックス値の対（以下，キャッシュエントリという）である。

(5) 入れ子クラス ListBasedCache.Entry は，キャッシュに格納するキャッシュエントリを表す。

① コンストラクタは，引数で指定されたインデックス値及びブロックデータをもつキャッシュエントリを生成する。

② メソッド getIndex 及び getBlockData は，それぞれインデックス値及びブロックデータを返す。

(6) 入れ子クラス ListBasedCache.Fifo は，キャッシュエントリの管理を FIFO で行う。

① メソッド hit は，無操作である。

(7) 入れ子クラス ListBasedCache.Lru は，キャッシュエントリの管理を LRU で行う。

① メソッド hit は，指定されたキャッシュエントリをリストの先頭に移動する。

なお，上記コンストラクタやメソッドを呼び出すときの引数に誤りはないものとする。

〔プログラム 1〕

```
public class BlockAccessor {
   private final Cache cache;
   private final BlockDevice device;

   public BlockAccessor(Cache.Policy policy) {
      device = BlockDevice.open();
      cache = Cache.createCache(policy);
   }

   public byte[] readBlock(int index) {
      byte[] blockData = cache.getCachedBlockData(index);
      if (blockData == null) {
         blockData = new byte[device.getBlockSize()];
         device.readBlock(index, blockData);
         cache.cacheBlockData(index, blockData);
      }
      return blockData.clone();
```

```
    }
}
```

〔プログラム2〕

```
class BlockDevice {
    private final byte[][] blocks = new byte[100][512];

    static BlockDevice open() {
        return new BlockDevice();
    }

    int getBlockSize() {
        return [ a ].length;
    }

    void readBlock(int index, byte[] buffer) {
        byte[] block = blocks[index];
        System.arraycopy(block, 0, buffer, 0, block.length);
    }
}
```

〔プログラム3〕

```
public abstract class Cache {
    public enum Policy {
        FIFO, LRU;
    }

    static [ b ] createCache(Policy policy) {
        switch (policy) {
        case FIFO:
            return new ListBasedCache.Fifo();
        case LRU:
            return new ListBasedCache.Lru();
        }
        throw new UnsupportedOperationException();
    }

    abstract byte[] getCachedBlockData(int index);
    abstract void cacheBlockData(int index, byte[] blockData);
}
```

〔プログラム4〕

```
import java.util.ArrayList;
import java.util.List;

abstract class ListBasedCache extends Cache {
    final List<Entry> entries = new ArrayList<Entry>();
```

```
    private static final int CACHE_SIZE = 20;

    byte[] getCachedBlockData(int index) {
        for (Entry entry : entries) {
            if (entry.getIndex() [ c ] index) {
                hit(entry);
                return entry.getBlockData();
            }
        }
        return null;
    }

    void cacheBlockData(int index, byte[] blockData) {
        if ( [ d ] ) {
            entries.remove( [ e ] );
        }
        entries.add(0, new Entry(index, blockData));
    }

    abstract void hit(Entry entry);

    private static class Entry {
        private final int index;
        private final byte[] blockData;

        private Entry(int index, byte[] blockData) {
            this.index = index;
            this.blockData = blockData;
        }

        int getIndex() { return index; }
        byte[] getBlockData() { return blockData; }
    }

    static class Fifo extends [ f ] {
        void hit(Entry entry) { }
    }

    static class Lru extends [ f ] {
        void hit(Entry entry) {
            entries.remove( [ g ] );
            entries.add(0, entry);
        }
    }
}
```

設問　プログラム中の［　　］に入れる正しい答えを，解答群の中から選べ。

aに関する解答群

ア　blocks	イ　blocks[]	ウ　blocks[0]
エ　blocks[][]	オ　blocks[0][]	カ　blocks[][0]

bに関する解答群

ア　Cache	イ　Cache<? extends Policy>	ウ　enum
エ　Policy	オ　Policy<? extends Cache>	カ　void

cに関する解答群

ア　!=	イ　<	ウ　<=
エ　==	オ　>	カ　>=

dに関する解答群

ア　!entries.isEmpty()	イ　entries.isEmpty()
ウ　entries.size() != CACHE_SIZE	エ　entries.size() != index
オ　entries.size() == CACHE_SIZE	カ　entries.size() == index

eに関する解答群

ア　CACHE_SIZE	イ　CACHE_SIZE - 1	ウ　CACHE_SIZE + 1
エ　index	オ　index - 1	カ　index + 1

fに関する解答群

ア　ArrayList	イ　ArrayList<Cache>	ウ　Cache
エ　List	オ　List<Cache>	カ　ListBasedCache

gに関する解答群

ア　0	イ　CACHE_SIZE - 1	ウ　entry
エ　entry.getIndex()	オ　entry.getIndex() - 1	カ　entry.getIndex() + 1

問12　次のアセンブラプログラムの説明及びプログラムを読んで，設問1～3に答えよ。

〔プログラム1の説明〕

副プログラム BITINS は，図1に示すように，ビット列A（長さ16ビット）をビット列B（長さ16ビット）の指定された位置に挿入し，ビット列C（長さ32ビット）を作成する。図1中の n（0≦n≦16）は，挿入位置（先頭からのビット数）を表す。

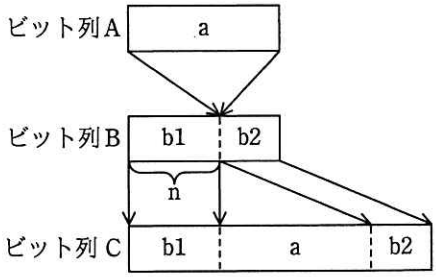


図1　ビット列の挿入

(1) GR0～GR3 には，それぞれ次の内容が設定されて，主プログラムから渡される。

GR0：ビット列A

GR1：ビット列Bが格納されている領域のアドレス

GR2：n

GR3：ビット列Cを格納する2語の領域の先頭アドレス

(2) 副プログラムから戻るとき，汎用レジスタ GR1～GR7 の内容は元に戻す。

〔プログラム 1〕

(行番号)

```
 1  BITINS  START
 2          RPUSH
 3          LD     GR6,=[ a ]
 4          SUBA   GR6,GR2          ; GR6 ← シフト数
 5          LD     GR1,0,GR1        ; GR1 ← ビット列 B
 6          LD     GR4,GR1          ; GR4 ← ビット列 B
 7          LD     GR5,GR0          ; GR5 ← ビット列 A
 8          SRL    GR1,0,GR6
 9          SLL    GR1,0,GR6        ; GR1 ← ビット列 B の左 n ビット
10          SLL    GR4,0,GR2
11          SRL    GR4,0,GR2        ; GR4 ← ビット列 B の右(16-n)ビット
12          [          b          ]
13          SLL    GR5,0,GR6
14          OR     GR1,GR0
15          OR     GR4,GR5
16          ST     GR1,0,GR3
17          ST     GR4,1,GR3
18          RPOP
19          RET
20          END
```

設問 1　プログラム 1 中の［　　］に入れる正しい答えを，解答群の中から選べ。

a に関する解答群

ア　0　　イ　8　　ウ　16　　エ　32

b に関する解答群

ア　SLL　GR0,0,GR2　　イ　SLL　GR0,0,GR6　　ウ　SRL　GR0,0,GR2

エ　SRL　GR0,0,GR6

設問2　次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。

主プログラムから渡されたnが10で，ビット列Aとビット列Bが図2のとき，行番号15のOR命令実行直前のGR4の内容は［　c　］であり，GR5の内容は［　d　］である。

ビット列A	0101 1110 1001 0110
ビット列B	1110 1111 0001 1000

図2　主プログラムから渡されたビット列Aとビット列B

解答群

ア　0000 0000 0001 0111　　イ　0000 0000 0001 1000

ウ　0000 0010 1001 0110　　エ　0000 0011 1011 1100

オ　0101 1100 0000 0000　　カ　0110 0000 0000 0000

キ　1010 0101 1000 0000　　ク　1110 1111 0000 0000

設問3　連続するN語を16×Nビットから成るビット列Yとする。図3に示すように，ビット列X（長さ16ビット）をビット列Yの指定された位置に挿入し，N+1語のビット列Zを作成する副プログラムBITINSLを，副プログラムBITINSを使って作成した。図3中のm（0≦m≦16×N）は，挿入位置（先頭からのビット数）を表す。プログラム2中の［　　］に入れる正しい答えを，解答群の中から選べ。

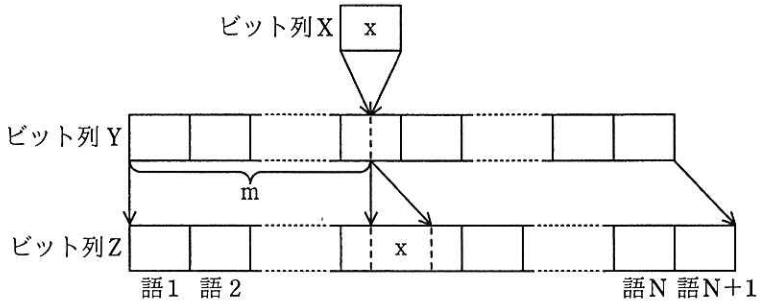


図3　N語から成るビット列への挿入

(1)　GR0～GR4には，それぞれ次の内容が設定されて，主プログラムから渡される。

GR0：ビット列X

GR1：ビット列Yが格納されている領域の先頭アドレス

GR2：m

GR3：ビット列Zを格納するN+1語の領域の先頭アドレス

GR4：N

(2)　副プログラムから戻るとき，汎用レジスタGR1～GR7の内容は元に戻す。

アセンブラ

〔プログラム2〕

```
BITINSL START
        RPUSH
LOOP1   CPA   GR2,=17
        [    e    ]
        CALL  COPY          ; ビット列Yの先頭から挿入位置の直前の
        SUBA  GR2,=16       ; 語までを結果の領域へコピー
        JUMP  LOOP1
INS     CALL  BITINS
        LAD   GR1,1,GR1
        LAD   GR3,2,GR3
        SUBA  GR4,=1
        [    f    ]
LOOP2   CALL  COPY          ; ビット列Yの残りの部分を
        JNZ   LOOP2         ; 結果の領域へコピー
FIN     RPOP
        RET
; ビット列Yの1語を結果の領域へコピー
COPY    LD    GR5,0,GR1
        ST    GR5,0,GR3
        LAD   GR1,1,GR1
        LAD   GR3,1,GR3
        SUBA  GR4,=1
        RET
        END
```

解答群

ア JMI FIN	イ JMI INS	ウ JPL FIN
エ JPL INS	オ JZE FIN	カ JZE INS

問13　次の表計算のワークシート及びマクロの説明を読んで，設問1，2に答えよ。

〔表計算の説明〕

ある PC 販売店では，セット値引きを導入することにした。セット値引きでは，指定された3種類の商品を同時に購入する場合，各商品を単品で購入する場合に比べて購入金額を安くする。セット値引きとなる商品の組合せ（以下，セットという）と値引きを適用していく順序は，商品の仕入状況と販売状況に基づいて適宜見直しを行う。セット値引きの導入に合わせて，これまで手書きであった購入伝票の作成について，表計算ソフトを使って行うことにした。

〔ワークシート：価格表〕

商品の単価を記載したワークシート“価格表”の例を，図1に示す。

	A	B	C	D
1	商品番号	商品名	商品コード	単価（円）
2	1	ミニタワーPC	PC001	39,800
3	2	スリム PC	PC002	42,800
4	3	コンパクト PC	PC003	37,800
5	4	ノート PC	PC004	42,800
6	5	無線 LAN（ルータ）	WLRTR	12,420
7	6	無線 LAN（子機）	WLCLT	2,480
8	7	プリンタ（USB 接続）	PRUSB	12,000
9	8	プリンタ（無線接続）	PRWLE	18,000
10	9	ネットワーク HDD（2TB）	ND200	25,000
11	10	ネットワーク HDD（4TB）	ND400	35,000
12	11	ネットワーク HDD（6TB）	ND600	45,000
13	12	キーボード（USB 接続）	KBUSB	5,000
14	13	キーボード（無線接続）	KBWLE	7,500
15	14	テンキーボード	TENKB	1,000
16	15	マウス（USB 接続）	MSUSB	2,000
⋮	⋮	⋮	⋮	⋮
51	50	ディスプレイケーブル	CBDSP	1,280

図1　ワークシート“価格表”の例

(1) 販売している商品は 50 品目あり，セル A2 ～ A51 には 1 から始まる連番で商品番号を入力する。

(2) セル B2 ～ B51 には商品名を，セル C2 ～ C51 には商品コードを，セル D2 ～ D51 には単価を入力する。商品名と商品コードは，それぞれ重複するものはない。

〔ワークシート：セット値引き表〕

セットと値引き額を記載したワークシート“セット値引き表”の例を，図 2 に示す。

	A	B	C	D	E	F	G	H	I	J	K	L	M
1			商品 1		商品 2		商品 3						
2	セット番号	セット名	商品コード	数量	商品コード	数量	商品コード	数量	値引き額(円)		商品 1 の商品番号	商品 2 の商品番号	商品 3 の商品番号
3	1	ミニタワーセット 1	PC001	1	KBUSB	1	MSUSB	1	−5,000		1	12	15
4	2	スリムセット 1	PC002	1	KBUSB	1	MSUSB	1	−4,800		2	12	15
5	3	コンパクトセット 1	PC003	1	KBUSB	1	MSUSB	1	−4,500		3	12	15
6	4	ミニタワーセット 2	PC001	1	DSP24	1	PRUSB	1	−4,400		1	18	7
7	5	スリムセット 2	PC002	1	DSP24	1	PRUSB	1	−4,200		2	18	7
8	6	コンパクトセット 2	PC003	1	DSP24	1	PRUSB	1	−4,000		3	18	7
9	7	ノートセット	PC004	1	MSUSB	1	TENKB	1	−3,000		4	15	14
10	8	プリンタ無線セット	PRWLE	1	WLRTR	1	WLCLT	1	−2,500		8	5	6
11	9	HDD 無線セット	ND400	1	WLRTR	1	WLCLT	2	−2,200		10	5	6
12	10	周辺機器セット	HD100	1	PRUSB	1	DSP200	1	−2,000		22	7	17

図 2　ワークシート“セット値引き表”の例

(1) セットは 10 種類あり，値引きを適用していく順に入力する。

(2) セル A3 ～ A12 には，1 から始まる連番でセット番号を入力する。

(3) セル B3 ～ B12 には，セット名を入力する。

(4) セットは 3 種類の商品の組合せから成り，セル C3 ～ H12 には組み合わせる商品の商品コードと数量を入力する。商品コードは，ワークシート“価格表”に登録されているものを用いる。

(5) セル I3 ～ I12 には，セットの値引き額を入力する。

(6) 列 K ～ M は，セット値引き適用処理のための作業領域に用いる。

〔ワークシート：購入伝票〕

購入する商品の一覧（購入内容），セット値引き適用状況の一覧（セット値引き内容），合計金額などを表示するワークシート“購入伝票”の例を，図3に示す。

	A	B	C	D	E
1				合計金額（円）	249,710
2				合計セット値引き金額（円）	−15,500
3				購入金額（円）	234,210
4					
5	購入内容				
6	No.	商品名	数量	単価（円）	金額（円）
7	1	ミニタワーPC	2	39,800	79,600
8	2	ノート PC	1	42,800	42,800
9	3	ディスプレイ（24インチ）	1	38,650	38,650
10	4	キーボード（USB 接続）	2	5,000	10,000
11	5	マウス（USB 接続）	2	2,000	4,000
12	6	テンキーボード	1	1,000	1,000
13	7	無線 LAN（ルータ）	1	12,420	12,420
14	8	無線 LAN（子機）	2	2,480	4,960
15	9	プリンタ（無線接続）	1	18,000	18,000
16	10	ネットワーク HDD（4TB）	1	35,000	35,000
17	11	ディスプレイケーブル	1	1,280	1,280
18	12	マウス（USB 接続）	1	2,000	2,000
19	13				
⋮	⋮	⋮	⋮	⋮	⋮
26	20				
27					
28	セット値引き内容				
29	セット番号	セット名	数量	値引き額（円）	セット値引き金額（円）
30	1	ミニタワーセット 1	2	−5,000	−10,000
31	2	スリムセット 1		−4,800	
⋮	⋮	⋮	⋮	⋮	⋮
36	7	ノートセット	1	−3,000	−3,000
37	8	プリンタ無線セット	1	−2,500	−2,500
38	9	HDD 無線セット		−2,200	
39	10	周辺機器セット		−2,000	

図3　ワークシート“購入伝票”の例

(1) 1回に購入できる商品は20品目以下である。ワークシート“購入伝票”のセルB7～C26の各行に，商品名と数量を1件ずつ入力すると，各行の列Dと列Eにそれぞれ単価と金額（数量×単価）が，セルE1に金額の合計が表示される。ワークシート“価格表”に登録されていない商品名を入力することはない。また，同一の商品を2行以上入力することができる。

(2) マクロCalc_Discount_Priceを実行すると，セルC30～C39に，セット値引きを適用するセットについてだけ数量が設定され，対応する各行の列Eにセット値引き金額（数量×値引き額）が，セルE2にセット値引き金額の合計が表示される。

設問1 ワークシート“購入伝票”に関する次の説明中の［　　］に入れる正しい答えを，解答群の中から選べ。

(1) セルA7～A26には，1から始まる連番を入力する。

(2) セルE1には，購入する商品の合計金額を求めるための式を入力する。

(3) セルE2には，セット値引き金額の合計を求めるための式を入力する。

(4) セルE3には，購入金額を求めるための式を入力する。

(5) 商品ごとの単価を表示するために，セルD7に次の式を入力し，セルD8～D26に複写する。

IF(B7＝null, null, 垂直照合(B7, [a], 0))

(6) 商品ごとの金額を求めるために，セルE7に次の式を入力し，セルE8～E26に複写する。

IF(D7＝null, null, C7＊D7)

(7) セルB30～B39には，ワークシート“セット値引き表”のセット名を，同表と同じ順序で表示するための式を入力する。

(8) セルC30～C39には，マクロCalc_Discount_Priceを用いて，数量を設定する。

(9) ワークシート“セット値引き表”の値引き額を，同表と同じ順序で表示するために，セルD30に次の式を入力し，セルD31～D39に複写する。

[b]

(10) セット値引き金額を求めるために，セルE30に次の式を入力し，セルE31

表計算

～E39 に複写する。

IF(C30 = null, null, C30 * D30)

a に関する解答群

ア　価格表!A$2:D$51, 2　　イ　価格表!A$2:D$51, 3

ウ　価格表!A$2:D$51, 4　　エ　価格表!B$2:D$51, 2

オ　価格表!B$2:D$51, 3

b に関する解答群

ア　セット値引き表!A3　　イ　セット値引き表!A$3

ウ　セット値引き表!I3　　エ　セット値引き表!I$3

設問2　セット値引き適用処理に関する次の説明及びマクロ中の［　　］に入れる正しい答えを，解答群の中から選べ。

〔セット値引き適用処理の説明〕

“セット値引き表”に記載されているセットに対して，セット番号が小さいものから順にセット化して，値引きを適用していく。

(1)　セットを構成する商品の商品番号を表示するために，ワークシート“セット値引き表”のセル K3 に次の式を入力し，セル K4～K12 に複写する。

照合検索(C3, ［ c ］, ［ d ］)

同様の方法で，セル L3～L12 とセル M3～M12 にも，セットを構成する商品の商品番号を表示するための式を入力する。

(2)　ワークシート“購入伝票”にマクロ Calc_Discount_Price を格納する。

〔マクロの説明〕

マクロ Calc_Discount_Price は，次の手順で処理を行う。

(1)　購入する各商品の数量を求め，配列 Goods_Num に格納する。商品番号が i（i＝1, 2, …, 50）の商品の数量は，Goods_Num[i － 1]に格納される。

(2)　セット番号が 1, 2, …, 10 の順に，セット値引きの対象となる数量を求め

るために(3)～(5)の処理を繰り返す。

(3) 配列 Goods_Num について，セットを構成する 3 種類の商品の数量を調べ，セットにできる最大の数量 Set_Num を求める。

(4) 数量 Set_Num が 0 でない場合は，ワークシート“購入伝票”のセットの数量のセルに数量 Set_Num を格納し，セット値引きの対象とした 3 種類の商品の数量を，配列 Goods_Num から減らす。

(5) 数量 Set_Num が 0 の場合は，ワークシート“購入伝票”のセットの数量のセルに空値 null を格納する。

〔マクロ：Calc_Discount_Price〕

```
○マクロ: Calc_Discount_Price
○数値型: Goods_Num[50], Set_Num, Num, Val, I, J
○文字列型: Name
■ I: 0, I ≦ 49, 1
  • Name ← [ e ]
  • Goods_Num[I] ← 条件付合計(B7:B26, = Name, C7:C26)
■
■ I: 0, I ≦ 9, 1
  • Val ← Goods_Num[相対(セット値引き表!K3, I, 0) − 1] ／
          相対(セット値引き表!D3, I, 0)
  • Set_Num ← 整数部(Val)

  ■ J: 1, J ≦ 2, 1
    • Val ← Goods_Num[相対(セット値引き表!K3, I, J) − 1] ／
            相対(セット値引き表!D3, I, J ＊ 2)
    • Num ← 整数部(Val)
    ▲ [ f ]
    • Set_Num ← Num
  ■

  ▲ Set_Num ≠ 0
    • 相対(C30, I, 0) ← Set_Num
    ■ J: 0, J ≦ 2, 1
      • Goods_Num[相対(セット値引き表!K3, I, J) − 1] ←
              Goods_Num[相対(セット値引き表!K3, I, J) − 1] −
              相対(セット値引き表!D3, I, J ＊ 2) [ g ]
    ■
  ——
    • 相対(C30, I, 0) ← null
  ▼
■
```

c，dに関する解答群

ア　価格表!A$2:A$51　　イ　価格表!B$2:B$51

ウ　価格表!C$2:C$51　　エ　価格表!D$2:D$51

eに関する解答群

ア　価格表!A2　　イ　価格表!B2

ウ　価格表!C2　　エ　相対(価格表!A2, I, 0)

オ　相対(価格表!B2, I, 0)　　カ　相対(価格表!C2, I, 0)

fに関する解答群

ア　Num ＜ Set_Num　　イ　Num ＞ Set_Num

ウ　Num ＝ 0　　エ　Num ≠ 0

オ　Num ≠ Set_Num　　カ　Set_Num ＝ 0

キ　Set_Num ≠ 0

gに関する解答群

ア　＋ 1　　イ　＋ Set_Num

ウ　－ 1　　エ　－ Set_Num

オ　＊ Set_Num　　カ　／ Set_Num

■ Java プログラムで使用する API の説明

java.util

public interface List<E>

リスト（順序付けられたコレクション）のためのインタフェースを提供する。インタフェース Collection を継承する。

メソッド

public void add(int index, E e)

指定された要素をリスト内の指定された位置に挿入する。指定された位置及びその後に既に要素がある場合は，それらの要素をシフトする。

引数： index ― 挿入する位置のインデックス
e ― リストに追加する要素

public E remove(int index)

リスト内の指定された位置にある要素を削除する。

引数： index ― 返される要素のインデックス

戻り値：リスト内の指定された位置にあった要素

public boolean remove(Object obj)

指定された要素がこのリストにあれば，その最初の要素を削除する。

引数： obj ― リストから削除する要素

戻り値：リストから要素が削除されれば true
それ以外は false

public boolean isEmpty()

リストに要素がなければ true を返す。

戻り値：リストに要素が一つもなければ true
それ以外は false

public int size()

リスト内の要素数を返す。

戻り値：リスト内の要素数

java.util

public class ArrayList<E>

インタフェース List の配列による実装である。

メソッドの説明は，インタフェース List の項を参照。

コンストラクタ

public ArrayList()

空のリストを作る。

java.lang

public final class System

有用なクラスフィールドやクラスメソッドが定義されている。

メソッド

```
public static void arraycopy(Object src, int srcPos, Object dest,
                             int destPos, int length)
```

指定された複写元配列の指定位置から，指定された複写先配列の指定位置に配列要素を指定された個数分複写する。引数 src と引数 dest が同じ配列を参照している場合は，複写元と複写先の領域の重なりを考慮し，複写によるデータの破壊が起こらないようにする。

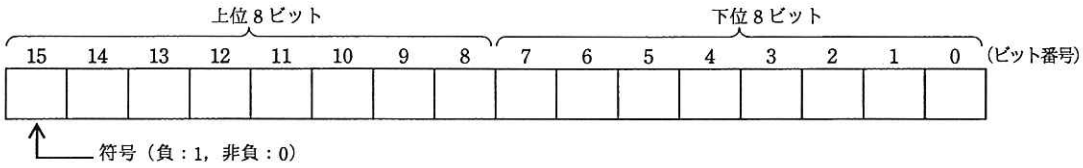
引数： src — 複写元配列
srcPos — 複写元配列内の開始位置
dest — 複写先配列
destPos — 複写先配列内の開始位置
length — 複写される配列要素の個数

■アセンブラ言語の仕様

1. システム COMETⅡの仕様

1.1 ハードウェアの仕様

(1) 1語は16ビットで，そのビット構成は，次のとおりである。



(2) 主記憶の容量は65536語で，そのアドレスは0～65535番地である。

(3) 数値は，16ビットの2進数で表現する。負数は，2の補数で表現する。

(4) 制御方式は逐次制御で，命令語は1語長又は2語長である。

(5) レジスタとして，GR（16ビット），SP（16ビット），PR（16ビット），FR（3ビット）の4種類がある。

GR（汎用レジスタ，General Register）は，GR0～GR7の8個があり，算術，論理，比較，シフトなどの演算に用いる。このうち，GR1～GR7のレジスタは，指標レジスタ（index register）としてアドレスの修飾にも用いる。

SP（スタックポインタ，Stack Pointer）は，スタックの最上段のアドレスを保持している。

PR（プログラムレジスタ，Program Register）は，次に実行すべき命令語の先頭アドレスを保持している。

FR（フラグレジスタ，Flag Register）は，OF（Overflow Flag），SF（Sign Flag），ZF（Zero Flag）と呼ぶ3個のビットからなり，演算命令などの実行によって次の値が設定される。これらの値は，条件付き分岐命令で参照される。

OF	算術演算命令の場合は，演算結果が−32768～32767に収まらなくなったとき1になり，それ以外のとき0になる。論理演算命令の場合は，演算結果が0～65535に収まらなくなったとき1になり，それ以外のとき0になる。
SF	演算結果の符号が負（ビット番号15が1）のとき1，それ以外のとき0になる。
ZF	演算結果が零（全部のビットが0）のとき1，それ以外のとき0になる。

(6) 論理加算又は論理減算は，被演算データを符号のない数値とみなして，加算又は減算する。

1.2 命令

命令の形式及びその機能を示す。ここで，一つの命令コードに対し2種類のオペランドがある場合，上段はレジスタ間の命令，下段はレジスタと主記憶間の命令を表す。

命令	書き方		命令の説明	FRの設定
	命令コード	オペランド		

(1) ロード，ストア，ロードアドレス命令

ロード LoaD	LD	r1,r2	r1 ← (r2)	○*1
		r,adr [,x]	r ← (実効アドレス)	
ストア STore	ST	r,adr [,x]	実効アドレス ← (r)	—
ロードアドレス Load ADdress	LAD	r,adr [,x]	r ← 実効アドレス	

(2) 算術，論理演算命令

算術加算 ADD Arithmetic	ADDA	r1,r2	r1 ← (r1) + (r2)	○
		r,adr [,x]	r ← (r) + (実効アドレス)	
論理加算 ADD Logical	ADDL	r1,r2	r1 ← (r1) $+_L$ (r2)	
		r,adr [,x]	r ← (r) $+_L$ (実効アドレス)	
算術減算 SUBtract Arithmetic	SUBA	r1,r2	r1 ← (r1) − (r2)	
		r,adr [,x]	r ← (r) − (実効アドレス)	
論理減算 SUBtract Logical	SUBL	r1,r2	r1 ← (r1) $-_L$ (r2)	
		r,adr [,x]	r ← (r) $-_L$ (実効アドレス)	
論理積 AND	AND	r1,r2	r1 ← (r1) AND (r2)	○*1
		r,adr [,x]	r ← (r) AND (実効アドレス)	
論理和 OR	OR	r1,r2	r1 ← (r1) OR (r2)	
		r,adr [,x]	r ← (r) OR (実効アドレス)	
排他的論理和 eXclusive OR	XOR	r1,r2	r1 ← (r1) XOR (r2)	
		r,adr [,x]	r ← (r) XOR (実効アドレス)	

(3) 比較演算命令

算術比較 ComPare Arithmetic	CPA	r1,r2	(r1) と (r2)，又は (r) と (実効アドレス) の算術比較又は論理比較を行い，比較結果によって，FR に次の値を設定する。	○*1
		r,adr [,x]		
論理比較 ComPare Logical	CPL	r1,r2		
		r,adr [,x]		

比較結果	FR の値 SF	FR の値 ZF
(r1) ＞ (r2) (r) ＞ (実効アドレス)	0	0
(r1) ＝ (r2) (r) ＝ (実効アドレス)	0	1
(r1) ＜ (r2) (r) ＜ (実効アドレス)	1	0

(4) シフト演算命令

算術左シフト Shift Left Arithmetic	SLA	r,adr [,x]	符号を除き (r) を実効アドレスで指定したビット数だけ左又は右にシフトする。 シフトの結果，空いたビット位置には，左シフトのときは 0，右シフトのときは符号と同じものが入る。	○*2
算術右シフト Shift Right Arithmetic	SRA	r,adr [,x]		
論理左シフト Shift Left Logical	SLL	r,adr [,x]	符号を含み (r) を実効アドレスで指定したビット数だけ左又は右にシフトする。 シフトの結果，空いたビット位置には 0 が入る。	
論理右シフト Shift Right Logical	SRL	r,adr [,x]		

(5) 分岐命令

正分岐 Jump on PLus	JPL	adr [,x]	FR の値によって，実効アドレスに分岐する。分岐しないときは，次の命令に進む。	—
負分岐 Jump on MInus	JMI	adr [,x]		
非零分岐 Jump on Non Zero	JNZ	adr [,x]		
零分岐 Jump on ZEro	JZE	adr [,x]		
オーバフロー分岐 Jump on OVerflow	JOV	adr [,x]		
無条件分岐 unconditional JUMP	JUMP	adr [,x]	無条件に実効アドレスに分岐する。	

命令	分岐するときの FR の値 OF	SF	ZF
JPL		0	0
JMI		1	
JNZ			0
JZE			1

(6) スタック操作命令

プッシュ PUSH	PUSH adr [,x]	SP ← (SP) $-_L$ 1, (SP) ← 実効アドレス	—
ポップ POP	POP r	r ← ((SP)), SP ← (SP) $+_L$ 1	

(7) コール，リターン命令

コール CALL subroutine	CALL adr [,x]	SP ← (SP) $-_L$ 1, (SP) ← (PR), PR ← 実効アドレス	—
リターン RETurn from subroutine	RET	PR ← ((SP)), SP ← (SP) $+_L$ 1	

(8) その他

スーパバイザコール SuperVisor Call	SVC adr [,x]	実効アドレスを引数として割出しを行う。実行後の GR と FR は不定となる。	—
ノーオペレーション No OPeration	NOP	何もしない。	

注記

r, r1, r2	いずれも GR を示す。指定できる GR は GR0 ～ GR7
adr	アドレスを示す。指定できる値の範囲は 0 ～ 65535
x	指標レジスタとして用いる GR を示す。指定できる GR は GR1 ～ GR7
[ ]	[ ] 内の指定は省略できることを示す。
( )	( ) 内のレジスタ又はアドレスに格納されている内容を示す。
実効アドレス	adr と x の内容との論理加算値又はその値が示す番地
←	演算結果を，左辺のレジスタ又はアドレスに格納することを示す。
$+_L$, $-_L$	論理加算，論理減算を示す。
FR の設定	○ ：設定されることを示す。 ○*1 ：設定されることを示す。ただし，OF には 0 が設定される。 ○*2 ：設定されることを示す。ただし，OF にはレジスタから最後に送り出されたビットの値が設定される。 － ：実行前の値が保持されることを示す。

1.3 文字の符号表

(1) JIS X 0201 ラテン文字・片仮名用 8 ビット符号で規定する文字の符号表を使用する。

(2) 右に符号表の一部を示す。1 文字は 8 ビットからなり，上位 4 ビットを列で，下位 4 ビットを行で示す。例えば，間隔，4，H，¥ のビット構成は，16 進表示で，それぞれ 20，34，48，5C である。16 進表示で，ビット構成が 21 ～ 7E（及び表では省略している A1 ～ DF）に対応する文字を図形文字という。図形文字は，表示（印刷）装置で，文字として表示（印字）できる。

(3) この表にない文字とそのビット構成が必要な場合は，問題中で与える。

行＼列	02	03	04	05	06	07
0	間隔	0	@	P	`	p
1	!	1	A	Q	a	q
2	"	2	B	R	b	r
3	#	3	C	S	c	s
4	$	4	D	T	d	t
5	%	5	E	U	e	u
6	&	6	F	V	f	v
7	'	7	G	W	g	w
8	(	8	H	X	h	x
9	)	9	I	Y	i	y
10	*	:	J	Z	j	z
11	+	;	K	[	k	{
12	,	<	L	¥	l	|
13	-	=	M	]	m	}
14	.	>	N	^	n	~
15	/	?	O	_	o	

2. アセンブラ言語 CASLⅡの仕様

2.1 言語の仕様

(1) CASLⅡは，COMETⅡのためのアセンブラ言語である。
(2) プログラムは，命令行及び注釈行からなる。
(3) 1命令は1命令行で記述し，次の行へ継続できない。
(4) 命令行及び注釈行は，次に示す記述の形式で，行の1文字目から記述する。

行の種類		記述の形式
命令行	オペランドあり	[ラベル] {空白} {命令コード} {空白} {オペランド} [{空白} [コメント]]
	オペランドなし	[ラベル] {空白} {命令コード} [{空白} [{;} [コメント]]]
注釈行		[空白] {;} [コメント]

注記 [ ] [ ] 内の指定が省略できることを示す。
{ } { } 内の指定が必須であることを示す。
ラベル その命令の（先頭の語の）アドレスを他の命令やプログラムから参照するための名前である。長さは 1～8 文字で，先頭の文字は英大文字でなければならない。以降の文字は，英大文字又は数字のいずれでもよい。なお，予約語である GR0～GR7 は，使用できない。
空白 1文字以上の間隔文字の列である。
命令コード 命令ごとに記述の形式が定義されている。
オペランド 命令ごとに記述の形式が定義されている。
コメント 覚え書きなどの任意の情報であり，処理系で許す任意の文字を書くことができる。

2.2 命令の種類

命令は，4 種類のアセンブラ命令（START，END，DS，DC），4 種類のマクロ命令（IN，OUT，RPUSH，RPOP）及び機械語命令（COMETⅡの命令）からなる。その仕様を次に示す。

命令の種類	ラベル	命令コード	オペランド	機能
アセンブラ命令	ラベル	START	[実行開始番地]	プログラムの先頭を定義 プログラムの実行開始番地を定義 他のプログラムで参照する入口名を定義
		END		プログラムの終わりを明示
	[ラベル]	DS	語数	領域を確保
	[ラベル]	DC	定数 [,定数] …	定数を定義
マクロ命令	[ラベル]	IN	入力領域,入力文字長領域	入力装置から文字データを入力
	[ラベル]	OUT	出力領域,出力文字長領域	出力装置へ文字データを出力
	[ラベル]	RPUSH		GR の内容をスタックに格納
	[ラベル]	RPOP		スタックの内容を GR に格納
機械語命令	[ラベル]	（「1.2 命令」を参照）		

2.3 アセンブラ命令

アセンブラ命令は，アセンブラの制御などを行う。

(1)

START	[実行開始番地]

START 命令は，プログラムの先頭を定義する。
実行開始番地は，そのプログラム内で定義されたラベルで指定する。指定がある場合はその番地から，省略した場合は START 命令の次の命令から，実行を開始する。
また，この命令につけられたラベルは，他のプログラムから入口名として参照できる。

(2) | END | |

END 命令は，プログラムの終わりを定義する。

(3) | DS | 語数 |

DS 命令は，指定した語数の領域を確保する。

語数は，10 進定数（≧0）で指定する。語数を 0 とした場合，領域は確保しないが，ラベルは有効である。

(4) | DC | 定数 [,定数] … |

DC 命令は，定数で指定したデータを（連続する）語に格納する。

定数には，10 進定数，16 進定数，文字定数，アドレス定数の 4 種類がある。

定数の種類	書き方	命 令 の 説 明
10 進定数	n	n で指定した 10 進数値を，1 語の 2 進数データとして格納する。ただし，n が−32768 ～ 32767 の範囲にないときは，その下位 16 ビットを格納する。
16 進定数	#h	h は 4 けたの 16 進数（16 進数字は 0 ～ 9，A ～ F）とする。h で指定した 16 進数値を 1 語の 2 進数データとして格納する（0000 ≦ h ≦ FFFF）。
文字定数	'文字列'	文字列の文字数（＞0）分の連続する領域を確保し，最初の文字は第 1 語の下位 8 ビットに，2 番目の文字は第 2 語の下位 8 ビットに，…と順次文字データとして格納する。各語の上位 8 ビットには 0 のビットが入る。 文字列には，間隔及び任意の図形文字を書くことができる。ただし，アポストロフィ（'）は 2 個続けて書く。
アドレス定数	ラベル	ラベルに対応するアドレスを 1 語の 2 進数データとして格納する。

2.4 マクロ命令

マクロ命令は，あらかじめ定義された命令群とオペランドの情報によって，目的の機能を果たす命令群を生成する（語数は不定）。

(1) | IN | 入力領域,入力文字長領域 |

IN 命令は，あらかじめ割り当てた入力装置から，1 レコードの文字データを読み込む。

入力領域は，256 語長の作業域のラベルであり，この領域の先頭から，1 文字を 1 語に対応させて順次入力される。レコードの区切り符号（キーボード入力の復帰符号など）は，格納しない。格納の形式は，DC 命令の文字定数と同じである。入力データが 256 文字に満たない場合，入力領域の残りの部分は実行前のデータを保持する。入力データが 256 文字を超える場合，以降の文字は無視される。

入力文字長領域は，1 語長の領域のラベルであり，入力された文字の長さ（≧0）が 2 進数で格納される。ファイルの終わり（end of file）を検出した場合は，−1 が格納される。

IN 命令を実行すると，GR の内容は保存されるが，FR の内容は不定となる。

(2) | OUT | 出力領域,出力文字長領域 |

OUT 命令は，あらかじめ割り当てた出力装置に，文字データを，1 レコードとして書き出す。

出力領域は，出力しようとするデータが 1 文字 1 語で格納されている領域のラベルである。格納の形式は，DC 命令の文字定数と同じであるが，上位 8 ビットは，OS が無視するので 0 でなくてもよい。

出力文字長領域は，1 語長の領域のラベルであり，出力しようとする文字の長さ（≧0）を 2 進数で格納しておく。

OUT 命令を実行すると，GR の内容は保存されるが，FR の内容は不定となる。

(3)

RPUSH	

RPUSH 命令は，GR の内容を，GR1，GR2，…，GR7 の順序でスタックに格納する。

(4)

RPOP	

RPOP 命令は，スタックの内容を順次取り出し，GR7，GR6，…，GR1 の順序で GR に格納する。

2.5 機械語命令

機械語命令のオペランドは，次の形式で記述する。

r，r1，r2	GR は，記号 GR0 ～ GR7 で指定する。
x	指標レジスタとして用いる GR は，記号 GR1 ～ GR7 で指定する。
adr	アドレスは，10 進定数，16 進定数，アドレス定数又はリテラルで指定する。 リテラルは，一つの 10 進定数，16 進定数又は文字定数の前に等号（=）を付けて記述する。CASLⅡは，等号の後の定数をオペランドとする DC 命令を生成し，そのアドレスを adr の値とする。

2.6 その他

(1) アセンブラによって生成される命令語や領域の相対位置は，アセンブラ言語での記述順序とする。ただし，リテラルから生成される DC 命令は，END 命令の直前にまとめて配置される。

(2) 生成された命令語，領域は，主記憶上で連続した領域を占める。

3. プログラム実行の手引

3.1 OS

プログラムの実行に関して，次の取決めがある。

(1) アセンブラは，未定義ラベル（オペランド欄に記述されたラベルのうち，そのプログラム内で定義されていないラベル）を，他のプログラムの入口名（START 命令のラベル）と解釈する。この場合，アセンブラはアドレスの決定を保留し，その決定を OS に任せる。OS は，実行に先立って他のプログラムの入口名との連係処理を行いアドレスを決定する（プログラムの連係）。

(2) プログラムは，OS によって起動される。プログラムがロードされる主記憶の領域は不定とするが，プログラム中のラベルに対応するアドレス値は，OS によって実アドレスに補正されるものとする。

(3) プログラムの起動時に，OS はプログラム用に十分な容量のスタック領域を確保し，その最後のアドレスに 1 を加算した値を SP に設定する。

(4) OS は，CALL 命令でプログラムに制御を渡す。プログラムを終了し OS に制御を戻すときは，RET 命令を使用する。

(5) IN 命令に対応する入力装置，OUT 命令に対応する出力装置の割当ては，プログラムの実行に先立って利用者が行う。

(6) OS は，入出力装置や媒体による入出力手続の違いを吸収し，システムでの標準の形式及び手続（異常処理を含む）で入出力を行う。したがって，IN，OUT 命令では，入出力装置の違いを意識する必要はない。

3.2 未定義事項

プログラムの実行等に関し，この仕様で定義しない事項は，処理系によるものとする。

表計算ソフトの機能・用語（基本情報技術者試験用）

表計算ソフトの機能，用語などは，原則として次による。

なお，ワークシートの保存，読出し，印刷，罫線作成やグラフ作成など，ここで示す以外の機能などを使用するときには，問題文中に示す。

1. ワークシート

(1) 列と行とで構成される升目の作業領域をワークシートという。ワークシートの大きさは 256 列，10,000 行とする。

(2) ワークシートの列と行のそれぞれの位置は，列番号と行番号で表す。列番号は，最左端列の列番号を A とし，A，B，…，Z，AA，AB，…，AZ，BA，BB，…，BZ，…，IU，IV と表す。行番号は，最上端行の行番号を 1 とし，1，2，…，10000 と表す。

(3) 複数のワークシートを利用することができる。このとき，各ワークシートには一意のワークシート名を付けて，他のワークシートと区別する。

2. セルとセル範囲

(1) ワークシートを構成する各升をセルという。その位置は列番号と行番号で表し，それをセル番地という。

［例］列 A 行 1 にあるセルのセル番地は，A1 と表す。

(2) ワークシート内のある長方形の領域に含まれる全てのセルの集まりを扱う場合，長方形の左上端と右下端のセル番地及び“：”を用いて，“左上端のセル番地：右下端のセル番地”と表す。これを，セル範囲という。

［例］左上端のセル番地が A1 で，右下端のセル番地が B3 のセル範囲は，A1:B3 と表す。

(3) 他のワークシートのセル番地又はセル範囲を指定する場合には，ワークシート名と“！”を用い，それぞれ“ワークシート名！セル番地”又は“ワークシート名！セル 範囲”と表す。

［例］ワークシート“シート1”のセル B5 ～ G10 を，別のワークシートから指定する場合には，
シート1!B5:G10 と表す。

3. 値と式

(1) セルは値をもち，その値はセル番地によって参照できる。値には，数値，文字列，論理値及び空値がある。

(2) 文字列は一重引用符“ ’ ”で囲って表す。

［例］文字列“ A ”，“ BC ”は，それぞれ 'A'，'BC' と表す。

(3) 論理値の真を true，偽を false と表す。

(4) 空値を null と表し，空値をもつセルを空白セルという。セルの初期状態は，空白セルとする。

(5) セルには，式を入力することができる。セルは，式を評価した結果の値をもつ。

(6) 式は，定数，セル番地，演算子，括弧及び関数から構成される。定数は，数値，文字列，論理値又は空値を表す表記とする。式中のセル番地は，その番地のセルの値を参照する。

(7) 式には，算術式，文字式及び論理式がある。評価の結果が数値となる式を算術式，文字列となる式を文字式，論理値となる式を論理式という。

(8) セルに式を入力すると，式は直ちに評価される。式が参照するセルの値が変化したときには，直ちに，適切に再評価される。

4. 演算子

(1) 単項演算子は，正符号“＋”及び負符号 “－”とする。

(2) 算術演算子は，加算“＋”，減算“－”，乗算“＊”，除算“／”及びべき乗“＾”とする。

(3) 比較演算子は，より大きい“＞”，より小さい“＜”，以上“≧”，以下“≦”，等しい“＝”及び等しくない“≠”とする。

(4) 括弧は丸括弧“(”及び“)”を使う。

(5) 式中に複数の演算及び括弧があるときの計算の順序は，次表の優先順位に従う。

演算の種類	演算子	優先順位
括弧	( )	高
べき乗演算	＾	▲
単項演算	＋，－	｜
乗除演算	＊，／	｜
加減演算	＋，－	▼
比較演算	＞，＜，≧，≦，＝，≠	低

5. セルの複写

(1) セルの値又は式を，他のセルに複写することができる。

(2) セルを複写する場合で，複写元のセル中にセル番地を含む式が入力されているとき，複写元と複写先のセル番地の差を維持するように，式中のセル番地を変化させるセルの参照方法を相対参照という。この場合，複写先のセルとの列番号の差及び行番号の差を，複写元のセルに入力された式中の各セル番地に加算した式が，複写先のセルに入る。

［例］セル A6 に式 A1 ＋ 5 が入力されているとき，このセルをセル B8 に複写すると，セル B8 には式 B3 ＋ 5 が入る。

(3) セルを複写する場合で，複写元のセル中にセル番地を含む式が入力されているとき，そのセル番地の列番号と行番号の両方又は片方を変化させないセルの参照方法を絶対参照という。絶対参照を適用する列番号と行番号の両方又は片方の直前には“$”を付ける。

［例］セル B1 に式 A1 ＋ $A2 ＋ A$5 が入力されているとき，このセルをセル C4 に複写す

ると，セル C4 には式 A1 + $A5 + B$5 が入る。

(4) セルを複写する場合で，複写元のセル中に，他のワークシートを参照する式が入力されているとき，その参照するワークシートのワークシート名は複写先でも変わらない。

［例］ワークシート“シート2”のセル A6 に式 シート1!A1 が入力されているとき，このセルをワークシート“シート3”のセル B8 に複写すると，セル B8 には式 シート1!B3 が入る。

6. 関数

式には次の表で定義する関数を利用することができる。

書式	解　　説
合計 (セル範囲[1])	セル範囲に含まれる数値の合計を返す。 ［例］合計 (A1:B5) は，セル A1 ～ B5 に含まれる数値の合計を返す。
平均 (セル範囲[1])	セル範囲に含まれる数値の平均を返す。
標本標準偏差 (セル範囲[1])	セル範囲に含まれる数値を標本として計算した標準偏差を返す。
母標準偏差 (セル範囲[1])	セル範囲に含まれる数値を母集団として計算した標準偏差を返す。
最大 (セル範囲[1])	セル範囲に含まれる数値の最大値を返す。
最小 (セル範囲[1])	セル範囲に含まれる数値の最小値を返す。
IF (論理式 , 式1, 式2)	論理式の値が true のとき式 1 の値を，false のとき式 2 の値を返す。 ［例］IF (B3 ＞ A4,' 北海道 ',C4) は，セル B3 の値がセル A4 の値より大きいとき文字列“北海道”を，それ以外のときセル C4 の値を返す。
個数 (セル範囲)	セル範囲に含まれるセルのうち，空白セルでないセルの個数を返す。
条件付個数 (セル範囲 , 検索条件の記述)	セル範囲に含まれるセルのうち，検索条件の記述で指定された条件を満たすセルの個数を返す。検索条件の記述は比較演算子と式の組で記述し，セル範囲に含まれる各セルと式の値を，指定した比較演算子によって評価する。 ［例1］条件付個数 (H5:L9, ＞ A1) は，セル H5 ～ L9 のセルのうち，セル A1 の値より大きな値をもつセルの個数を返す。 ［例2］条件付個数 (H5:L9, ＝ 'A4') は，セル H5 ～ L9 のセルのうち，文字列“A4 ”をもつセルの個数を返す。
整数部 (算術式)	算術式の値以下で最大の整数を返す。 ［例1］整数部 (3.9) は，3 を返す。 ［例2］整数部 (−3.9) は，−4 を返す。
剰余 (算術式1, 算術式2)	算術式1 の値を被除数，算術式2 の値を除数として除算を行ったときの剰余を返す。関数“剰余”と“整数部”は，剰余 (x,y) ＝ x − y ＊ 整数部 (x ／ y) という関係を満たす。 ［例1］剰余 (10,3) は，1 を返す。 ［例2］剰余 (−10,3) は，2 を返す。
平方根 (算術式)	算術式の値の非負の平方根を返す。算術式の値は，非負の数値でなければならない。
論理積 (論理式1, 論理式2, …)[2]	論理式1, 論理式2, … の値が全て true のとき，true を返す。それ以外のとき false を返す。
論理和 (論理式1, 論理式2, …)[2]	論理式1, 論理式2, … の値のうち，少なくとも一つが true のとき，true を返す。それ以外のとき false を返す。
否定 (論理式)	論理式の値が true のとき false を，false のとき true を返す。

切上げ(算術式,桁位置) 四捨五入(算術式,桁位置) 切捨て(算術式,桁位置)	算術式の値を指定した桁位置で，関数“切上げ”は切り上げた値を，関数“四捨五入”は四捨五入した値を，関数“切捨て”は切り捨てた値を返す。ここで，桁位置は小数第1位の桁を0とし，右方向を正として数えたときの位置とする。 ［例1］切上げ(−314.059,2) は，−314.06 を返す。 ［例2］切上げ(314.059,−2) は，400 を返す。 ［例3］切上げ(314.059,0) は，315 を返す。
結合(式1,式2,…)[2]	式1，式2，…のそれぞれの値を文字列として扱い，それらを引数の順につないでできる一つの文字列を返す。 ［例］結合('北海道','九州',123,456) は，文字列“北海道九州123456”を返す。
順位(算術式,セル範囲[1],順序の指定)	セル範囲の中での算術式の値の順位を，順序の指定が0の場合は昇順で，1の場合は降順で数えて，その順位を返す。ここで，セル範囲の中に同じ値がある場合，それらを同順とし，次の順位は同順の個数だけ加算した順位とする。
乱数( )	0 以上 1 未満の一様乱数（実数値）を返す。
表引き(セル範囲,行の位置,列の位置)	セル範囲の左上端から行と列をそれぞれ1，2，…と数え，セル範囲に含まれる行の位置と列の位置で指定した場所にあるセルの値を返す。 ［例］表引き(A3:H11,2,5) は，セル E4 の値を返す。
垂直照合(式,セル範囲,列の位置,検索の指定)	セル範囲の左端列を上から下に走査し，検索の指定によって指定される条件を満たすセルが現れる最初の行を探す。その行に対して，セル範囲の左端列から列を1，2，…と数え，セル範囲に含まれる列の位置で指定した列にあるセルの値を返す。 ・検索の指定が0の場合の条件：式の値と一致する値を検索する。 ・検索の指定が1の場合の条件：式の値以下の最大値を検索する。このとき，左端列は上から順に昇順に整列されている必要がある。 ［例］垂直照合(15,A2:E10,5,0) は，セル範囲の左端列をセル A2，A3，…，A10 と探す。このとき，セル A6 で 15 を最初に見つけたとすると，左端列 A から数えて 5 列目の列 E 中で，セル A6 と同じ行にあるセル E6 の値を返す。
水平照合(式,セル範囲,行の位置,検索の指定)	セル範囲の上端行を左から右に走査し，検索の指定によって指定される条件を満たすセルが現れる最初の列を探す。その列に対して，セル範囲の上端行から行を1，2，…と数え，セル範囲に含まれる行の位置で指定した行にあるセルの値を返す。 ・検索の指定が0の場合の条件：式の値と一致する値を検索する。 ・検索の指定が1の場合の条件：式の値以下の最大値を検索する。このとき，上端行は左から順に昇順に整列されている必要がある。 ［例］水平照合(15,A2:G6,5,1) は，セル範囲の上端行をセル A2，B2，…，G2 と探す。このとき，15 以下の最大値をセル D2 で最初に見つけたとすると，上端行 2 から数えて 5 行目の行 6 中で，セル D2 と同じ列にあるセル D6 の値を返す。
照合検索(式,検索のセル範囲,抽出のセル範囲)	1行又は1列を対象とする同じ大きさの検索のセル範囲と抽出のセル範囲に対して，検索のセル範囲を左端又は上端から走査し，式の値と一致する最初のセルを探す。見つかったセルの検索のセル範囲の中での位置と，抽出のセル範囲の中での位置が同じセルの値を返す。 ［例］照合検索(15,A1:A8,C6:C13) は，セル A1 ～ A8 をセル A1，A2，…と探す。このとき，セル A5 で 15 を最初に見つけたとすると，セル C6 ～ C13 の上端から数えて 5 番目のセル C10 の値を返す。

照合一致(式,セル範囲,検索の指定)	1行又は1列を対象とするセル範囲に対して，セル範囲の左端又は上端から走査し，検索の指定によって指定される条件を満たす最初のセルを探す。見つかったセルの位置を，セル範囲の左端又は上端から1，2，…と数えた値とし，その値を返す。 ・検索の指定が0の場合の条件：式の値と一致する値を検索する。 ・検索の指定が1の場合の条件：式の値以下の最大値を検索する。このとき，セル範囲は左端又は上端から順に昇順に整列されている必要がある。 ・検索の指定が −1 の場合の条件：式の値以上の最小値を検索する。このとき，セル範囲は左端又は上端から順に降順に整列されている必要がある。 ［例］照合一致(15,B2:B12,−1) は，セル B2 ～ B12 をセル B2，B3，…と探す。このとき，15 以上の最小値をセルB9 で最初に見つけたとすると，セル B2 から数えた値 8 を返す。
条件付合計(検索のセル範囲,検索条件の記述,合計のセル範囲[1])	行数及び列数が共に同じ検索のセル範囲と合計のセル範囲に対して，検索と合計を行う。検索のセル範囲に含まれるセルのうち，検索条件の記述で指定される条件を満たすセルを全て探す。検索条件の記述を満たした各セルについての左上端からの位置と，合計のセル範囲中で同じ位置にある各セルの値を合計して返す。 検索条件の記述は比較演算子と式の組で記述し，検索のセル範囲に含まれる各セルと式の値を，指定した比較演算子によって評価する。 ［例1］条件付合計 (A1:B8,＞E1,C2:D9) は，検索のセル範囲であるセル A1 ～ B8 のうち，セル E1 の値より大きな値をもつ全てのセルを探す。このとき，セル A2，B4，B7 が見つかったとすると，合計のセル範囲であるセル C2 ～ D9 の左上端からの位置が同じであるセル C3，D5，D8 の値を合計して返す。 ［例2］条件付合計 (A1:B8,＝160,C2:D9) は，検索のセル範囲であるセル A1 ～ B8 のうち，160 と一致する値をもつ全てのセルを探す。このとき，セル A2，B4，B7 が見つかったとすると，合計のセル範囲であるセル C2 ～ D9 の左上端からの位置が同じであるセル C3，D5，D8 の値を合計して返す。

注[1] 引数として渡したセル範囲の中で，数値以外の値は処理の対象としない。

[2] 引数として渡すことができる式の個数は，1 以上である。

7. マクロ

(1) ワークシートとマクロ

ワークシートには複数のマクロを格納することができる。

マクロは一意のマクロ名を付けて宣言する。マクロの実行は，表計算ソフトのマクロの実行機能を使って行う。

［例］○マクロ: Pro

例は，マクロ Pro の宣言である。

(2) 変数とセル変数

変数の型には，数値型，文字列型及び論理型があり，変数は宣言することで使用できる。変数名にセル番地を使用することはできない。

［例］○数値型: row, col

例は，数値型の変数 row，col の宣言である。

セルを変数として使用でき，これをセル変数という。セル変数は，宣言せずに使用できる。

セル変数の表現方法には，絶対表現と相対表現とがある。

セル変数の絶対表現は，セル番地で表す。

セル変数の相対表現は，次の書式で表す。

書式	解　　説
相対(セル変数,行の位置,列の位置)	セル変数で指定したセルを基準のセルとする。そのセルの行番号と列番号の位置を 0 とし，下又は右方向を正として数え，行の位置と列の位置で指定した数と一致する場所にあるセルを表す変数である。

［例1］相対(B5, 2, 3) は，セル E7 を表す変数である。

［例2］相対(B5, −2, −1) は，セル A3 を表す変数である。

(3) 配列

数値型，文字列型又は論理型の配列は宣言することで使用できる。添字を“[”及び“]”で囲み，添字が複数ある場合はコンマで区切る。添字は 0 から始まる。

なお，数値型及び文字列型の変数及び配列の要素には，空値を格納することができる。

［例］○文字列型: table[100, 200]

例は，100 × 200 個 の文字列型の要素をもつ 2 次元配列 table の宣言である。

(4) 宣言，注釈及び処理

宣言，注釈及び処理の記述は，“共通に使用される擬似言語の記述形式”の〔宣言，注釈及び処理〕に従う。

処理の記述中に式又は関数を使用する場合，その記述中に変数，セル変数又は配列の要素が使用できる。

［例］○数値型: row

```
row: 0, row < 5, 1
  ・相対(E1, row, 1) ← 垂直照合(相対(E1, row, 0), A1:B10, 2, 0) * 10
```

例は，セル E1，E2，…，E5 の各値に対して，セル A1 ～ A10 の中で同じ値をもつセルが現れる最初の行を探し，見つけた行の列 B のセルの値を10倍し，セル F1，F2，…，F5 の順に代入する。

〔 メ モ 用 紙 〕

〔 メ モ 用 紙 〕

〔 メ モ 用 紙 〕

6. 退室可能時間に途中で退室する場合には，手を挙げて監督員に合図し，答案用紙が回収されてから静かに退室してください。

退室可能時間	13:40 ～ 15:20

7. **問題に関する質問にはお答えできません。**文意どおり解釈してください。
8. 問題冊子の余白などは，適宜利用して構いません。
9. Java プログラムで使用する API の説明，アセンブラ言語の仕様及び表計算ソフトの機能・用語は，この冊子の末尾を参照してください。
10. 試験時間中，机上に置けるものは，次のものに限ります。

    なお，会場での貸出しは行っていません。

    受験票，黒鉛筆及びシャープペンシル（B又はHB），鉛筆削り，消しゴム，定規，時計（時計型ウェアラブル端末は除く。アラームなど時計以外の機能は使用不可），ハンカチ，ポケットティッシュ，目薬

    これら以外は机上に置けません。使用もできません。
11. 試験終了後，この問題冊子は持ち帰ることができます。
12. 答案用紙は，いかなる場合でも提出してください。回収時に提出しない場合は，採点されません。
13. 試験時間中にトイレへ行きたくなったり，気分が悪くなったりした場合は，手を挙げて監督員に合図してください。

試験問題に記載されている会社名又は製品名は，それぞれ各社又は各組織の商標又は登録商標です。
なお，試験問題では，™ 及び ® を明記していません。

©2015　独立行政法人情報処理推進機構