FE

平成22年度　春期
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. この注意事項は，問題冊子の**裏表紙**に続きます。必ず読んでください。
4. **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。**
5. 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7	問8	問9 ～ 問13
選択方法	5問選択	必須	1問選択

6. 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) B又はHBの黒鉛筆又はシャープペンシルを使用してください。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) 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，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。**マークの記入方法**のとおりマークされていない場合は，読み取れないことがあります。
   (3) **受験番号欄**に，**受験番号**を記入及びマークしてください。正しくマークされていない場合は，採点されません。
   (4) **生年月日欄**に，受験票に印字されているとおりの**生年月日**を記入及びマークしてください。正しく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。
   (5) **選択した問題**については，右の例に従って，**選択欄**の**問題番号**の(選)をマークしてください。マークがない場合は，採点の対象になりません。問1～問7について，6問以上マークした場合は，はじめの5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。
   (6) 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。

〔問1，問3，問4，問6，問7，問9を選択した場合の例〕

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

〔例題〕　次の［　　］に入れる正しい答えを，解答群の中から選べ。

春の情報処理技術者試験は，［ a ］月に実施される。

解答群　　ア　2　　イ　3　　ウ　4　　エ　5

正しい答えは“ウ　4”ですから，次のようにマークしてください。

例題	a	(ア) (イ) ● (エ)

裏表紙の注意事項も，必ず読んでください。

〔問題一覧〕

●問 1～問 7（7 問中 5 問選択）

問題番号	出題分野	テーマ
問 1	ハードウェア	キャッシュメモリ
問 2	ソフトウェア	コンパイラの処理内容
問 3	データベース	関係データベース
問 4	ネットワーク	動画のストリーミングサーバの設置計画
問 5	ソフトウェア設計	配達サービス管理システム
問 6	プロジェクトマネジメント	プロジェクトにおける品質管理
問 7	経営・関連法規	事業の分析

●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	マージソート

●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）	英文テキストの整形出力
問 10	ソフトウェア開発（COBOL）	セキュリティルームの入退室管理とログ解析
問 11	ソフトウェア開発（Java）	リバーシゲームの支援
問 12	ソフトウェア開発（アセンブラ）	浮動小数点数の加算
問 13	ソフトウェア開発（表計算）	喫茶店の料金計算

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　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**から**問7**までの7問については，この中から5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，6問以上選択した場合には，はじめの5問について採点します。

問1　キャッシュメモリに関する次の記述を読んで，設問1，2に答えよ。

キャッシュメモリとは，主記憶とCPUの間に置く高速アクセスが可能なメモリである。キャッシュメモリとCPU及び主記憶との関係を図1に示す。データをキャッシュメモリに保持しておくことによって，CPUは速度の遅い主記憶に直接アクセスしなくて済むので，処理の高速化が図れる。

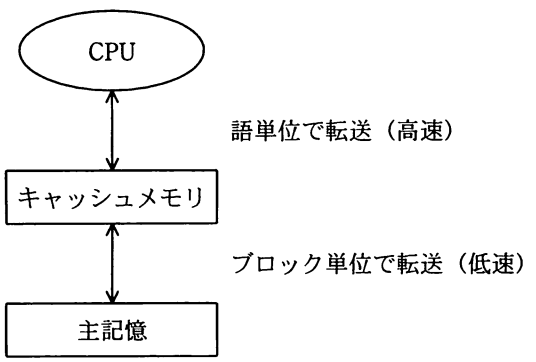


図1　キャッシュメモリとCPU及び主記憶との関係

ここでは，ハードウェアのアーキテクチャを次のように仮定する。

(1)　主記憶はブロック（1ブロックは100語から成る）に分割されている。各ブロックには，その先頭番地が小さいものから順に1，2，3，…とブロック番号が振られている。主記憶とキャッシュメモリ間はブロック単位でデータが転送される。

(2)　キャッシュメモリには，命令を保持しておく命令キャッシュと，データを保持しておくデータキャッシュの2種類がある。ここでは，データキャッシュ（以下，キャッシュという）だけを考える。

(3)　キャッシュの構成は，図2のとおりとする。

①　キャッシュは，ディレクトリ部とデータ部から成る。

②　データ部はバッファ1～3の三つのバッファから成り，各バッファは1ブロック分の主記憶の内容を保持できる。

③ ディレクトリ部は，データ部のバッファ1～3に対応したディレクトリ1～3から成る。それぞれのディレクトリは次の内容を保持する三つのフィールドから成る。

なお，データ部のバッファが未使用の場合は，対応するディレクトリの三つのフィールドすべてに0が入っている。

(イ) ブロック番号：対応するデータ部のバッファが保持する主記憶のブロック番号

(ロ) 順位　：キャッシュ内に最も古くから存在するブロックから順に 1，2，3 と番号が振られる。

(ハ) フラグ：対応するデータ部のバッファにブロックを読み込んだとき，0に初期化される。対応するデータ部のバッファに保持されている内容がCPUの処理によって変更されると，1に変わる。

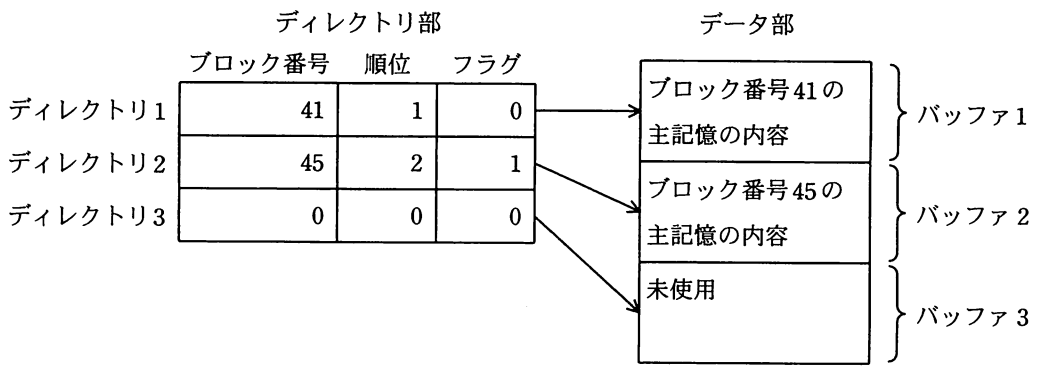


図2　キャッシュの構成

擬似言語で表現された次の繰返し処理を実行する場合について考える。

〔繰返し処理〕

```
■ i: 99, i ≧ 0, −1
  ・A[i] ← A[i] + B[i] + C[i]
  ・D[i] ← A[i]
```

(1) 配列A，B，C及びDは100個の要素から成り，1要素は1語である。添字は0から始まるものとする。

(2) データ領域の主記憶への割付けは，次のとおりとする。

① Aの配列領域：4000～4099番地（ブロック番号41）

A[0]は4000番地，A[1]は4001番地という順に割り付けられる。

② Bの配列領域：4100～4199番地（ブロック番号42）

③ Cの配列領域：4200～4299番地（ブロック番号43）

④ Dの配列領域：4300～4399番地（ブロック番号44）

⑤ 定数－1と99の格納領域：4400，4401番地（ブロック番号45）

(3) 変数iはレジスタを使用し，主記憶への割付けは行わない。

(4) プログラム領域の内容は表1のとおりとする。参照ブロックの数値は，その命令を実行するときにCPUが参照するデータのブロック番号を示す。

表1　プログラム領域の内容

番地	命令	処理の内容	参照ブロック
1000	LOAD　R1,4400	－1をR1に設定	45
1001	LOAD　R2,4401	iの初期値(99)をR2に設定	45
1002	LOAD　R3,4000,R2	A[i]の内容をR3に設定	41
1003	ADD　R3,4100,R2	B[i]の内容をR3に加算	42
1004	ADD　R3,4200,R2	C[i]の内容をR3に加算	43
1005	STORE R3,4000,R2	R3の内容をA[i]に格納	41
1006	STORE R3,4300,R2	R3の内容をD[i]に格納	44
1007	ADDR　R2,R1	i ← i － 1	－
1008	JNM　1002	i ≧ 0ならば1002番地にジャンプ	－

命令の形式は次のとおりとする。

LOAD / ADD / STORE　Rn , adr [, Rx] ：

定数adrはアドレスを示す。Rxは指標レジスタである（省略可能）。Rxを指定した場合の実効アドレスは，定数adrとRxの内容を加算した値が示す番地とする。

LOADは，実効アドレスが示す番地に格納されている内容を，レジスタRnに設定する。

ADDは，レジスタRnの内容に実効アドレスが示す番地に格納されている内容を加えて，レジスタRnに設定する。

STOREは，レジスタRnの内容を実効アドレスが示す番地に格納する。

ADDR　Rn , Rm ：レジスタRnの内容にレジスタRmの内容を加えて，レジスタRnに設定する。

JNM　adr ：直前の演算結果が0以上ならばadr番地へジャンプする。

設問1　次の(1)～(6)に従って，1000番地のLOAD命令を実行した直後，及び1006番地のSTORE命令を最初に実行した直後のディレクトリ部の内容を，図3と図4に示す。［　　］に入れる正しい答えを，解答群の中から選べ。

(1)　参照ブロックがキャッシュ内にある場合は，キャッシュ内のデータが使用される。

(2)　参照ブロックがキャッシュ内にない場合（以下，ミスという）は，参照ブロックが主記憶からキャッシュに読み込まれ，対応するディレクトリのフラグの内容は0に初期化される。

(3)　CPUが参照ブロックに対してSTORE命令を実行した場合は，対応するディレクトリのフラグの内容は1に変わる。

(4)　ミスが起こったときにキャッシュ内に未使用のバッファがある場合は，未使用のバッファの中で最も番号が小さいバッファに参照ブロックを読み込み，順位を更新する。

(5)　ミスが起こったときにキャッシュ内に未使用のバッファがない場合は，次のキャッシュ更新ロジックを用いる。

　キャッシュ内に最も古くから存在するブロックが格納されているバッファに，参照ブロックを読み込み，順位を更新する。ただし，対応するディレクトリのフラグの内容が1だったときは，そのブロックを主記憶に書き戻してから，参照ブロックを読み込み，順位を更新する。

(6)　プログラム実行開始時は，キャッシュ内にデータが入っていないものとする。

	ブロック番号	順位	フラグ
ディレクトリ1	a	1	0
ディレクトリ2	0	0	0
ディレクトリ3	0	0	0

図3　1000番地のLOAD命令を実行した直後のディレクトリ部

	ブロック番号	順位	フラグ
ディレクトリ1	b	2	0
ディレクトリ2	44	3	1
ディレクトリ3	c	1	0

図4　1006番地のSTORE命令を最初に実行した直後のディレクトリ部

解答群

ア　0　　イ　41　　ウ　42

エ　43　　オ　44　　カ　45

設問2　設問1の(5)のキャッシュ更新ロジックを，次に示す新しいキャッシュ更新ロジックに変えた場合の，ディレクトリ部のブロック番号とフラグの内容の変化を，表2に示す。［　　］に入れる正しい答えを，解答群の中から選べ。

〔新しいキャッシュ更新ロジック〕

参照されていない時間が最も長いブロックが格納されているバッファに，参照ブロックを読み込む。

なお，この更新ロジックでは，順位の付け方も変更されていて，キャッシュ内で参照されていない時間が最も長いブロックから順に1，2，3と番号が振られる。

表2　ディレクトリ部の内容の変化

	実行命令の番地	ディレクトリ1		ディレクトリ2		ディレクトリ3	
		ブロック番号	フラグ	ブロック番号	フラグ	ブロック番号	フラグ
	1000	45	0	0	0	0	0
	1001	45	0	0	0	0	0
1回目のループ	1002	45	0	41	0	0	0
	1003	45	0	41	0	42	0
	1004	43	0	41	0	42	0
	1005	43	0	41	1	42	0
	1006			d			
	1007						
	1008						
2回目のループ	1002					e	
	1003	f					
	1004						
	1005						
	1006						
	1007						
	1008						

注　網掛けの部分は，表示していない。

解答群

	ブロック番号	フラグ
ア	41	0
イ	41	1
ウ	42	0
エ	42	1
オ	43	0
カ	43	1
キ	44	0
ク	44	1
ケ	45	0
コ	45	1

問2　コンパイラの処理内容に関する次の記述を読んで，設問1～4に答えよ。

コンパイラとは，プログラム言語で記述された原始プログラムを翻訳して目的プログラムを生成するためのソフトウェアである。コンパイラの処理過程において，構文解析は字句解析が出力する字句を読み込みながら構文木を生成し，その字句の列が文法で許されているかどうかを解析する。

設問1　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。

2項演算子から成る式の構文は，2分木で表現される。2分木から目的プログラムを生成するには，2分木を深さ優先で探索しながら，帰り掛けに節の演算子を評価する。式の構文木と探索順序の例を図に示す。

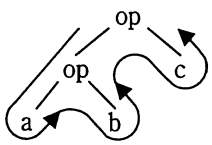


図　構文木と探索順序の例

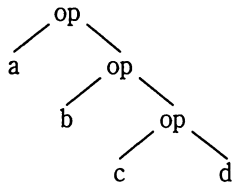
探索の結果，図の構文木の例は，aとbに対して演算opを施し，その結果とcに対して演算opを施すと解釈される。

括弧を含む式では，演算の優先度は，括弧内の優先度が高く，それ以外は左から右に式を評価する。このとき，次の式に対する構文木は［　　］である。

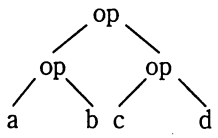
式：　a op (b op c) op d

解答群

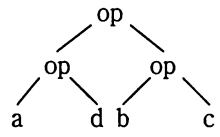
ア



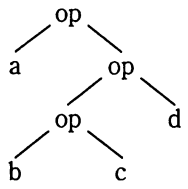
イ



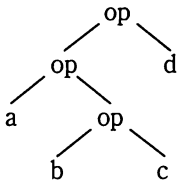
ウ



エ



オ



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。

プログラム言語の文法は，構文規則で規定される。式の構文規則では式の構文を規定しているだけでなく，演算子の優先順位や結合規則も規定している。例として，優先順位が異なる演算子 op1，op2 と括弧を用いた式の構文規則を考える。

〔式の構文規則〕

(1)　式　→　項 | 式 op1 項

(2)　項　→　因子 | 項 op2 因子

(3)　因子 →　名前 | (式)

(4)　名前 →　a | b | c | d | e

“→”は，左側の構文要素が右側で定義されることを示す。

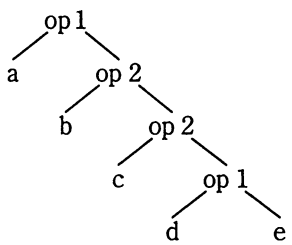
“|”は，“又は”を意味する。

深さ優先で探索すると仮定すれば，与えられた式に含まれる演算子op1とop2の演算順序は，〔式の構文規則〕から生成可能な構文木で表現できる。例えば，次の式に対して〔式の構文規則〕を適用した構文木は［　　］である。

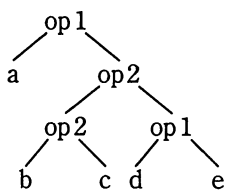
式：　a op1 b op2 c op2 (d op1 e)

解答群

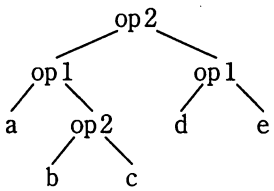
ア



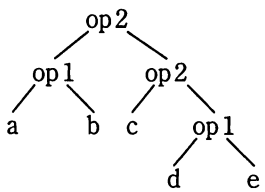
イ



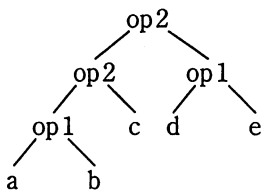
ウ



エ



オ



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。

構文解析した結果は，構文木で表現する以外に，2分木と等価な表現の後置表記法（逆ポーランド表記法）で表現できる。後置表記法では，演算で使用する二つのオペランドを演算子の前に記述する。そして，後置表記法は，構文木を探索して得られる演算順序を表現したものであると考えることができる。例えば，設問1の図の例で，演算子opを加算＋とした場合，後置表記法ではａｂ＋ｃ＋となる。これは，aとbを加算し，その結果とcを加算することを表している。

後置表記法を用いて式ａ×ｂ＋ｃ×ｄ＋ｅを表現すると［　　］になる。ここで，加算＋と乗算×は，それぞれ設問2の演算子op1とop2に対応し，演算の優先順位や結合規則は，〔式の構文規則〕に従うものとする。

解答群

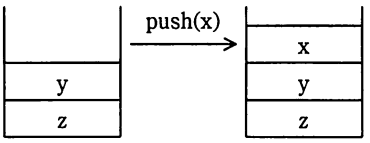
ア　ａｂｃｄ×＋ｅ＋　　イ　ａｂｃ×＋ｄ×ｅ＋

ウ　ａｂ×ｃｄ×＋ｅ＋　　エ　ａｂ×ｃ＋ｄ×ｅ＋

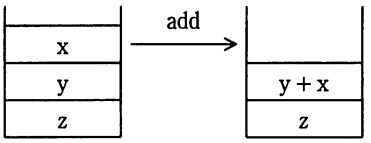
設問4　次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。

後置表記法で表現された式は，スタックを使って左から右に評価することができる。式 a + b + c × d を後置表記法に変換して評価する場合，スタックの操作順序は［　　］である。ここで，スタックを操作する命令として次の種類がある。また，加算 + と乗算 × は，それぞれ設問2の演算子 op1 と op2 に対応し，演算の優先順位や結合規則は，〔式の構文規則〕に従うものとする。

push(x)：スタックにxをプッシュする。



add：スタックからオペランドを二つポップして加算し，その結果をスタックにプッシュする。



mul：スタックからオペランドを二つポップして乗算し，その結果をスタックにプッシュする。

解答群

ア　push(a) → add → push(b) → add → push(c) → mul → push(d)

イ　push(a) → push(b) → add → push(c) → mul → push(d) → add

ウ　push(a) → push(b) → add → push(c) → push(d) → mul → add

エ　push(a) → push(b) → push(c) → push(d) → add → add → mul

問3　関係データベースに関する次の記述を読んで，設問1～4に答えよ。

ある中学校では，これまで表計算ソフトを使用して管理していた試験の成績をデータベース化することになった。この中学校に在籍している，又は過去に在籍したことがある生徒及び教員の情報を管理するデータベースは既に運用中であり，これらと連携させて運用する。

現在運用中のデータベースの構造は図1のとおりである。下線付きの項目は主キーを表す。

生徒表

生徒番号	氏名	住所	電話番号

履歴表

生徒番号	年度	学年	クラス	教員番号

注　教員番号には，担任教員を識別する教員番号が格納されている。

教員表

教員番号	氏名	住所	電話番号

図1　現在運用中のデータベースの構造

この中学校は，1年度が3期に分かれており，それぞれの期中に中間試験と期末試験を実施している。試験科目は，国語，数学，英語，理科，社会の5科目である。また，試験は学年別，科目別に統一した問題で行われている。

設問1　成績を管理するデータベースの構造について，案A及び案Bが提案された。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。

案A　生徒番号，年度，試験IDを主キーとして管理する。

成績表

生徒番号	年度	試験ID	国語	数学	英語	理科	社会

注　国語，数学，英語，理科，社会には，それぞれの科目の得点が格納される。

試験表

試験ID	試験名

注　試験名には，試験の名称（1学期中間，1学期期末，2学期中間，2学期期末，3学期中間，3学期期末）が格納される。

案B　生徒番号，年度，試験ID，科目IDを主キーとして管理する。

成績表

生徒番号	年度	試験ID	科目ID	得点

注　得点には，当該試験の科目IDで識別される科目の得点が格納される。

試験表

試験ID	試験名

注　試験名には，試験の名称（1学期中間，1学期期末，2学期中間，2学期期末，3学期中間，3学期期末）が格納される。

科目表

科目ID	科目名

注　科目名には，科目の名称（国語，数学，英語，理科，社会）が格納される。

データベースを設計する上で，拡張性の考慮は重要である。その点，案Bは案Aと比較して［　a　］に対して柔軟に対応することができる。その反面，表の増加に伴い，データを検索する際のSQL文が複雑になりやすく，また，1人の生徒の1年間の成績を格納するために必要な成績表のレコード件数は，案Aでは［　b　］件で済むが，案Bでは［　c　］件となる。扱うデータの総量にもよるが，長期の運用を考えた場合，これらの点はデータ操作時の性能にも影響を与えるおそれがある。

データベースの設計においては，運用時の利用形態を想定した上で，操作性，拡張性，実行性能などを多角的に評価する必要がある。この中学校では操作性を重視して案Aを採用した。

aに関する解答群

ア　学期の増減　　イ　試験回数の増減

ウ　試験科目の増減　　エ　生徒の増減

b，cに関する解答群

ア　2　　イ　4　　ウ　6　　エ　12

オ　24　　カ　30　　キ　48　　ク　60

設問2　2009年度1学年の2学期中間試験を対象に，生徒ごとの全科目合計点を求め，降順に整列して表示する。次のSQL文の［　　］に入れる正しい答えを，解答群の中から選べ。

```
SELECT 生徒表.氏名, クラス, 国語 + 数学 + 英語 + 理科 + 社会
   FROM 生徒表, 履歴表, 成績表, 試験表
   WHERE 生徒表.生徒番号 = 履歴表.生徒番号
         AND 履歴表.生徒番号 = 成績表.生徒番号
         AND 成績表.試験 ID = 試験表.試験 ID
         AND [                                    ]
   ORDER BY 国語 + 数学 + 英語 + 理科 + 社会 DESC
```

解答群

ア
```
履歴表.年度 = 2009 AND 履歴表.学年 = 1
AND 試験表.試験名 = '2学期中間'
```

イ
```
履歴表.年度 = 2009 AND 履歴表.学年 = 1
AND 試験表.試験名 = '2学期中間' AND 成績表.年度 = 2009
```

ウ
```
履歴表.年度 = 2009 AND 履歴表.学年 = 1
AND 試験表.試験 ID = (SELECT 試験 ID FROM 試験表
                      WHERE 試験表.試験名 = '2 学期中間')
```

エ
```
履歴表.年度 = 2009 AND 履歴表.学年 = 1
AND 試験表.試験 ID = ANY(SELECT 試験 ID FROM 試験表
                      WHERE 試験表.試験名 = '2 学期中間')
```

設問3　2008年度1学年を対象に，1年間に実施したすべての試験の各科目の平均点をクラスごとに求め，担任教員の氏名とともに表示する。次のSQL文の［　　］に入れる正しい答えを，解答群の中から選べ。

```
SELECT クラス，教員表.氏名，AVG(国語)，AVG(数学)，AVG(英語)，
       AVG(理科)，AVG(社会)
  FROM 履歴表，成績表，教員表
  WHERE [                                ]
  ORDER BY クラス
```

解答群

ア
```
履歴表.生徒番号 = 成績表.生徒番号
AND 履歴表.教員番号 = 教員表.教員番号
AND 履歴表.年度 = 2008 AND 履歴表.学年 = 1
GROUP BY クラス，教員表.氏名
```

イ
```
履歴表.生徒番号 = 成績表.生徒番号
AND 履歴表.教員番号 = 教員表.教員番号
AND 履歴表.年度 = 2008 AND 履歴表.学年 = 1
AND 成績表.年度 = 2008
GROUP BY クラス，教員表.氏名
```

ウ
```
履歴表.生徒番号 = 成績表.生徒番号
AND 履歴表.年度 = 2008 AND 履歴表.学年 = 1
AND 成績表.試験ID = ANY(SELECT 試験ID FROM 試験表)
GROUP BY クラス，教員表.氏名
```

エ
```
履歴表.生徒番号 = 成績表.生徒番号
AND 履歴表.年度 = 2008 AND 履歴表.学年 = 1
AND 履歴表.教員番号 = ANY(SELECT 教員番号 FROM 教員表)
GROUP BY クラス，教員表.氏名
```

設問4　生徒“情報太郎”の成績を，試験IDの昇順に表示する。次のSQL文の［　　］に入れる正しい答えを，解答群の中から選べ。

なお，生徒“情報太郎”が複数人いた場合は，過去に在籍したことがある生徒も含めて生徒番号の昇順に表示する。

```
SELECT 生徒表.生徒番号，成績表.年度，試験名，国語，数学，英語，
       理科，社会
  FROM 生徒表，成績表，試験表
  WHERE [                              ]
  ORDER BY 生徒表.生徒番号，成績表.年度，試験表.試験ID
```

解答群

ア
```
生徒表.生徒番号 = 成績表.生徒番号
AND 成績表.試験ID = 試験表.試験ID
AND 生徒表.氏名='情報太郎'
```

イ
```
生徒表.生徒番号 = 成績表.生徒番号
AND 成績表.試験ID = 試験表.試験ID
AND 生徒表.氏名='情報太郎'
AND 生徒表.生徒番号 = (SELECT 生徒番号 FROM 生徒表)
```

ウ
```
成績表.生徒番号 = (SELECT 生徒番号 FROM 生徒表 WHERE 氏名='情報太郎')
GROUP BY 試験表.試験ID
```

エ
```
成績表.生徒番号 = (SELECT 生徒番号 FROM 生徒表 WHERE 氏名='情報太郎')
GROUP BY 成績表.年度
```

問4　動画のストリーミングサーバの設置計画に関する次の記述を読んで，設問1，2に答えよ。

S社は表1に示す要件で動画のストリーミングサーバ（以下，サーバという）の設置を計画している。

表1　動画ストリーミングの要件

動画のビット速度	0.5Mビット／秒
クライアントから1分当たりに要求される動画の本数	15本／分
動画1本の平均再生時間（再生時間は指数分布に従い，動画の再生時間と送信時間は等しいものとする）	4分
サーバに格納する動画の本数	1,000本
クライアントからの要求が待たされる確率	10％以下

設問1　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。

サーバに格納される動画のデータ量は約［ a ］Gバイトであり，サーバから動画を送信するときに要求されるビット速度は，平均で［ b ］Mビット／秒である。ここで，1Gバイト＝10^9バイト，1Mビット＝10^6ビットとする。

aに関する解答群

ア　1.5　　イ　12　　ウ　15　　エ　30　　オ　120

bに関する解答群

ア　7.5　　イ　15　　ウ　30　　エ　60　　オ　75

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。

サーバは複数の動画を同時に送信可能である。また，クライアントからの要求はランダムに発生し，サーバが同時に送信可能な本数を上回る要求は待ち行列に入れられ，順次処理されるものとする。

このとき，クライアントの要求が待ち行列に入る確率Pは，待ち行列理論によって，二つの引数をとる関数fで計算することができる。引数の一つは，サーバが同時に送信可能な動画の本数nであり，もう一つは，要求されるトラフィックTである。この場合，Tは，［ c ］で，次の式で計算できる。

クライアントからの1分当たりの要求本数 × 動画1本の平均再生時間（分）

fを使いP，n，Tの関係を求めたものを，表2に示す。表2の網掛けの部分がTである。表2によれば，要件のとおり，要求が待たされる確率を10％以下にするためには，サーバが最低［ d ］本の動画を同時に送信できなければならないことが分かる。

また，仮に，サーバが86本の動画を同時に送信できる能力をもつとき，要求が待たされる確率はおよそ［ e ］％である。

なお，回線容量は十分にあり，通信の遅延はないものとする。

表2　P，n，Tの関係（一部）

		P（％）										
		0.1	0.2	0.4	0.8	2	4	7	10	15	20	30
n（本）	70	46.8	48.2	49.6	51.2	53.5	55.5	57.3	58.6	60.1	61.3	63.1
	71	47.6	49.0	50.5	52.0	54.4	56.4	58.2	59.5	61.0	62.2	64.1
	72	48.5	49.8	51.3	52.9	55.3	57.3	59.1	60.4	62.0	63.2	65.1
	73	49.3	50.7	52.1	53.8	56.1	58.2	60.0	61.3	62.9	64.1	66.0
	74	50.1	51.5	53.0	54.6	57.0	59.1	60.9	62.2	63.8	65.1	66.9
	75	50.9	52.3	53.8	55.5	57.9	60.0	61.9	63.2	64.8	66.0	67.9
	86	60.0	61.5	63.2	65.0	67.6	69.9	71.9	73.3	75.0	76.3	78.4

cに関する解答群

ア　1分当たりに送信される動画の本数

イ　1分当たりの動画の延べ再生時間

ウ　ネットワーク帯域の最大占有率

エ　ネットワークの平均消費帯域幅

dに関する解答群

ア　71　　イ　72　　ウ　73　　エ　74　　オ　75

eに関する解答群

ア　0.1　　イ　0.2　　ウ　0.4　　エ　0.8　　オ　4

問5　配達サービス管理システムに関する次の記述を読んで，設問1～3に答えよ。

家具を販売する小売業のF社では，顧客が購入した家具を，顧客が希望した日に配達するサービスを実施している。F社では，この配達サービスについて，家具の配達を受け付けてから配達が完了するまでの状況を管理する，配達サービス管理システム（以下，管理システムという）を構築することになった。

〔配達サービスの仕組み〕

(1)　配達サービスとは，F社で家具を購入した顧客に対するサービスであり，購入した家具を，指定した配達先に，希望した日に配達する。

(2)　F社では家具それぞれに固有の家具番号を付与している。

(3)　配達サービスは，F社の配達窓口で受付登録をすることによって開始される。配達サービスを受けたい顧客は，F社の配達窓口担当者（以下，担当者という）に配達を依頼する。

(4)　担当者は，受付登録時に，配達先の郵便番号，住所，氏名及び家具番号を，依頼受付画面から管理システムに登録する。1回の受付登録では，配達先は1か所に限られ，家具は10個まで登録できる。受付登録後，画面には，その配達先住所に配達可能な配達業者（以下，業者という）の社名と受付登録した日（以下，受付日という）の翌々日以降で配達可能な日付（以下，配達可能日という）の組合せが表示される。

(5)　担当者は，配達可能日を顧客に伝える。顧客は，その中から配達を希望する日付（以下，配達予定日という）を指定する。

(6)　担当者は指定された配達予定日を，依頼受付画面から管理システムに登録する。登録が完了すると，依頼受付画面に予約番号が表示される。担当者は，予約番号を顧客に伝え，受付が完了する。

(7)　担当者は，配達予定日の前日に，配達先情報や配達家具の情報を記載した配達依頼リストを，管理システムから出力する。

(8)　業者は，配達予定日の前日に，配達すべき家具と配達依頼リストを担当者から受け取る。配達予定日に配達先住所に家具を届け，配達完了を担当者に連絡する。

(9)　配達完了の連絡を受けた担当者は，配達完了日を管理システムに登録する。

配達サービスにおける情報の流れを，図1に示す。

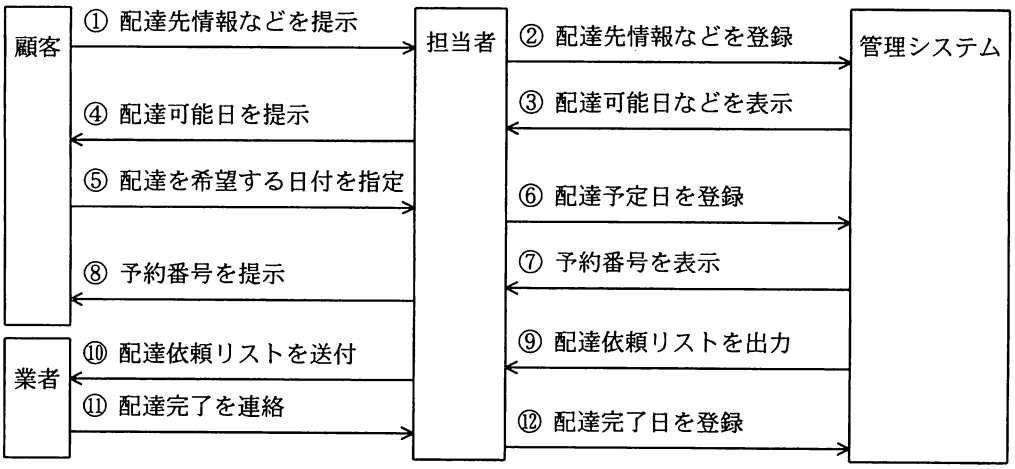


図1　配達サービスにおける情報の流れ

〔管理システムの概要〕

(1)　管理システムは，端末とサーバで構成される。端末には担当者が用いる画面インタフェースを構築し，サーバでは，配達に関連する情報を，受付ファイル，業者ファイル，郵便番号別業者ファイル及び予約ファイルとして保持する。

(2)　受付ファイルには，顧客から配達の依頼があったときに付与される一意の受付番号とともに，住所など配達に必要な顧客の情報を登録する。受付ファイルのレコード様式を，図2に示す。

受付番号	受付日	配達先 郵便番号	配達先 住所	配達先 氏名	家具の個数	家具番号 1	…	家具番号 10

図2　受付ファイルのレコード様式

(3)　業者ファイルには，業者コード，業者名及び業者連絡先が登録されている。業者ファイルのレコード様式を，図3に示す。

業者コード	業者名	業者連絡先

図3　業者ファイルのレコード様式

(4) 郵便番号別業者ファイルには，ある配達可能日にその郵便番号が示す地域に家具を配達可能な業者と，その業者がその日に配達可能な件数の上限値を登録する。郵便番号別業者ファイルのレコード様式を，図4に示す。ここで，一つの業者は一つの地域だけを対象とする。

郵便番号	配達可能日	業者コード	配達先件数の上限値

図4　郵便番号別業者ファイルのレコード様式

(5) 予約ファイルには，配達予定日が確定したときに付与される一意の予約番号とともに，業者コード，配達予定日及び受付番号を登録する。配達完了日には，配達完了までは空白が格納されており，配達が完了するとその日付が格納される。予約ファイルのレコード様式を，図5に示す。

予約番号	業者コード	配達予定日	受付番号	配達完了日

図5　予約ファイルのレコード様式

(6) 配達依頼リストは，業者が家具を配達する際に用いるリストであり，業者名，顧客への配達予定日，その業者の配達先件数，配達先情報，予約番号，家具の個数及び家具番号が記載されている。顧客への配達予定日の前日に，業者コード順に管理システムから出力される。配達依頼リストの例を，図6に示す。

配達依頼リスト　　作成日 2010-04-18

業者名　H社　　配達予定日 2010-04-19
配達先件数　3

No.	配達先情報	予約番号	家具の個数	家具番号
1	東京都○○区△△1-2-3 情報　花子	1234	3	3456 4567 5999
2	東京都○○区△△12-34 試験　太郎	1023	2	0509 1006
3	⋮	⋮	⋮	⋮

図6　配達依頼リストの例

設問1　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。

依頼を受けた配達先住所に対して，その配達先住所へ配達可能な業者と配達可能日の組合せを表示する機能を考える。

まず，郵便番号別業者ファイルから，受付時に入力した配達先郵便番号と一致し，かつ，配達可能日が［　a　］レコードを取り出す。

次に，郵便番号別業者ファイルから取り出したレコード1件ごとに予約ファイルを検索する。検索の条件は，業者コードが等しく，かつ，予約ファイルの配達予定日が［　b　］ことである。この検索によって，その業者で，配達予定日が確定している配達先の件数が判明する。この確定している配達先の件数が，配達先件数の上限値に達していなければ，その業者とその日付は配達可能と判定でき，表示の対象とする。

解答群

ア　受付日の翌々日以降となる
イ　受付日以降となる
ウ　受付日と同じ
エ　受付日以前となる
オ　配達可能日以降となる
カ　配達可能日と同じ
キ　配達可能日と同じでない
ク　配達可能日以前となる
ケ　配達予定日と同じ
コ　配達予定日と同じでない

設問2　図7は，配達依頼リストを作成する処理の流れである。図7の処理Fで作成する集計結果ファイルのレコード様式を，図8に示す。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。

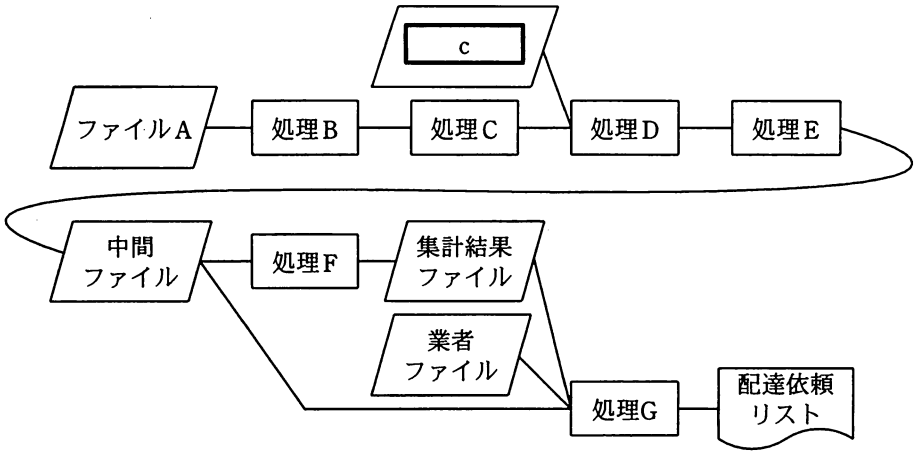


図7　配達依頼リストを作成する処理の流れ

業者コード	配達先件数

図8　集計結果ファイルのレコード様式

処理B：ファイルAから，配達予定日が翌日のデータを取り出す。

処理C：処理Bの処理結果を，［　d　］で並べ替える。

処理D：処理Cの処理結果と［　c　］の内容を，［　d　］をキーとして突き合わせる。一致した［　c　］のレコード内容と合わせて出力する。

処理E：処理Dの処理結果を，業者コードで並べ替え，中間ファイルを作成する。

処理F：処理Eの処理結果である中間ファイルの，業者コードごとの［　e　］を数え，その値を配達先件数に入れ，集計結果ファイルを作成する。

処理G：中間ファイル，集計結果ファイル及び業者ファイルから，配達依頼リストを作成する。

cに関する解答群

ア　受付ファイル　　イ　郵便番号別業者ファイル　　ウ　予約ファイル

dに関する解答群

ア　受付番号　　イ　受付日　　ウ　配達可能日

エ　配達予定日　　オ　郵便番号　　カ　予約番号

eに関する解答群

ア　家具の個数　　イ　配達先件数の上限値　　ウ　レコード件数

設問3　担当者は，業者から配達完了の連絡を受けて配達完了日を登録する。配達完了の登録に必要な“配達完了日”以外の項目として正しい答えを，解答群の中から選べ。

解答群

ア　受付日，郵便番号

イ　業者コード，郵便番号

ウ　業者コード，郵便番号，配達可能日

エ　配達予定日

オ　予約番号

問6　プロジェクトにおける品質管理に関する次の記述を読んで，設問1，2に答えよ。

システムインテグレータX社は，Y社のシステム開発を3年前から担当している。初年度の新規開発が終了後，半年ごとにシステムの機能拡張を継続的に行っている。今年度は比較的大規模なシステム開発（以下，プロジェクトPという）をすることになり，表1のとおりに開発体制を変更（新規メンバを，グループG1に1名，G2に2名追加）した。プロジェクトPでは，開発体制の変更に伴うシステムの品質低下を防止するために，従来以上にプロジェクトにおける品質管理を徹底することにした。

表1　Y社向けシステムの開発体制の変更

グループ名	G1	G2	G3
要員数（グループリーダを含む）	4名→5名	3名→5名	3名
担当サブシステム	S1	S2	S3

注　“→”は要員数の変更を表す。

過去のプロジェクトで蓄積された品質データ（バグ件数，バグ摘出率など）は，次のプロジェクトの品質管理に役立てることができる。蓄積された品質データを基に，新規プロジェクトの目標値を設定し，各開発工程での実績値との差異を分析する。差異が生じた場合には，その原因を見いだして，品質向上のための施策の実行，目標値の見直しなど，適切に対処することを繰り返す。これによって，開発工程が進むにつれて品質管理の精度が向上し，システムの品質が確保される。

(1)　プロジェクトPの開発工程は，設計工程，製造工程，単体テスト工程及び結合テスト工程の四つの工程から成る。

(2)　設計工程の開始時点では，Y社のシステム開発における過去のプロジェクトでの品質データの実績値を基に，バグ総件数を予測し，工程ごとのバグ摘出率の目標値を設定する。設計工程の終了時には，バグ摘出件数の予測値と実績値の比較・分析を行い，バグ総件数の予測値を見直して，以降の工程での残存バグ件数を予測する。製造工程，単体テスト工程及び結合テスト工程の終了時に，この予測を繰り返す。

(3) 設計工程又は製造工程で生じた誤り（バグ）をテスト工程で発見して修正する場合には多くの工数を要するので，開発の生産性及びシステムの品質向上には，バグの早期発見が重要である。

X社では，設計工程及び製造工程でのバグ摘出率の向上を目指し，品質管理の全社目標を，“設計工程及び製造工程でのバグ摘出率65％以上”に設定している。各グループのこれまでの品質データに今回の開発体制の変更の影響を加味した結果と全社目標を基に，プロジェクトPの設計工程開始時点での工程ごとの目標バグ摘出率を表2のように設定した。

表2　各工程での目標バグ摘出率

工程	設計	製造	単体テスト	結合テスト
目標バグ摘出率（％）	30	35	20	15

(4) 各サブシステムとも，設計工程は計画どおりの期間で終了した。設計工程でのバグ摘出件数の実績値及び算出したバグ摘出率は，表3のとおりであった。

表3　設計工程でのバグ摘出件数（実績値）及びバグ摘出率

サブシステム名	S1	S2	S3
バグ摘出件数（件）	280	175	112
バグ摘出率（％）	35	25	28

設計工程でのバグ摘出率（％）は，次の式で算出する。

$$\text{設計工程でのバグ摘出率} = \frac{\text{設計工程でのバグ摘出件数（実績値）}}{\text{全工程でのバグ総件数（予測値）}} \times 100$$

(5) 製造工程でのバグ摘出件数の実績値は，表4のとおりであった。バグ摘出件数の予測値と実績値を比較・分析した結果，製造工程の終了時に各サブシステムでの残存バグ件数を表4のとおりに予測した。

表4 製造工程でのバグ摘出件数（実績値）と残存バグ件数の予測値

サブシステム名	S1	S2	S3
バグ摘出件数（件）	210	170	143
残存バグ件数の予測値（件）	262	170	137

(6) テスト工程（単体テスト工程及び結合テスト工程）でのバグ摘出件数の実績値は，表5のとおりであった。

表5 テスト工程でのバグ摘出件数（実績値）

サブシステム名		S1	S2	S3
バグ摘出件数	単体テスト工程（件）	160	100	90
	結合テスト工程（件）	100	60	45

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。

プロジェクトマネージャのM課長は，表3の結果を見て，グループG1とG2の設計品質に対して表6に示す疑問をもち，グループリーダに各サブシステムの設計品質についての見解を説明させた。グループリーダからの回答内容は，表7のとおりであった。M課長は，この説明に納得して，各グループに製造工程への着手を指示した。

表6 設計品質に対するM課長の疑問点

グループ	設計品質への疑問内容
G1	設計工程が終了した時点で算出したバグ摘出率（35％）が目標値（30％）よりも高くなっている。担当サブシステムの難度が予測よりも高かったのか，新規メンバのスキルに問題があったのではないか。
G2	新規メンバを2名も追加したにもかかわらず，バグ摘出率（25％）が目標値（30％）よりも低くなっている。設計レビューが適切に実施されなかったのではないか。

表7　M課長の疑問点に対する回答

グループ	グループリーダからの回答内容
G1	[a]と[b]が要因として考えられる。今回，設計品質の向上のために，これまでよりも設計レビューを強化した。これらのことから，前述の要因によって発生したと考えられるバグを含めて全体のバグ摘出件数が増加した。よって，バグ摘出率が目標値を上回っているが，バグの改修は終了しており，品質を十分に確保した。
G2	[c]によって，設計の再利用率が計画値よりも高まった。さらに，[d]によって，メンバの生産性が計画値よりも高まった。これらのことから，バグ摘出件数が減少した。よって，バグ摘出率が目標値を下回っているが，設計レビューは適切に実施しており，品質を十分に確保した。

解答群

ア　過去のシステムの機能拡張で改造した機能と類似しているモジュールが予想以上に多かったこと

イ　新規メンバが要求仕様を完全に理解していなかったためにバグが発生したこと

ウ　新規メンバの１人が，類似システムの開発に関して，既存メンバを上回る経験を有していたこと

エ　設計の難度が高いモジュールが予測以上に多かったこと

オ　他システムの保守対応など緊急の割込み業務が多発して工数不足だったこと

設問2　テスト工程での品質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。

なお，解答群の数値は小数点以下を四捨五入した値である。

表4及び表5の結果を見たM課長は，S2に関して，製造工程終了時の残存バグ件数の予測値が170件であるのに対し，テスト工程でのバグ摘出件数は160件であり，10件の摘出不足があることから，まだバグが残っており，テスト不足ではないのかとの疑問をもった。

そこで，S2に対して追加テストが必要かどうかを見極めるために，テスト工程でのバグ成長曲線を確認することにした。テスト工程における検査項目の完了数とバグ摘出件数との関係を表すグラフが［　e　］であることから，S2のテスト工程での品質は十分に安定していると評価した。また，S2の設計工程においても，表7の回答内容から品質の良さが想定される。さらに，最終的な製造工程のバグ摘出率は［　f　］%であり，製造工程での品質確保も十分であると考えられる。最終的な製造工程のバグ摘出率（%）は，次の式で算出する。

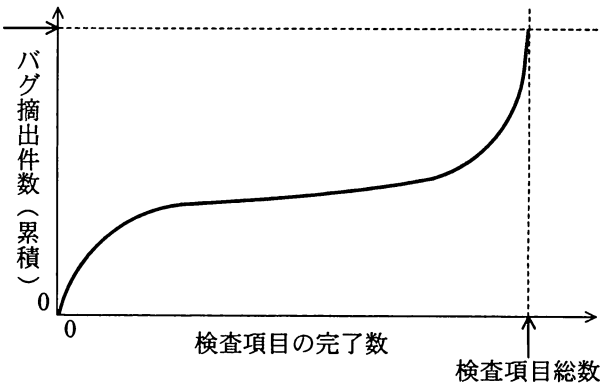
$$\text{製造工程でのバグ摘出率} = \frac{\text{製造工程でのバグ摘出件数（実績値）}}{\text{全工程でのバグ総件数（実績値）}} \times 100$$

これらのことから，S2のバグ総件数の実績値は当初の予測値よりも少ない値であるが，バグは残っていないと判断してテストを完了した。

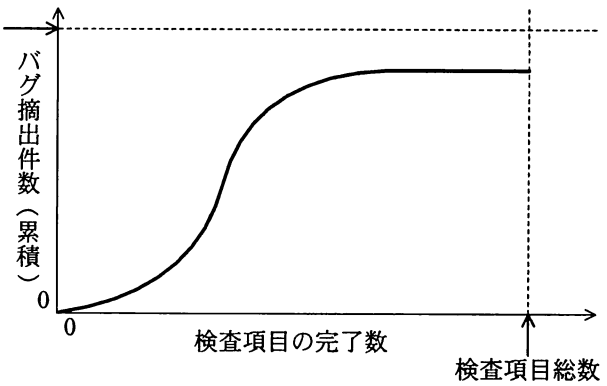
その後，Y社へシステムをリリースして3か月が経過したが，Y社からの不具合の報告はなかった。この結果から，最終的に設計工程及び製造工程でのバグ摘出率の合算値が最も高かったのは，グループ［　g　］であり，バグ摘出率の合算値は［　h　］%である。

eに関する解答群

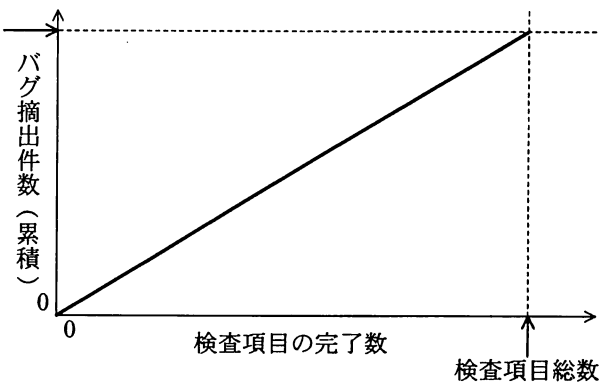
ア



イ



ウ



fに関する解答群

ア 30　　イ 32　　ウ 34　　エ 36　　オ 38

gに関する解答群

ア G1　　イ G2　　ウ G3

hに関する解答群

ア 66　　イ 68　　ウ 70　　エ 72　　オ 74

問7　事業の分析に関する次の記述を読んで，設問1～3に答えよ。

SWOT分析は，企業又は組織の強み，弱み，機会及び脅威を分析する手法であり，事業戦略を策定するために利用される。ある地域で食料品の生産及び販売をしているA社の企画課では，自社の健康飲料事業の戦略を策定するためにこのSWOT分析を使って，内部環境や外部環境から導かれる課題を考察することにした。

設問1　A社の健康飲料事業の強みと弱み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。

企画課のK氏は，上司から，自社の健康飲料事業の，競合他社と比較した強みと弱みを列挙し，それらを資産に関するものと業務プロセスに関するものとに分類するように指示を受けた。

K氏は，A社の健康飲料事業の強みと弱みについて分析し，表1にまとめた。

表1　A社の健康飲料事業の強みと弱み

強み	①　生産業務を効率よく行っていることによって，売上総利益率を業界での上位に保っている。 ②　知的財産の件数が，業界平均を上回っている。 ③　就業環境の改善を図るため，従業員満足度に関する調査・分析業務を10年以上継続しており，その結果を業務改善に結び付けるノウハウを蓄積している。 ④　売れ筋製品で使用している特殊な原料について，1社しかない供給業者との独占購入権を保有している。
弱み	①　情報システムのアプリケーション開発プロセスの効率が低い。 ②　顧客の製品ブランドに対する好感度が，競合他社と比較して低い。 ③　本社スタッフ部門による営業支援が十分ではない。 ④　物流工程が，周辺環境に配慮したプロセスとなっていない。

表1の強みのうち，業務プロセスに関する記述は，①と③である。②の知的財産の件数や④の独占購入権は健康飲料事業のもつ資産に関する強みといえる。しかし，K 氏の分析した強みの③については，［ a ］，上司の指示どおりになっていないので見直す必要がある。弱みのうち，資産に関する記述は，［ b ］である。

aに関する解答群

ア　A社の健康飲料事業の内部で効果を上げているだけで，競合他社と比較すると劣っていることが明らかであり

イ　A社の健康飲料事業の内部で効果を上げているものの，競合他社と比較した強みかどうかは不明確であり

ウ　競合他社と比較した強みであることは間違いないが，A社の健康飲料事業の内部で効果を上げているかは不明確であり

エ　競合他社と比較した強みであることは間違いないが，A社の健康飲料事業の内部では明らかに効果を上げておらず

bに関する解答群

ア　①	イ　②	ウ　③	エ　④
オ　①と②	カ　①と③	キ　①と④	ク　②と③
ケ　②と④	コ　③と④		

設問2　A社の健康飲料事業の機会と脅威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。

K氏は，A社の健康飲料事業に関するSWOT分析の機会と脅威を考察するための準備として，A社の事業地域における健康飲料業界の収益性について分析を行うように上司から指示を受けた。K氏は，ファイブフォース分析を使って，業界における，売り手の交渉力，買い手の交渉力，新規参入者の脅威，代替製品の脅威及び競争業者間の敵対関係の強さを分析して，業界の収益性や成長性を評価することにした。

例えば，新規参入者の脅威は，国の規制に守られている場合や，多大な設備投資が必要な業界の場合に低くなる。また，売り手の交渉力は，売り手の提供する製品やサービスが特殊ではなく，買い手が売り手を多数の候補の中から選べる場合に弱くなる。代替製品の脅威は，提供している製品やサービスに代わるものが，ほかにあまりない場合に低くなる。売り手や買い手の交渉力が弱いほど，また新規参入者や代替製品の脅威が低く競争業者間の敵対関係が弱いほど，業界の収益性は高くなりやすいと考えられている。

K氏は，A社の事業地域における健康飲料業界の分析を行って上司に報告した。図は，K氏と上司が完成させた分析結果である。

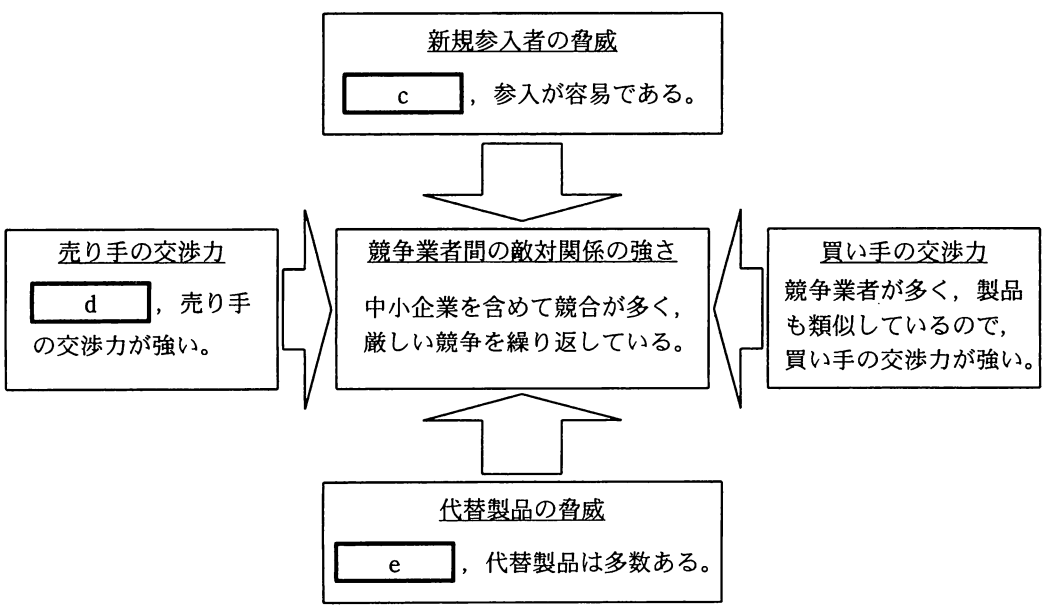


図　A社の事業地域における健康飲料業界の分析結果

解答群

ア　栄養補助食品などの競合製品が多く

イ　脅威となる競合製品はほとんどなく

ウ　供給業者の少ない固有の包装資材が多く

エ　供給業者の少ない固有の包装資材がなく

オ　消費者の健康意識レベルが高く

カ　消費者の健康意識レベルが低く

キ　特殊な設備が必要であり

ク　特殊な設備が不要であり

設問3　A社の健康飲料事業の収益性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。

K氏は，分析した結果から，A社の健康飲料事業の収益性は高くなりにくいと考えた。K氏は，A社の健康飲料事業を社内のほかの事業と比較するため，表2のデータを入手した。

表2　A社の各事業の収益性に関するデータ

単位 百万円

事業	売上	売上総利益	営業利益	経常利益
健康飲料事業	90	40	3	1
清涼飲料事業	240	90	10	11
加工食品事業	500	180	25	18

表2から，A社の健康飲料事業の収益性は，［ f ］ことが分かる。K氏はこれまでの分析結果から考察した収益性と，表2から得た収益性から，A社の健康飲料事業が単独で収益性を高めることは難しいと考えた。そこでK氏は，A社の清涼飲料事業や加工食品事業の強みを健康飲料事業に横展開することが必要ではないかと考え，［ g ］という取組みを考えた。

fに関する解答群

ア　売上総利益率，営業利益率，経常利益率のすべてで最も高い

イ　売上総利益率，営業利益率，経常利益率のすべてで最も低い

ウ　売上総利益率と営業利益率は最も高いが，経常利益率は最も低い

エ　売上総利益率は最も高いが，営業利益率と経常利益率は最も低い

オ　売上総利益率は最も低いが，営業利益率と経常利益率は最も高い

カ　営業利益率と経常利益率は最も高いが，売上総利益率は最も低い

キ　営業利益率は最も高いが，売上総利益率と経常利益率は最も低い

ク　経常利益率は最も高いが，売上総利益率と営業利益率は最も低い

gに関する解答群

ア　健康飲料事業から撤退する

イ　健康飲料事業の生産業務を清涼飲料事業や加工食品事業にも応用する

ウ　清涼飲料事業，加工食品事業及び健康飲料事業のそれぞれの強みとなる業務を社外に委託する

エ　清涼飲料事業や加工食品事業の営業ネットワークを健康飲料事業でも利用する

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〕

プログラム Sort は配列に格納された整数値のデータを再帰的に分割し，分割したデータの値の大小を比較しながら併合していくことでデータを昇順に整列するプログラムである。Sort は併合に副プログラム Merge を使用する。

(1) num 個（num ≧ 1）のデータを配列 list に格納して Sort を呼び出すと，整列された結果が配列 list に返却される。

(2) Sort では，次の手順で配列 list に格納された整数値のデータを整列する。

① 配列 list に格納されているデータを，先頭から num ÷ 2 個と残り num − num ÷ 2 個とに分割して，二つの配列 slist1 と slist2 に格納し，それぞれの配列に対して再帰的に Sort を呼び出す。ここで，配列 slist1 と slist2 の大きさは省略されているが，必要な領域は確保されている。この再帰的な呼出しは，引数で渡される配列 list のデータの個数が 1 になると終了する。

② Merge を使用し，二つの配列 slist1 と slist2 を併合して一つの配列 list にする。

(3) Merge では，次の手順で，整列済の二つの配列 slist1 と slist2 を併合し，整列した一つの配列 list を作成する。

① 配列 slist1 又は slist2 のどちらか一方の要素がなくなるまで，次の ② を繰り返す。

② 配列 slist1 と slist2 の要素を順に比較して，小さい方から順に配列 list に格納する。

③ 配列 slist1 又は slist2 の残った要素を配列 list に追加する。

(4) SortとMergeの引数の仕様を表1，2に示す。配列の添字は0から始まる。

表1　Sortの引数の仕様

引数名／返却値	データ型	入力／出力	意味
list[]	整数型	入力及び出力	データが格納されている1次元配列
num	整数型	入力	配列listのデータの個数

表2　Mergeの引数の仕様

引数名／返却値	データ型	入力／出力	意味
slist1[]	整数型	入力	整列済のデータが格納されている1次元配列
num1	整数型	入力	配列slist1のデータの個数
slist2[]	整数型	入力	整列済のデータが格納されている1次元配列
num2	整数型	入力	配列slist2のデータの個数
list[]	整数型	出力	併合したデータを格納する1次元配列

次のデータを例にして，整列処理の流れを図に示す。

配列listのデータ：5, 7, 4, 2, 3, 8, 1

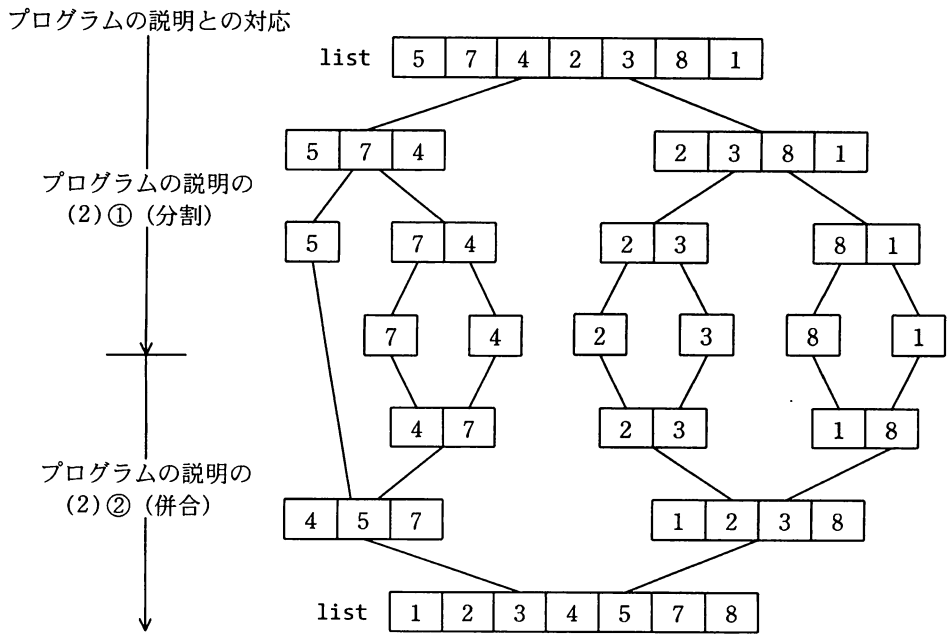


図　整列処理の流れ

〔プログラム〕

```
/* プログラム Sort */
○Sort(整数型 : list[], 整数型 : num)
○整数型 : i, num1, num2
○整数型 : slist1[], slist2[]                    /* 配列の宣言 */
    a
  ・ num1 ← num ÷ 2                             /* slist1 の要素数計算 */
  ・ num2 ← num − num1                          /* slist2 の要素数計算 */
  ■ i:0, i < num1, 1
    ・ slist1[i] ← list[i]
  ■

  ■ i:0, i < num2, 1
    ・ slist2[i] ← [ b ]
  ■
  ・ Sort(slist1,num1)
  ・ Sort(slist2,num2)
  ・ Merge(slist1, num1, slist2, num2, list)
                                                  ← α
/* プログラム Sort の終わり */

/* 副プログラム Merge */
○Merge(整数型 : slist1[], 整数型 : num1,
       整数型 : slist2[], 整数型 : num2,
       整数型 : list[] )
○整数型 : i, j
・ i ← 0
・ j ← 0
■  c
    slist1[i] < slist2[j]
    ・ list[i+j] ← slist1[i]
    ・ i ← i + 1
   ―
    ・ list[i+j] ← slist2[j]
    ・ j ← j + 1
■
┌──────────────────────────────┐
■ (i < num1) or (j < num2)     │
    i < num1                   │
    ・ list[i+j] ← slist1[i]   │
    ・ i ← i + 1               │
   ―                           │
    ・ list[i+j] ← slist2[j]   │
    ・ j ← j + 1               │
■                              │
└──────────────────────────────┘
                                                  ← β
/* 副プログラム Merge の終わり */
```

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　num ≧ 0　　イ　num ≧ 1

ウ　num ＞ 1　　エ　num ＞ 2

bに関する解答群

ア　list[i]　　イ　list[num+i]

ウ　list[num1+i]　　エ　list[num2+i]

cに関する解答群

ア　(i ＜ num1) and (j ＜ num2)　　イ　(i ＜ num1) or (j ＜ num2)

ウ　(j ＜ num1) and (i ＜ num2)　　エ　(j ＜ num1) or (i ＜ num2)

オ　(i+j) ＜ (num1+num2)　　カ　(i+j) ≦ (num1+num2)

キ　(i+j) ＞ (num1+num2)　　ク　(i+j) ≧ (num1+num2)

設問2　最初に与えられた配列listのデータが次の場合，プログラムSortの α における配列listの内容の移り変わりとして正しい答えを，解答群の中から選べ。

配列listのデータ：　3, 8, 2, 7, 5, 1

なお，解答群の“→”は，内容が左から右へ移り変わっていくことを示している。

解答群

ア　2 → 3 → 2,3 → 2,3,8 → 1 → 5 → 1,5 → 1,5,7 → 1,2,3,5,7,8

イ　3 → 8 → 3,8 → 2,3,8 → 7 → 5 → 7,5 → 1,5,7 → 1,2,3,5,7,8

ウ　2,8 → 2,3,8 → 1,5 → 1,5,7 → 1,2,3,5,7,8

エ　3,8 → 2,3,8 → 7,5 → 1,5,7 → 1,2,3,5,7,8

オ　2,3,8 → 1,5,7 → 1,2,3,5,7,8

カ　3,8,2 → 7,5,1 → 1,2,3,5,7,8

設問3　副プログラムMergeの β 部分と同じ結果を得る処理として正しい答えを，解答群の中から選べ。

解答群

ア

```
i < num1
  ・list[i] ← slist1[i]
  ・i ← i + 1

j < num2
  ・list[j] ← slist2[j]
  ・j ← j + 1
```

イ

```
i < num1
  ・list[i+num1] ← slist1[i]
  ・i ← i + 1

j < num2
  ・list[j+num2] ← slist2[j]
  ・j ← j + 1
```

ウ

```
i < num1
  ・list[j+num1] ← slist1[i]
  ・i ← i + 1

j < num2
  ・list[i+num2] ← slist2[j]
  ・j ← j + 1
```

エ

```
i < num1
  ・list[i+num2] ← slist1[i]
  ・i ← i + 1

j < num2
  ・list[j+num1] ← slist2[j]
  ・j ← j + 1
```

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問以上選択した場合には，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〕

関数 printout_text は，ファイルから英文テキストを読み込んで，単語が行末の改行で切れないように整形して出力するプログラムである。

(1) 関数 printout_text の引数は，次のとおりである。ここで，ファイル名に誤りはないものとする。

filename	英文テキストが格納されているファイル名

(2) 英文テキストに含まれる文字は，次のものである。

① 英字　A ～ Z，a ～ z

② 数字　0 ～ 9

③ 記号　! " # % & ' ( ) * + , - . / : ; < = > ? [ ] ^ _ { | } ~

④ 空白文字

⑤ 改行文字

(3) 出力する1行の最大文字数（改行文字を含まない）は80文字である。

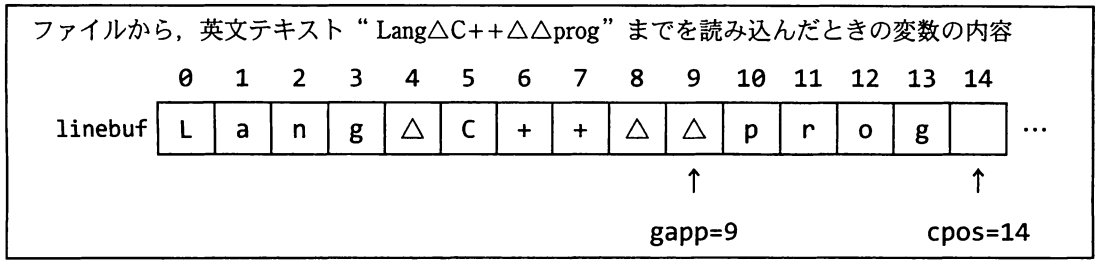
(4) 単語は空白文字及び改行文字を含まない文字列であり，1文字以上の連続する空白文字又は改行文字で区切られている。

(5) 空白文字及び改行文字はそのまま出力する。

(6) 単語の途中で1行の最大文字数を超える場合は，その単語が次の行の先頭になるように出力する。

(7) 単語の長さは，1行の最大文字数を超えない。

(8) プログラム中で使われる変数 cpos は出力用行バッファ linebuf に次の文字を格納する位置を，変数 gapp は linebuf に最後に格納した空白文字の位置をもつ。変数 cpos，gapp の例を，図1に示す。



注 “△”は空白文字を表す。

図1 変数cpos，gappの例

〔プログラム〕

(行番号)

```
 1  #include <stdio.h>

 2  #define COLUMNS 80  /* 1行の最大文字数 */

 3  void printout_text(char *);

 4  void printout_text(char *filename) {
 5      FILE *fp;
 6      char linebuf[COLUMNS + 1];
 7      int  ch, ch1, cpos = 0, gapp = -1, i;

 8      fp = fopen(filename, "r");
 9      while ((ch = fgetc(fp)) != EOF) {
10          switch (ch) {
11            case '\n':
12              linebuf[cpos] = '\0';
13              printf("%s\n", linebuf);
14              cpos = 0;
15              gapp = -1;
16              break;
17            case ' ':
18              if ( [ a ] ) {
19                 linebuf[cpos] = '\0';
20                  printf("%s\n", linebuf);
21                  cpos = 0;
22              }
23              [ b ];
24              [ c ] = ch;
25              break;
```

```
26              default:
27                if ( [ a ] ) {
28                    i = gapp + 1;
29                    ch1 = linebuf[i];
30                    linebuf[i] = '\0';
31                    printf("%s\n", linebuf);
32                    cpos = 0;
33                    if (i < COLUMNS) {
34                        linebuf[cpos++] = ch1;
35                        [ d ];
36                        while (i < COLUMNS) {
37                            linebuf[cpos++] = linebuf[i++];
38                        }
39                    }
40                    gapp = -1;
41                }
42                [ c ] = ch;
43            } /* switch 文の終わり */
44        }
45        if (cpos > 0) {
46            linebuf[cpos] = '\0';
47            printf("%s", linebuf);
48        }
49        fclose(fp);
50    }
```

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　cpos < COLUMNS　　イ　cpos <= COLUMNS

ウ　cpos > COLUMNS　　エ　cpos >= COLUMNS

オ　gapp < COLUMNS　　カ　gapp <= COLUMNS

キ　gapp > COLUMNS　　ク　gapp >= COLUMNS

bに関する解答群

ア　gapp = -1　　イ　gapp = cpos

ウ　gapp = COLUMNS　　エ　gapp++

オ　gapp--

cに関する解答群

ア linebuf[cpos]	イ linebuf[cpos++]
ウ linebuf[gapp]	エ linebuf[gapp++]
オ linebuf[++cpos]	カ linebuf[++gapp]

dに関する解答群

ア i = -1	イ i = cpos	ウ i = COLUMNS
エ i++	オ 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。

入力した英文テキスト（以下，入力英文テキストという）中に複数回出現する単語をキーワードという。最初の出現場所でキーワードとする単語と参照番号を定義し，それ以降の出現場所では参照番号で参照するように入力英文テキストの様式を変更した。入力英文テキストを整形して出力するときには，参照番号で示されているキーワードは，本来の文字列に戻して出力するようにプログラムを変更する。ここで，プログラム中の［ a ］～［ d ］には，正しい答えが入っているものとする。

(1) 入力英文テキスト中でのキーワードの表現様式は図2のとおりである。

なお，キーワードの長さは20文字以下であり，キーワードの個数は50以下である。また，入力英文テキストに含まれる文字として文字“\”を追加する。追加された文字“\”は，キーワードの定義及び参照以外に使われることはない。

最初の出現場所でのキーワードと参照番号の定義様式：

\D 参照番号\単語\

　　参照番号　　キーワードの参照番号（0～49）

　　単語　　キーワードとする単語

キーワードの参照様式：

\R 参照番号\

　　参照番号　　キーワードの参照番号（0～49）

図2　キーワードの定義様式と参照様式

(2) 入力英文テキストを，次の規則で，出力する英文テキスト（以下，出力英文テキストという）に変換しながら，キーワードを含め単語が行末の改行で切れないように整形して出力する。

① キーワードと参照番号を定義している部分については，定義しているキーワードだけを出力英文テキストに残す。変換例を図3に示す。

入力英文テキスト	Lang△\D2\C++\△△prog
出力英文テキスト	Lang△C++△△prog

注 “△”は空白文字を表す。

図3 キーワードと参照番号を定義している部分の変換例

② キーワードを参照している部分については，参照番号に対応するキーワードで置き換える。変換例を図4に示す。

入力英文テキスト	Develop△△\R2\△△compiler
出力英文テキスト	Develop△△C++△△compiler

注 “△”は空白文字を表す。

図4 キーワードを参照している部分の変換例

③ その他の部分は，〔プログラムの説明〕の(1)～(8)に従う。

(3) キーワード処理に対応するために，プログラムを表のとおりに変更する。

表 プログラムの変更内容

処置	プログラムの変更内容
行番号1と2の間に追加	`#include <string.h>`
行番号2と3の間に追加	`#define WORDL 20  /* キーワードの最大文字数 */` `#define REFMX 50  /* キーワードの最大個数 */`
行番号7と8の間に追加	`char reftbl[REFMX][WORDL + 1];` `int  ch2, j;`
行番号25と26の間に追加	〔プログラムの追加部分〕

〔プログラムの追加部分〕で使用している関数 strlen の仕様は次のとおりである。

```
unsigned int strlen(const char *s)
```

機能：文字列 s の長さを求める。

返却値：終端を示すナル文字に先行する文字の個数。

〔プログラムの追加部分〕

```
case '\\':
    ch1 = fgetc(fp);
    i = 0;
    while ((ch2 = fgetc(fp)) != '\\') {
        i = [    e    ];
    }
    if ([    f    ]) {
        j = 0;
        while ((reftbl[i][j++] = fgetc(fp)) != '\\');
        reftbl[i][j - 1] = '\0';
    }
    j = strlen(reftbl[i]);
    if ([    g    ]) {
        linebuf[cpos] = '\0';
        printf("%s\n", linebuf);
        cpos = 0;
        gapp = -1;
    }
    strcpy(&linebuf[cpos], reftbl[i]);
    cpos += j;
    break;
```

eに関する解答群

ア ch2	イ ch2 - '0'
ウ i + ch2	エ i + ch2 - '0'
オ i * 10 + ch2	カ i * 10 + ch2 - '0'

fに関する解答群

ア ch1 == 'D'	イ ch1 == 'R'	ウ ch1 == '\\'
エ i == 0	オ i != 0	

gに関する解答群

ア　cpos >= COLUMNS

イ　cpos + j >= COLUMNS

ウ　cpos + j - 1 >= COLUMNS

エ　gapp >= COLUMNS

オ　gapp + j - 1 >= COLUMNS

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１～３に答えよ。

〔プログラム１の説明〕

G社では，セキュリティ対策のため，入退室時のIDカード操作によって資格保有者以外の立入りを禁止し，かつ，入室者の入退室日時を記録しておくセキュリティルームを設置した。プログラム１は，セキュリティルームを管理するシステムのサブプログラムであり，処理コードと従業員番号を受け取って，結果コードを返す。

(1) パラメタの形式は，図１のとおりである。

処理コード １けた	従業員番号 ８けた	結果コード １けた

図1 パラメタの形式

① 処理コードには，入室時には１が，退室時には２が設定される。

② 結果コードには，入室時に入室を許可する場合は０を，許可しない場合は，事象に対応するコードを設定する。退室時には０を設定する。

(2) 従業員ファイルは，従業員番号をキーとする，図２に示すレコード様式の索引ファイルである。

従業員番号 ８けた	職級 １けた	資格 １けた

図2 従業員ファイルのレコード様式

① 従業員番号には，従業員に対して一意に割り振られた８けたの数字が設定される。ただし，従業員番号０の従業員はいない。

② 職級には，当該従業員の職級が設定される。一般社員，係長，課長，部長に対応する値として，それぞれ１，２，３，４が設定される。

③ 一般社員及び係長の資格には，入室許可申請によって入室を許可された場合には１が，許可されていない場合には０が設定される。課長及び部長の資格に格納される値は不定である。

(3) セキュリティルームに入室できるのは，課長，部長，入室を許可された一般社員及び入室を許可された係長である。

(4) セキュリティルーム内に同時に在室できる人員は99名までとする。

(5) 入室者の入退室日時を記録する入退室ログファイルは，図3に示すレコード様式の順ファイルである。

従業員番号 8けた	入退室コード 1けた	入退室日時 14けた

図3 入退室ログファイルのレコード様式

① 入退室コードには，入室記録の場合は1が，退室記録の場合は2が設定される。

② 入退室日時には，西暦の日付と24時間表記の時刻を格納する。この情報は，組込み関数 CURRENT-DATE の関数値の先頭14文字を使用する。

③ レコードは，入退室処理時に出力する。したがって，このファイル中の全レコードは，入退室日時の昇順に整列されている。

④ セキュリティルームを管理するシステムは，メンテナンスのため月末の深夜に一時運用を停止して，入退室ログファイルを新しいファイルに切り換える。

なお，メンテナンス開始時はセキュリティルーム内を無人にし，完了するまで入室を禁止する。したがって，同一人物による入室と退室のレコードは必ず対で格納される。

(6) このプログラムは，最初に呼び出されたときを除き，CANCEL文などによって初期状態になることはない。

(7) セキュリティルームには，本人のIDカードを使って1人ずつ入退室する。セキュリティルームの入退室はすべて記録される。

〔プログラム1〕

```
DATA DIVISION.
FILE SECTION.
FD  EMPLOYEE.
01  EMP-REC.
    02  EMP-KEY.
        03  EMP-NO        PIC 9(8).
    02  EMP-CLASS         PIC 9(1).
    02  EMP-QUAL          PIC 9(1).
FD  LOG-FILE.
01  LOG-REC.
    02  LOG-EMP           PIC 9(8).
    02  LOG-CD            PIC 9(1).
    02  LOG-TIME          PIC X(14).
```

```
WORKING-STORAGE SECTION.
01  CNT                 PIC 9(3) VALUE ZERO.
LINKAGE SECTION.
01  PRM.
    02  PRM-CD          PIC 9(1).
        88 CD-ENTER       VALUE 1.
        88 CD-EXIT        VALUE 2.
    02  PRM-EMP         PIC 9(8).
    02  PRM-RTN         PIC 9(1).
PROCEDURE DIVISION  USING PRM.
CTL-PROC SECTION.
    MOVE ZERO TO PRM-RTN.
    EVALUATE TRUE
      WHEN CD-ENTER PERFORM QUAL-CHECK
      WHEN CD-EXIT  PERFORM REC-LOG
                    SUBTRACT 1 FROM CNT
      WHEN OTHER    MOVE 1 TO PRM-RTN
    END-EVALUATE.
    EXIT PROGRAM.
QUAL-CHECK SECTION.
    OPEN INPUT EMPLOYEE.
    MOVE PRM-EMP TO EMP-NO.
    READ EMPLOYEE
      INVALID KEY
        MOVE 2 TO PRM-RTN
      NOT INVALID KEY
        EVALUATE TRUE
          WHEN [          a          ]
            MOVE 3 TO PRM-RTN
          WHEN CNT >= 99
            MOVE 4 TO PRM-RTN
          WHEN OTHER
            PERFORM REC-LOG
            [          b          ]
        END-EVALUATE
    END-READ.
    CLOSE EMPLOYEE.
REC-LOG SECTION.
    OPEN EXTEND LOG-FILE.
    MOVE PRM-EMP TO LOG-EMP.
    MOVE PRM-CD  TO LOG-CD.
    MOVE FUNCTION CURRENT-DATE TO LOG-TIME.
    WRITE LOG-REC.
    CLOSE LOG-FILE.
```

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　EMP-CLASS <= 2

イ　EMP-CLASS <= 2 AND EMP-QUAL = 0

ウ　EMP-CLASS <= 2 OR EMP-QUAL = 0

エ　EMP-QUAL = 0

bに関する解答群

ア　ADD 1 TO CNT　　イ　MOVE ZERO TO CNT

ウ　SET CD-ENTER TO TRUE　　エ　SET CD-EXIT TO TRUE

設問2　パラメタで渡された従業員番号が，従業員ファイル中に存在しなかった場合に設定する結果コードを，解答群の中から選べ。

解答群

ア　1　　イ　2　　ウ　3　　エ　4

設問3　プログラム1によって作成され，保管していた，ある月の入退室ログファイルを読み込み，指定した日時にセキュリティルーム内にいたすべての従業員の従業員番号を表示するプログラム2を作成した。指定した日時より前に入室し，かつ，指定した日時か，それより後に退室したすべての従業員の従業員番号を表示する。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。

〔プログラム2〕

```
DATA  DIVISION.
FILE SECTION.
FD  LOG-FILE.
01  LOG-REC.
    02  LOG-EMP          PIC 9(8).
    02  LOG-CD           PIC 9(1).
        88  CD-ENTER       VALUE 1.
        88  CD-EXIT        VALUE 2.
    02  LOG-TIME         PIC X(14).
```

```
WORKING-STORAGE SECTION.
77  CNT                  PIC 9(3).
77  SRCH-FLAG           PIC X(1).
    88 SRCH-END           VALUE "E".
01  STAY-ROOM.
    02  EMP-NO          PIC 9(8) OCCURS 99.
LINKAGE SECTION.
01  PRM-TIME         PIC X(14).
PROCEDURE DIVISION  USING PRM-TIME.
MAIN-PROC SECTION.
    MOVE SPACE TO SRCH-FLAG.
    PERFORM VARYING CNT FROM 1 BY 1 UNTIL CNT > 99
       MOVE ZERO TO EMP-NO(CNT)
    END-PERFORM.
    OPEN INPUT LOG-FILE.
    READ LOG-FILE AT END SET SRCH-END TO TRUE.
    PERFORM UNTIL SRCH-END
       IF [    c    ] THEN
          PERFORM PROC-DISP
          SET SRCH-END TO TRUE
       ELSE
          EVALUATE TRUE
             WHEN CD-ENTER PERFORM PROC-ENTER
             WHEN CD-EXIT  PERFORM PROC-EXIT
          END-EVALUATE
          READ LOG-FILE AT END SET SRCH-END TO TRUE
       END-IF
    END-PERFORM.
    CLOSE LOG-FILE.
    EXIT PROGRAM.
PROC-ENTER SECTION.
    PERFORM VARYING CNT FROM 1 BY 1 UNTIL [    d    ]
       CONTINUE
    END-PERFORM.
    MOVE LOG-EMP  TO EMP-NO(CNT).
PROC-EXIT SECTION.
    PERFORM VARYING CNT FROM 1 BY 1 UNTIL [    e    ]
       CONTINUE
    END-PERFORM.
    MOVE ZERO TO EMP-NO(CNT).
PROC-DISP SECTION.
    PERFORM VARYING CNT FROM 1 BY 1 UNTIL [    f    ]
       IF EMP-NO(CNT) NOT = ZERO THEN
          DISPLAY EMP-NO(CNT)
       END-IF
    END-PERFORM.
```

c に関する解答群

ア　LOG-TIME < PRM-TIME

イ　LOG-TIME <= PRM-TIME

ウ　LOG-TIME = PRM-TIME

エ　LOG-TIME > PRM-TIME

オ　LOG-TIME >= PRM-TIME

d ～ f に関する解答群

ア　CNT < 99

イ　CNT > 99

ウ　EMP-NO(CNT) = ZERO

エ　LOG-EMP = EMP-NO(CNT)

オ　LOG-EMP = ZERO

カ　LOG-EMP NOT = EMP-NO(CNT)

キ　LOG-EMP NOT = ZERO

ク　SRCH-END

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1，2に答えよ。

〔プログラムの説明〕

リバーシゲームで2人のプレーヤの対戦を支援するプログラムである。

リバーシは，8行8列の升からなる盤に，2人のプレーヤが，一方の面が白，他方の面が黒の石を置いていき，盤上の石の個数を競うゲームである。各プレーヤは自分の石の色（白又は黒）をもつ。盤には，初期状態として，図1のとおり中央に白黒2個ずつの石を配置する。升の位置は，列をアルファベット，行を数字で表し，列・行の順に指定する。例えば，図1で最も左上の升はa1，最も右下の升はh8である。

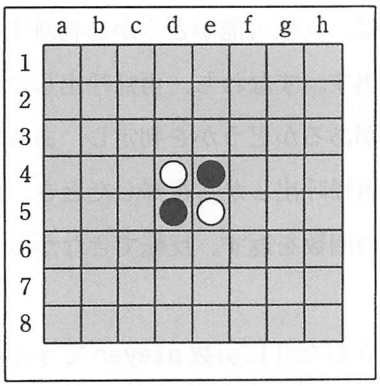


図1　盤の初期状態

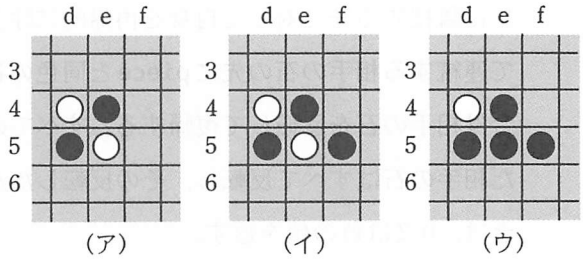


図2　盤に石を置く例

プレーヤは，空いている升に石を置いて，縦，横，斜めのどれかの方向に隣接する相手の石を自分の色の石で挟まなければならない。このとき，挟んだ相手の石をすべて反転させ自分の色の石とする。図2に，黒をもつプレーヤが盤に石を置く例を示す。（ア）は図1の中央部分を抜き出したものである。（イ）は，黒を升f5に置いて，升e5の白を挟んだ状態である。（ウ）は，挟んだ白を反転させて黒にした状態である。相手の石を挟むことができないときは，そのプレーヤは石を置くことができないので，手番をパスする。片方の色の石が盤上からなくなったとき，すべての升が石で埋まったとき，又は空いているどの升にも石が置けなくなったとき，ゲームは終了し，自分の色の石の個数の多い方のプレーヤが勝ちとなる。

列挙Pieceは石の面を表す。フィールドBLACKは黒，WHITEは白を表す。

クラスPlayerは，プレーヤを表す。フィールドpieceは，そのプレーヤの石の色を表し，フィールドcountは，盤上での石の個数を表す。

クラスBoardは盤を表し，対戦を支援する次の(1)～(7)の機能を実現する。

(1) 配列gridは，盤の升を表す。配列要素grid[row][col]において，rowは行，colは列である。例えば，升d2は，grid[1][3]で表される。

(2) 列挙Directionは，配列gridにおいて，縦，横，斜め8方向の隣接する升へのrow及びcolのオフセットを表す定数である。

(3) コンストラクタは，インスタンスを生成し，盤を初期状態にする。

(4) メソッドcanPlaceは，引数playerで与えられたプレーヤが石を置ける升があるかどうかを調べ，あるときはtrue，ないときはfalseを返す。

(5) メソッドreverseは，引数row，colで与えられた升に引数pieceで与えられた色の石を置いた場合，その升から引数dirで与えられた方向に隣接する升の石が反転可能であれば反転する。このメソッドは，反転可能かどうかを判断するために隣接する升に対して自身を再帰的に呼び出す。すなわち，再帰呼出しによって連続する相手の石の先にpieceと同色の石があるかどうかを判定し，あった場合は相手の石を反転して復帰する。すべての再帰呼出しから復帰したとき，挟んだ相手の石はすべて反転し，その反転した石の個数を返す。反転できなかったときは，0又は負の値を返す。

(6) メソッドplaceは，引数positionで指定された升に引数playerで与えられたプレーヤの石が置けるかどうかを判定する。石が置ける場合は，升に石を置き，挟んだ相手の石を反転し，反転した石の個数を返す。この処理は，メソッドreverseを呼び出して行う。指定された升に既に石がある場合，又は相手の石を1個も反転できない場合は，0又は負の値を返し，指定された升に石は置かない。

(7) メソッドdisplayは，盤の状態を表示する。

クラスReversiはゲームを実行する。変数playerは，石を置く番のプレーヤである。変数opponentは，playerの対戦相手である。黒をもつプレーヤが常に先手である。playerが石を置いた後に，それぞれのプレーヤの色の石の個数を更新し，ゲームの終了を判断する。ゲームが終了した場合は，盤の最後の状態を表示してプログラムを終了する。終了しなかったときは，playerとopponentが交代し，ゲームを続行する。メソッドpromptは，引数playerの石の色を表示し，石を置く升を“列行”の形式で入力するように促し，入力された升の位置を文字列で返す。

〔プログラム1〕

```
enum Piece { BLACK, WHITE }
```

〔プログラム2〕

```
class Player {
   final Piece piece;
   int count = 2;
   Player(Piece piece) { this.piece = piece; }
   public String toString() { return piece + ": " + count; }
}
```

〔プログラム3〕

```
class Board {
   private final Piece[][] grid = new Piece[8][8];

   private static enum Direction {
      N(-1, 0), NE(-1, 1), E(0, 1), SE(1, 1),
      S(    a    ), SW(1, -1), W(0, -1), NW(-1, -1);
      final int drow, dcol;
      private Direction(int drow, int dcol) {
         this.drow = drow; this.dcol = dcol;
      }
   }

   Board() {
      grid[3][3] = Piece.WHITE; grid[3][4] = Piece.BLACK;
      grid[4][3] = Piece.BLACK; grid[4][4] = Piece.WHITE;
   }

   boolean canPlace(Player player) { /* 実装は省略 */ }

   private int reverse(Piece piece, int row, int col,
                       Direction dir) {
      int nrow = row + dir.drow;
      int ncol = col + dir.dcol;
      Piece next = null;
      if ((nrow >= 0 && nrow < grid.length) &&
          (ncol >= 0 && ncol < grid[nrow].length)) {
         next = grid[nrow][ncol];
      }
      if (next == null)
         return -1;
      if (next == piece)
         return 0;
      int count = reverse(piece, nrow, ncol, dir);
      if (count >= 0) {
         b     = piece;
         count++;
```

```
        }
        return count;
    }

    int place(Player player, String position) {
        int count = 0;
        try {
            int col = position.charAt(0) - 'a';
            int row = position.charAt(1) - '1';
            if (grid[row][col] != null)
                return -1;
            for (Direction dir : Direction.values()) {
                int n = reverse(player.piece, row, col, dir);
                if (n > 0)
                    count [   c   ] n;
            }
            if (count > 0)
                grid[row][col] = player.piece;
        } catch (IndexOutOfBoundsException e) { }
        return count;
    }

    void display() { /* 実装は省略 */ }
}
```

〔プログラム4〕

```
public class Reversi {
    public static void main(String[] args) {
        Board board = new Board();
        Player player   = new Player(Piece.BLACK),
               opponent = new Player(Piece.WHITE);
        while (true) {
            if (board.canPlace(player)) {
                board.display();
                int n;
                while ((n = board.place(player, prompt(player))) <= 0)
                    System.out.println("wrong position");
                player.count += [   d   ];
                opponent.count -= n;
                System.out.println(player + ", " + opponent);
                if (player.count + opponent.count == 8 * 8
                    || opponent.count == 0) {
                    board.display();
                    break;
                }
            } else if (!board.canPlace(opponent)) {
                board.display();
                break;
            }
            Player p = [   e   ];
```

```
            opponent = player;
            player = p;
        }
    }

    private static String prompt(Player player) { /* 実装は省略 */ }
}
```

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　-1, 0　　イ　-1, 1　　ウ　0, 0

エ　0, 1　　オ　1, 0

bに関する解答群

ア　grid[count][0]　　イ　grid[nrow][col]

ウ　grid[nrow][ncol]　　エ　grid[row][col]

オ　grid[row][ncol]

cに関する解答群

ア　*=　　イ　+=　　ウ　-=

エ　/=　　オ　=

dに関する解答群

ア　++n　　イ　n　　ウ　n + 1

エ　n - 1　　オ　n++

eに関する解答群

ア　board　　イ　opponent　　ウ　Piece.BLACK

エ　Piece.WHITE　　オ　player

設問2　次の表は，プログラム3のメソッド reverse の仕様をまとめたものである。表中の［　　］に入れる正しい答えを，解答群の中から選べ。ここで，b には正しい答えが入っているものとする。

戻り値	意味
正の値	引数 row，col で与えられた升から引数 dir で与えられた方向で反転した石の個数
0	引数 row，col で与えられた升から引数 dir で与えられた方向に［ f ］
-1	引数 row，col で与えられた升から引数 dir で与えられた方向に［ g ］

解答群

ア　隣接する升が存在しない，又は隣接する升に石がない場合

イ　隣接する升にPiece以外のクラスのインスタンスが存在する場合

ウ　隣接する升の石が引数pieceで与えられた石と同色の場合

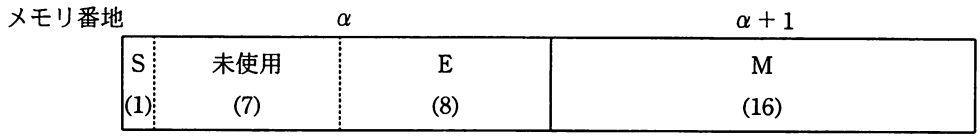
エ　隣接する升の石が引数pieceで与えられた石と異なる色の場合

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1，2に答えよ。

〔プログラムの説明〕

浮動小数点数の加算を行う副プログラムFADDである。

(1) 浮動小数点数は，メモリ中の連続する2語に次の形式で格納する。



S：符号部　（正は0，負は1）　　（ ）内はビット数
未使用　　（ビット0を詰める）
E：指数部　（0～255の符号なし2進整数，実際の指数に127を加算した値）
M：仮数部　（2進固定小数点数，最上位ビットの左に小数点があると仮定する）

この形式で表される浮動小数点数は$(-1)^{S}\times 0.M\times 2^{E-127}$である。

S＝E＝M＝0でゼロを表現する（M＝0のときは，必ずS＝E＝0であること）。

(2) FADDは，被加数と加数について，いずれか指数の大きい方にもう一方の指数をそろえてから仮数を加算し，結果がゼロの場合を除き正規化（仮数部の最上位ビットが1となるように指数部を調整）する。指数部調整の過程で，アンダフロー（指数部が0未満になるので正規化が不可能），オーバフロー（指数部が255を超えるので正規化が不可能）は発生しないものとする。

正規化の例

〔E＝10000100　　M＝0010000000000000〕
↓正規化（Mを2ビット左に論理シフトし，Eから2を減じる）
〔E＝10000010　　M＝1000000000000000〕

(3) FADDは，被加数X，加数Y及び結果Zの格納領域の先頭番地を，それぞれGR1，GR2，GR3に設定して呼び出される。

(4) FADD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〕

（行番号）

```
 1  FADD    START               ; Z ← X + Y
 2          RPUSH
 3          PUSH  0,GR3         ; 結果 Z の格納領域の先頭番地を退避
 4          LD    GR4,0,GR1
 5          AND   GR4,=#00FF    ; Ex: X の指数
 6          LD    GR5,0,GR2
 7          AND   GR5,=#00FF    ; Ey: Y の指数
 8          LD    GR6,1,GR1     ; Mx: X の仮数
 9          LD    GR7,1,GR2     ; My: Y の仮数
10  ; 加算前の準備（指数をそろえる）
11  ; GR4 ← max(Ex,Ey) , GR6 ← 調整済 Mx , GR7 ← 調整済 My
12          LD    GR3,GR4
13          CPL   GR4,GR5
14          JZE   MADD          ; Ex = Ey の場合
15          JMI   BIGEY         ; Ex < Ey の場合
16          SUBL  GR3,GR5
17          SRL   GR7,0,GR3     ; My を調整
18          JUMP  MADD
19  BIGEY   [        a        ]
20          SUBL  GR5,GR3
21          SRL   GR6,0,GR5     ; Mx を調整
22  ; 符号を考慮した仮数の加算
23  ; Sz: Z の符号 , Ez: Z の指数 , Mz: Z の仮数
24  ; GR4 ← (Sz,Ez) , GR5 ← Mz
25  MADD    LD    GR1,0,GR1     ; X の符号の検査
26          JMI   XMINUS        ;   負の場合
27          LD    GR2,0,GR2     ; Y の符号の検査
28          [        b        ]
29          LD    GR5,GR6       ; X ≧ 0, Y ≦ 0 の場合
30          SUBL  GR5,GR7       ; Mz ← 調整済 Mx − 調整済 My
31          JUMP  SCHECK
32  XMINUS  LD    GR2,0,GR2
33          JMI   YMINUS
34          LD    GR5,GR7
35          SUBL  GR5,GR6
36          [        c        ]
37  YMINUS  OR    GR4,=#8000    ; Sz に負符号を設定
38  ADDMXY  LD    GR5,GR6
39          ADDL  GR5,GR7       ; Mz ← 調整済 Mx + 調整済 My
40          JOV   ADJST         ; けた上がりがある場合の正規化
41          JUMP  NORM
```

アセンブラ

```
42  SCHECK  JOV   NEGMZ        ; Mz の符号を検査
43          JUMP  NORM
44  NEGMZ   OR    GR4,=#8000   ; Sz に負符号を設定
45          XOR   GR5,=#FFFF   ; Mz ← −Mz
46          ADDL  GR5,=1
47  ; 加算結果の正規化
48  NORM    LD    GR5,GR5      ; ゼロチェック
49          JNZ   LOOP
50          LD    GR4,=0
51          JUMP  FIN
52  LOOP    LD    GR5,GR5      ; 正規化完了？
53          JMI   FIN
54          [     d      ]
55          SUBL  GR4,=1
56          JUMP  LOOP
57  ADJST   SRL   GR5,1
58          OR    GR5,=#8000   ; Mz の最上位ビットを 1 に設定
59          [     e      ]
60  FIN     POP   GR3
61          ST    GR4,0,GR3    ; 結果 Z の格納
62          ST    GR5,1,GR3
63          RPOP
64          RET
65          END
```

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	ADDL GR4,GR5	イ	ADDL GR5,GR4
ウ	LD GR4,GR5	エ	LD GR5,GR4
オ	SRL GR4,0,GR5	カ	SRL GR5,0,GR4

bに関する解答群

ア	JMI ADDMXY	イ	JMI XMINUS
ウ	JMI YMINUS	エ	JPL ADDMXY
オ	JPL XMINUS	カ	JPL YMINUS

cに関する解答群

ア	JOV	ADDMXY	イ	JOV	SCHECK
ウ	JPL	ADDMXY	エ	JPL	SCHECK
オ	JUMP	ADDMXY	カ	JUMP	SCHECK

dに関する解答群

ア	SLL	GR5,0,GR4	イ	SLL	GR5,1
ウ	SRA	GR5,0,GR4	エ	SRA	GR5,1
オ	SRL	GR5,0,GR4	カ	SRL	GR5,1

eに関する解答群

ア	ADDL	GR4,=1	イ	ADDL	GR5,=1
ウ	SLL	GR4,1	エ	SLL	GR5,1
オ	SUBL	GR4,=1	カ	SUBL	GR5,=1

設問2　プログラムを減算用に変更する場合，行番号27及び32の直後に追加する命令として正しい答えを，解答群の中から選べ。

なお，プログラム中のコメントは適宜読み替えるものとする。

解答群

ア	AND	GR2,=#7FFF	イ	AND	GR2,=#8000
ウ	OR	GR2,=#7FFF	エ	OR	GR2,=#8000
オ	XOR	GR2,=#7FFF	カ	XOR	GR2,=#8000

問13 次の表計算及びワークシートの説明を読んで，設問1，2に答えよ。

〔表計算の説明〕

ある喫茶店では，これまで手書き伝票と電卓を使って料金計算を行ってきたが，飲料の品目サイズ別メニュー化及び料理オプションの追加に伴い，表計算ソフトを利用することにした。

〔ワークシート：メニュー〕

(1) この喫茶店の飲料，料理を記載した表計算ソフトのワークシート“メニュー”を作成した。そのワークシートを図1に示す。

	A	B	C	D	E	F	G	H	I	J
1	飲料メニュー						料理メニュー			
2	飲料品目	サイズ					料理品目	オプション		
3		S	M	L	セット割引			単品	ランチ	大盛
4	ブレンド	300	350	400	−150		サンドイッチ	550	650	—
5	カプチーノ	350	400	450	−200		ハンバーガー	400	550	—
6	カフェモカ	400	450	500	−250		カレー	650	850	950
7	抹茶ラテ	450	500	550	−300		ナポリタン	700	900	1,050

図1 ワークシート“メニュー”

(2) ワークシート“メニュー”中の各項目の説明は，次のとおりである。

飲料品目 ：“ブレンド”などの4品目である。

サイズ ：“S”，“M”，“L”の3種類で，品目ごとサイズごとの価格である。

セット割引：料理と飲料を同時に注文した場合に適用される飲料の割引額である。ワークシートには，負の値を設定する。

料理品目 ：“サンドイッチ”などの4品目である。

オプション：“単品”，“ランチ”，“大盛”の3種類である。“単品”は，料理だけを注文した場合の価格である。“ランチ”は，単品にスープ又はサラダが付く注文をした場合の価格である。“大盛”は，大盛を注文した場合の価格であり，“カレー”と“ナポリタン”のランチに限る。

〔ワークシート：伝票〕

(1) 注文された飲料及び料理の数量の入力によって，料金計算を行うためのワークシート“伝票”を作成した。そのワークシートの例を，図2に示す。

	A	B	C	D	E	F
1	種類	品目	タイプ	単価	数量	料金
2	飲料	ブレンド	S	300	4	1,200
3	飲料	ブレンド	M	350	0	0
4	飲料	ブレンド	L	400	0	0
5	飲料	ブレンド	セット割引	−150	4	−600
6	飲料	カプチーノ	S	350	0	0
7	飲料	カプチーノ	M	400	5	2,000
8	飲料	カプチーノ	L	450	0	0
9	飲料	カプチーノ	セット割引	−200	4	−800
10	飲料	カフェモカ	S	400	0	0
11	飲料	カフェモカ	M	450	3	1,350
12	飲料	カフェモカ	L	500	0	0
13	飲料	カフェモカ	セット割引	−250	1	−250
14	飲料	抹茶ラテ	S	450	0	0
15	飲料	抹茶ラテ	M	500	0	0
16	飲料	抹茶ラテ	L	550	3	1,650
17	飲料	抹茶ラテ	セット割引	−300	1	−300
18	料理	サンドイッチ	単品	550	0	0
19	料理	サンドイッチ	ランチ	650	3	1,950
20	料理	ハンバーガー	単品	400	0	0
21	料理	ハンバーガー	ランチ	550	3	1,650
22	料理	カレー	単品	650	0	0
23	料理	カレー	ランチ	850	2	1,700
24	料理	カレー	大盛	950	1	950
25	料理	ナポリタン	単品	700	1	700
26	料理	ナポリタン	ランチ	900	2	1,800
27	料理	ナポリタン	大盛	1,050	1	1,050
28	合計料金					14,050

図2　ワークシート“伝票”の例

(2) ワークシート“伝票”には次の文字列及び計算式を入力する。

① 行2～27の種類，品目，タイプには，あらかじめ，次のすべての取り得る組合せを入力する。

種類　：“飲料”又は“料理”を入力する。

品目　：飲料又は料理の品目を入力する。

タイプ：飲料の場合は，サイズの種類又は“セット割引”を入力する。

料理の場合は，オプションの種類を入力する。

② 行2～27の単価には，ワークシート“メニュー”を参照して，該当する価格を求める計算式を入力する。

③ 行2～27の料金には，品目，タイプごとの料金を求める計算式を，行28には，セット割引を加味した合計料金を求める計算式を入力する。

(3) ワークシート“伝票”を作成する。

① ワークシート“伝票”は，顧客のグループ単位に注文をまとめて作成する。

② グループ内の個々の顧客の注文に基づき，品目，タイプごとの注文数量を数量に入力する。

③ 品目，タイプごとの料金が計算され，更に合計料金が計算される。

表1　ワークシート“伝票”で用いる関数

書式	説明
配列(範囲,行位置,列位置)	2次元の範囲の中から，行位置と列位置を指定して値を取り出す。行位置は，範囲の中の上端から数えた行数で指定する。列位置は，範囲の中の左端から数えた列数で指定する。
照合一致(検索値,範囲)	1次元の範囲の中を左端又は上端から検索し，検索値と初めて一致した値が何番目に位置するかを数値で返す。

設問1　ワークシート“伝票”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。

セルD2に単価を求めるための次の計算式を入力して，セルD3～D27に複写する。ここで，複数のワークシート間でデータを参照するには，

“ワークシート名！セル”又は“ワークシート名！セルの範囲”という形式で指定する。

IF(A2＝'飲料',［　a　］,［　b　］)

解答群

ア　照合一致(配列(メニュー!$B$4～$E$7,B2,C2),メニュー!$A$4～$A$7)

イ　照合一致(配列(メニュー!$B$4～$E$7,B2,C2),メニュー!$B$3～$E$3)

ウ　照合一致(配列(メニュー!$H$4～$J$7,B2,C2),メニュー!$G$4～$G$7)

エ　照合一致(配列(メニュー!$H$4～$J$7,B2,C2),メニュー!$H$3～$J$3)

オ　配列(メニュー!$B$4～$E$7,照合一致(B2,メニュー!$A$4～$A$7),
照合一致(C2,メニュー!$B$3～$E$3))

カ　配列(メニュー!$B$4～$E$7,照合一致(B2,メニュー!$B$3～$E$3),
照合一致(C2,メニュー!$A$4～$A$7))

キ　配列(メニュー!$B$4～$E$7,照合一致(C2,メニュー!$B$3～$E$3),
照合一致(B2,メニュー!$A$4～$A$7))

ク　配列(メニュー!$H$4～$J$7,照合一致(B2,メニュー!$G$4～$G$7),
照合一致(C2,メニュー!$H$3～$J$3))

ケ　配列(メニュー!$H$4～$J$7,照合一致(B2,メニュー!$H$3～$J$3),
照合一致(C2,メニュー!$G$4～$G$7))

コ　配列(メニュー!$H$4～$J$7,照合一致(C2,メニュー!$H$3～$J$3),
照合一致(B2,メニュー!$G$4～$G$7))

〔ワークシート：割引〕

顧客満足度向上のための施策として，次の割引策を適用することにした。

(1) 最適割引

顧客のグループ内で，料理と飲料のセットの組合せを変えることで，注文時のセット割引額よりも割引額が大きくなる場合がある。注文時の伝票の入力内容を調べ，注文された料理又は飲料の注文数量の少ない方をセット数の上限として，割引額の高い飲料からセットに割り当てていくことによって，セット割引額の合計が最大となるように割引を行う。

(2) 数量割引

数量割引は，飲料を15個以上かつ料理を10個以上，又は料理と飲料を合わせて30個以上を注文した場合に適用され，最適割引時の合計料金から5％の割引を行う。

最適割引及び数量割引の計算を行うためのワークシート“割引”を作成した。そのワークシートの例を，図3に示す。

なお，数量割引後の合計料金は，小数点以下を切り捨てて表示する。

	A	B	C	D	E	F	G	H
1	セット割引	10						
2	飲料	15						
3	料理	13						
4	最適セット数	13						
5								
6								
7	品目	セット割引	注文時			最適割引時		
8		単価	注文数	セット数	割引額	残数	セット数	割引額
9	抹茶ラテ	−300	3	1	−300	13	3	−900
10	カフェモカ	−250	3	1	−250	10	3	−750
11	カプチーノ	−200	5	4	−800	7	5	−1,000
12	ブレンド	−150	4	4	−600	2	2	−300
13	合計		15	10	−1,950	0	13	−2,950
14	最適割引時の合計料金	13,050						
15	数量割引後の合計料金	12,397						

図3　ワークシート“割引”の例

表2　ワークシート“割引”で用いる関数

書式	説明
照合合計(照合値 , 照合範囲 , 対応範囲)	照合範囲内のセルにおいて，照合値と等しい値をもつセルをすべて探し出す。そして，照合値と等しい値をもつセルの相対位置と同じ位置にある対応範囲のセルの値（数値）を合計して返す。 照合範囲及び対応範囲は，同じ行数及び同じ列数でなければならない。対応範囲には数値が入っていなければならない。
複数条件照合合計((照合値 1 , 照合条件 1 , 照合範囲 1),(照合値 2 , 照合条件 2 , 照合範囲 2), 対応範囲)	照合範囲 1 のセルにおいて照合値 1 に対して照合条件 1 を満足し，併せて照合範囲 2 内のセルにおいて照合値 2 に対して照合条件 2 を同時に満足するセルをすべて探し出す。双方の照合条件に一致したセルの相対位置と同じ位置にある対応範囲のセルの値（数値）を合計して返す。 照合範囲 1 ，照合範囲 2 及び対応範囲は，同じ行数及び同じ列数でなければならない。 照合条件 1 及び照合条件 2 には，比較演算子を指定する。 対応範囲には数値が入っていなければならない。

設問2　ワークシート“割引”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。

(1)　行1～4では，セット割引，飲料，料理の注文数量から，最適セット数を求める。

①　ワークシート“伝票”におけるセット割引，飲料の注文数の合計を求める次の計算式をセルB1，B2に入力する。

セルB1　［　c　］

セルB2　［　d　］

セルB3に料理の注文数の合計を求める計算式を入力する。

②　行4の最適セット数は，最適割引を行うためのセットの注文数の合計であり，飲料又は料理の注文数の合計値の少ない方になる。最適セット数を求める計算式を，セルB4に入力する。

(2) 行7～13では，“伝票”を参照しながら，注文時の注文数，セット数及び割引額を，セット割引単価の大きい順に並べ替えた品目ごとに集計する。さらに，最適割引を行った場合のセット数の割当数及び割引額の計算を行う。

① セルA9～A12にセット割引単価の大きい順に並べ替えた飲料の品目を入力する。

② セルB9～B12に品目ごとのセット割引額を入力する。

③ セルC9に注文時の品目ごとの注文数を求める次の計算式を入力し，セルC10～C12に複写する。

e

④ セルD9に注文時の品目ごとのセット数を求める計算式を入力し，セルD10～D12に複写する。

⑤ セルE9～E12には，注文時の品目ごとの割引額の計算式を入力する。

⑥ セルF9に最適割引を行うための最適セット数であるセルB4の値を複写する。次に，セット割引をセット割引単価の大きい品目の順に割り当てたセット数を求めるためにセルG9に次の計算式を入力し，セルG10～G12に複写する。

f

⑦ セルF10～F13に，セット数を品目ごとに順次割り当てたときの割当数の残数を求める計算式を入力する。

⑧ セルH9～H12に，最適割引時の品目ごとの割引額の計算式を入力する。

⑨ セルC13，D13，E13に，注文時の注文数，セット数，割引額の合計値を求める計算式を入力する。セルF13には，合計値ではなく，⑦で割当数の残数を求める計算式を入力する。セルG13，H13に，最適割引時のセット数，割引額の合計値を求める計算式を入力する。

(3) セルB14に，最適割引時の合計料金を求める計算式を入力する。

(4) 数量割引後の合計料金を求める次の計算式をセルB15に入力する。

g

c，dに関する解答群

ア　照合合計(A1,伝票!C2～C27,伝票!E2～E27)

イ　照合合計(A2,伝票!A2～A27,伝票!E2～E27)

ウ　照合合計(A2,伝票!A2～A27,伝票!E2～E27)－B1

エ　照合合計(A3,伝票!A2～A27,伝票!E2～E27)

オ　照合合計(A3,伝票!A2～A27,伝票!E2～E27)－B1

カ　複数条件照合合計((A2,'≠',伝票!A2～A27),(A1,'＝',伝票!C2～C27),
伝票!E2～E27)

キ　複数条件照合合計((A3,'＝',伝票!A2～A27),(A1,'＝',伝票!C2～C27),
伝票!E2～E27)

ク　複数条件照合合計((A3,'＝',伝票!A2～A27),(A1,'≠',伝票!C2～C27),
伝票!E2～E27)

eに関する解答群

ア　照合合計(A9,伝票!$B$2～$B$27,伝票!$E$2～$E$27)

イ　照合合計(A9,伝票!$B$2～$B$27,伝票!$E$2～$E$27)
－照合合計(B$7,伝票!$C$2～$C$27,伝票!$E$2～$E$27)

ウ　照合合計(B$7,伝票!$C$2～$C$27,伝票!$E$2～$E$27)

エ　照合合計(B$7,伝票!$C$2～$C$27,伝票!$E$2～$E$27)
－照合合計(A9,伝票!$B$2～$B$27,伝票!$E$2～$E$27)

オ　複数条件照合合計((A9,'＝',伝票!$B$2～$B$27),
(B$7,'＝',伝票!$C$2～$C$27),伝票!$E$2～$E$27)

カ　複数条件照合合計((A9,'＝',伝票!$B$2～$B$27),
(B$7,'≠',伝票!C$2～C$27),伝票!E$2～E$27)

キ　複数条件照合合計((A9,'≠',伝票!B$2～B$27),
(B$7,'＝',伝票!C$2～C$27),伝票!E$2～E$27)

ク　複数条件照合合計((A9,'≠',伝票!B$2～B$27),
(B$7,'≠',伝票!C$2～C$27),伝票!E$2～E$27)

fに関する解答群

ア　IF(F9≧C9,C9,F9)　　イ　IF(F9≧C9,C9,F9＋C9)

ウ　IF(F9≧C9,C9,F9－C9)　　エ　IF(F9≦C9,C9,F9)

オ　IF(F9≦C9,C9,F9＋C9)　　カ　IF(F9≦C9,C9,F9－C9)

gに関する解答群

ア　B14－IF(論理和(論理積(B2≧15,B3≧10),(B2＋B3)≧30),0.05,0)＊H13

イ　B14－IF(論理積(論理和(B2≧15,B3≧10),(B2＋B3)≧30),0.05,0)＊H13

ウ　(1＋IF(論理和(論理積(B2≧15,B3≧10),(B2＋B3)≧30),0.05,0))＊B14

エ　(1＋IF(論理積(論理和(B2≧15,B3≧10),(B2＋B3)≧30),0.05,0))＊B14

オ　(1－IF(論理和(論理積(B2≧15,B3≧10),(B2＋B3)≧30),0.05,1))＊B14

カ　(1－IF(論理積(論理和(B2≧15,B3≧10),(B2＋B3)≧30),0.05,1))＊B14

キ　(1－IF(論理積(論理和(B2≧15,B3≧10),(B2＋B3)≧30),0.05,0))＊B14

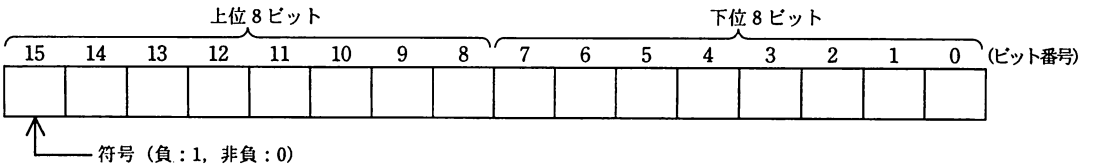
ク　(1－IF(論理和(論理積(B2≧15,B3≧10),(B2＋B3)≧30),0.05,0))＊B14

■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)	0	0
(r) ＞ (実効アドレス)		
(r1) ＝ (r2)	0	1
(r) ＝ (実効アドレス)		
(r1) ＜ (r2)	1	0
(r) ＜ (実効アドレス)		

(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
JOV	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） 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) [ ] [ ] 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。CASL Ⅱ 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

表計算ソフトの機能，用語などは，原則として次による。

1. **ワークシート**

表計算ソフト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（列Aから列Z，列AAから列AZ，さらに列BAから列BZと続き，列IVまで続く），10,000行（行1から行10,000まで）とする。

2. **セル**

(1) ワークシートを縦・横に分割したときの一つのます目をセルという。列A行1のセルはA1と表す。

(2) 長方形の形をしたセルの集まりを範囲として指定することができる。範囲の指定はA1～B3のように表す。

(3) 範囲に名前を付けることができる。範囲名は[ ]を用いて，“セルA1～B3に[金額]と名前を付ける”などと表す。

(4) データが入力されていないセルを，空白セルという。

3. **セルへの入力**

(1) セルに数値，文字列，計算式を入力できる。

(2) セルを保護すると，そのセルへの入力を不可能にすることができる。セルの保護を解除すると，そのセルへの入力が再び可能になる。

(3) セルA1に数値5を入力するときは，“セルA1に5を入力”と表す。

(4) セルB2に，文字列ABCを入力するときは，“セルB2に'ABC'を入力”と表す。

(5) セルC3に，セルA1とセルB2の和を求める計算式を入力するときは，“セルC3に計算式A1＋B2を入力”などと表す。

4. **セルの内容の表示**

(1) セルに数値を入力すると，右詰めで表示される。

(2) セルに文字列を入力すると，左詰めで表示される。

(3) セルに計算式を入力すると，計算結果が数値ならば右詰めで，文字列ならば左詰めで表示される。

(4) セルの内容の表示については，左詰め，中央揃(そろ)え，右詰めに変更できる。

5. **計算式**

(1) 計算式には，数学で用いられる数式が利用できる。

(2) 計算式で使用する算術演算子は，“＋”（加算），“－”（減算），“＊”（乗算），“／”（除算）及び“^”（べき算）とする。

(3) 算術演算子による計算の優先順位は，数学での優先順位と同じである。

6. 再計算

(1) セルに計算式を入力すると，直ちに計算結果を表示する。

(2) セルの数値が変化すると，そのセルを参照しているセルも自動的に再計算される。この再計算はA1，A2，A3，…，B1，B2，B3，…の順に1回だけ行われる。

7. 関数

(1) 計算式には次の表で定義する関数を利用することができる。

関数名と使用例	解　　説
合計(A1～A5)	セルA1からセルA5までの範囲のすべての数値の合計を求める。
平均(B2～F2)	セルB2からセルF2までの範囲のすべての数値の平均を求める。
平方根(I6)	セルI6の値（正の数値でなければならない）の正の平方根を求める。
標準偏差(D5～D19)	セルD5からセルD19までの範囲のすべての数値の標準偏差を求める。
最大(C3～E7)	セルC3からセルE7までの範囲のすべての数値のうちの最大値を求める。
最小([得点])	[得点]と名前を付けた範囲のすべての数値のうちの最小値を求める。
IF(B3＞A4，'北海道'，'九州')	第1引数に指定された論理式が真（成立する）ならば第2引数が，偽（成立しない）ならば第3引数が求める値となる。左の例では，セルB3がA4より大きければ文字列'北海道'が，それ以外の場合には文字列'九州'が求める値となる。論理式中では，比較演算子として，＝，≠，＞，＜，≦，≧を利用することができる。第2引数，第3引数に，更にIF関数を利用して，IF関数を入れ子にすることができる。
個数(G1～G5)	セルG1からG5までの範囲のうち，空白セルでないセルの個数を求める。
条件付個数(H5～H9，'＞25')	第1引数に指定された範囲のうち，第2引数に指定された条件を満たすセルの個数を求める。左の例では，セルH5からH9までの範囲のうち，値として25より大きな数値を格納しているセルの個数を求める。
整数部(A3)	セルA3の値（数値でなければならない）を超えない最大の整数を求める。 例えば， 整数部(3.9)＝3 整数部(－3.9)＝－4 となる。
剰余(C4，D4)	セルC4の値を被除数，D4の値を除数とし，被除数を除数で割ったときの剰余を求める。剰余の値は常に除数と同じ符号をもつ。“剰余”関数と“整数部”関数は，次の関係を満たしている。 剰余(x，y)＝x－y＊整数部(x／y)
論理積(論理式1，論理式2，…)	引数として指定された論理式がすべて真であれば，真を返す。引数のうち一つでも偽のものがあれば，偽を返す。引数として指定できる論理式の数は任意である。
論理和(論理式1，論理式2，…)	引数として指定された論理式がすべて偽であれば，偽を返す。引数のうち一つでも真のものがあれば，真を返す。引数として指定できる論理式の数は任意である。
否定(論理式)	引数として指定された論理式が真であれば偽を，偽であれば真を返す。
注　“合計”，“平均”，“標準偏差”，“最大”，“最小”は，引数で指定された範囲のセルのうち，値として数値以外を格納しているものは無視する。	

(2) 関数の引数には，セルを用いた計算式，範囲，範囲名，論理式を指定することができる。

8. **セルの複写**

(1) セルに入力された数値，文字列，計算式を他のセルに複写することができる。

(2) セルに入力された計算式が他のセルを参照している場合は，複写先のセルでは相対的にセルが自動的に変更される。例えば，セルA6に合計(A1～A5)を入力した場合，セルA6をセルB7に複写すると，セルB7の計算式は合計(B2～B6)となる。

9. **絶対参照**

(1) 計算式を複写しても参照したセルが変わらない参照を絶対参照といい，記号$を用いて$A$1などと表す。例えば，セルB1に計算式$A$1+5を入力した場合，セルB1をセルC4に複写してもセルC4の計算式は$A$1+5のままである。

(2) 絶対参照は行と列の一方だけについても指定可能であり，$A1，A$1などと表す。例えば，セルD2に計算式$C1−3を入力した場合，セルD2をセルE3に複写すると，セルE3の計算式は$C2−3となる。また，セルG3に計算式F$2−3を入力した場合，セルG3をH4に複写すると，セルH4の計算式はG$2−3となる。

10. **マクロ**

(1) ワークシートには幾つかのマクロを保存できる。マクロはマクロP，マクロQなどと表す。

(2) マクロについては“マクロPを実行するとワークシートを保存する。”，“セルA1からセルA10までを昇順に並べ替える手続をマクロQに登録する。”，“マクロR：数値を入力。”，“C列のデータがその数値以下のものを抽出する。”などと記述する。

11. **その他**

ワークシートの“保存”，“読出し”，“印刷”や，罫線機能，グラフ化機能など市販されている多くの表計算ソフトに備わっている機能は使用できるものとする。

〔メモ用紙〕

〔 メ モ 用 紙 〕

〔 メ モ 用 紙 〕

7. 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

8. **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。
9. 問題冊子の余白などは，適宜利用して構いません。
10. 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。
11. 試験時間中，机上に置けるもの及び使用できるものは，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，黒鉛筆又はシャープペンシル，鉛筆削り，消しゴム，定規，時計（アラームなど時計以外の機能は使用不可），ハンカチ，ティッシュ
    これら以外は机上に置けません。使用もできません。
12. 試験終了後，この問題冊子は持ち帰ることができます。
13. 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。
14. 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。