FE

平成26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	ネットワークセキュリティ

●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	JK フリップフロップ
問 3	ソフトウェア	OS におけるプロセスのスケジューリング
問 4	データベース	書籍を管理する関係データベースの設計及び運用
問 5	ソフトウェア設計	共通ライブラリのオブジェクト指向設計
問 6	サービスマネジメント	サービスデスクにおける問合せ対応
問 7	システム戦略	受発注システムの改修

●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	編集距離の算出

●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）	利用者 ID の管理状況の確認
問 10	ソフトウェア開発（COBOL）	売上傾向の分析
問 11	ソフトウェア開発（Java）	可変オブジェクトとその問題点
問 12	ソフトウェア開発（アセンブラ）	バブルソート
問 13	ソフトウェア開発（表計算）	鉄道運賃の計算

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〕

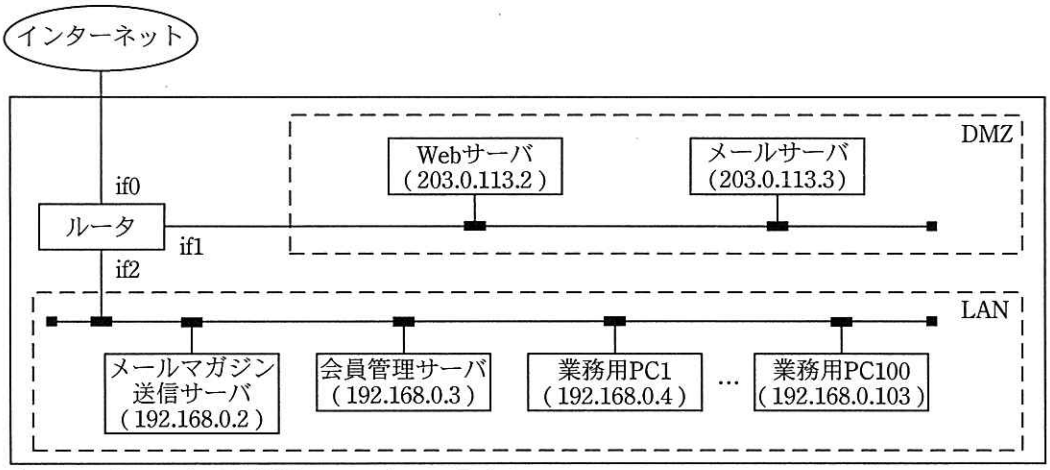
true，false

次の問1は必須問題です。必ず解答してください。

問1　ネットワークセキュリティに関する次の記述を読んで，設問1～4に答えよ。

A社は，社内に設置したWebサーバ上に，自社の製品を紹介するWebサイトを構築し，運営している。A社は，このWebサイトで会員登録を受け付け，登録された会員に対してメールマガジンを発行している。

A社のネットワーク構成を，図1に示す。



注記1　カッコ内は各機器のIPアドレスを表す。
注記2　if0，if1及びif2はルータのネットワークインタフェースを表す。

図1　A社のネットワーク構成

会員登録処理の流れは次のとおりである。

(1)　登録希望者は，インターネットを介し，Webサーバが管理する入会申込用Webページに HTTP over SSL/TLS（以下，HTTPSという）でアクセスし，メールアドレスを入力する。

(2)　Webサーバは，登録希望者ごとに，登録希望者専用の会員情報入力用 Web ページを生成し，その URL を記載した電子メール（以下，メールという）を，入力されたメールアドレス宛てに送信する。

(3)　登録希望者は，(2)で送信されたメールに記載された URL が示す Web ページ

に HTTPS でアクセスして，氏名や職業などの会員情報（メールアドレスは含まない）を入力する。

(4) Web サーバは，(1)のメールアドレスと(3)の会員情報を，会員管理サーバ上で稼働しているデータベース（以下，会員情報 DB という）に登録する。

(5) Web サーバは，会員登録完了を知らせるメールを，(1)のメールアドレス宛てに送信する。

メールマガジン発行の流れは次のとおりである。

(1) メールマガジン担当者は，業務用 PC の Web ブラウザから，メールマガジン送信サーバのメールマガジン入力用 Web ページに HTTP でアクセスし，メールマガジンの本文を入力する。

(2) メールマガジン送信サーバは，会員情報 DB から全ての会員のメールアドレスを取得し，取得したメールアドレス宛てにメールでメールマガジンを送信する。

なお，メールは，メールサーバで稼働しているメール転送サービスを介して送信する。また，本問では，URL やメールアドレスなどの名前解決については，考慮しなくてよいこととする。

設問 1 会員登録の際，登録希望者が最初にアクセスする入会申込用 Web ページでは，登録希望者のメールアドレスだけを入力させ，会員情報の入力は別途行わせる方式を採っている。このように 2 段階の手順を踏む主な目的として適切な答えを，解答群の中から選べ。

解答群

ア 他人のメールアドレスや間違ったメールアドレスが登録されないようにする。

イ 通信を暗号化し，登録希望者の会員情報が第三者に漏れないようにする。

ウ 登録希望者が会員情報 DB にアクセスできないようにする。

エ 間違った会員情報（メールアドレスは含まない）が登録されないようにする。

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。

ルータは，動的なパケットフィルタ型のファイアウォール機能を搭載していて，設定で許可したパケットだけを通過させる。

設定は，送信元，送信先，通信ポートの順に“,”で区切って記述する。これは，送信元から送信先の通信ポート宛てのパケットの通過を許可することを意味する。許可されたパケットに対する応答パケットの通過も許可される。

送信元及び送信先には，IP アドレス又はネットワークインタフェースを指定する。IP アドレスを指定したときに許可の対象となるパケットは，送信元又は送信先の IP アドレスが，指定された IP アドレスであるパケットである。ネットワークインタフェースを指定したときに許可の対象となるパケットは，そのネットワークインタフェースから入ってくるパケット（送信元として指定したとき），又は出ていくパケット（送信先として指定したとき）である。

通信ポートには，各サービスがパケットを待ち受けるポート番号を指定する。A 社の各サーバ上で稼働している各サービスが使用するプロトコルと待受けポート番号を，表1に示す。

表1　サービスごとのプロトコルと待受けポート番号

サービス	プロトコル	待受けポート番号
Web	HTTP	80
	HTTPS	443
会員情報 DB	独自プロトコル	4194
メール取得	POP3	110
メール転送	SMTP	25

現在のルータの設定を，図2に示す。

1 行目の設定で，インターネットから Web サーバに HTTP でアクセスすることを許可し，2 行目の設定で，インターネットから入ってくるメールを，メールサーバに転送することを許可している。

```
if0,203.0.113.2,80
if0,203.0.113.3,25
203.0.113.3,if0,25
if2,203.0.113.2,80
if2,203.0.113.2,443
if2,203.0.113.3,25
if2,203.0.113.3,110
if0,[ a ]
203.0.113.2,[ b ]
```

図2　現在のルータの設定

解答群

ア　192.168.0.2,443　　イ　192.168.0.2,4194

ウ　192.168.0.3,4194　　エ　203.0.113.2,443

オ　203.0.113.2,4194　　カ　if0,443

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。

A社は，Webサーバのメンテナンスを外部委託し，委託先内の特定のPCから，インターネットを介して，Webサーバを操作できるようにした。そのために，Webサーバ上に待受けポート番号22でSSHサービスを稼働させ，ルータの設定に“［　c　］”の行を追加した。

なお，このPCから送信されたパケットがルータに到着したとき，このパケットの送信元IPアドレスは，198.51.100.2となっている。

解答群

ア　198.51.100.2,203.0.113.2,22　　イ　198.51.100.2,if0,22

ウ　203.0.113.2,198.51.100.2,22　　エ　if0,198.51.100.2,22

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。

SSH サービスがクライアントを認証する方式には，パスワード認証方式と公開鍵認証方式がある。A 社は公開鍵認証方式を採用した。

公開鍵認証方式では，秘密鍵で作成した署名が，対応する公開鍵で検証できることを利用して，次のようにクライアントを認証する。

(1)　クライアントは，秘密鍵を使って作成した署名と，その秘密鍵に対応する公開鍵をサーバに送る。

(2)　サーバは，(1)の公開鍵がサーバに登録されていることを確認し，公開鍵で(1)の署名を検証する。

(3)　検証に成功すれば，クライアントがサーバに登録されている公開鍵に対応する秘密鍵をもっていることが証明されるので，サーバは，クライアントを認証する。

このように，公開鍵認証方式では，クライアントがサーバの SSH サービスを利用する際に，パスワードや［　d　］をネットワーク上に流す必要がない。

解答群

ア　公開鍵

イ　秘密鍵

ウ　秘密鍵及び公開鍵

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　JKフリップフロップに関する次の記述を読んで，設問1～3に答えよ。

JK フリップフロップは，二つの信号入力端子 J と K，一つのクロック信号入力端子 CLK，及び二つの信号出力端子 Q と $\overline{Q}$ をもつ回路である。図1に JK フリップフロップの記号を示す。

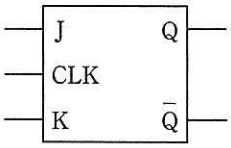


図1　JKフリップフロップの記号

入出力される信号の値は高，低の二つの電圧レベルのいずれかである。クロック信号の値は，周期的に高と低を繰り返す。Q の値は，$\overline{Q}$ の値が高であれば低，低であれば高となる。

各入出力端子の信号の値を当該端子記号で表し，信号の値が高の場合を論理値の 1，低の場合を論理値の 0 として表記する。また，信号の値が低から高に変化することを 0→1，高から低に変化することを 1→0 と表記する。

CLK の立ち下がり（1→0）時に，その時点での J，K，Q の値に基づき，その後の Q の値が決定される。この様子を図2に示す。CLK の立ち下がり時刻を t_1，その後の Q の値が決定した時刻を t_2 として，時刻 t_1 での J，K，Q の値（J_1，K_1，Q_1 と表記）と時刻 t_2 の Q の値（Q_2 と表記）の関係を表1の真理値表に示す。ここで，時刻 t_1 と t_2 の時間間隔は極めて短く，CLK の1周期に比べても十分に短いものとする。

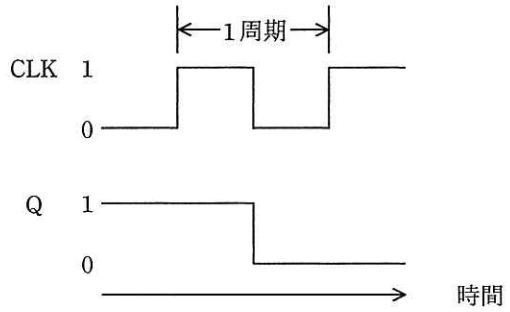


図２　CLKの立ち下がりとQの値の変化例

表１　真理値表

J_1	K_1	Q_1	Q_2
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

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。

図３に示すとおり，JとQ，Kと$\overline{Q}$をそれぞれ同一の値の信号とする回路（端子間を結線する）にクロック信号（CLK）を入力したとき，CLK の立ち下がりでQの値は［ a ］。ここで，Qの初期値は０とする。

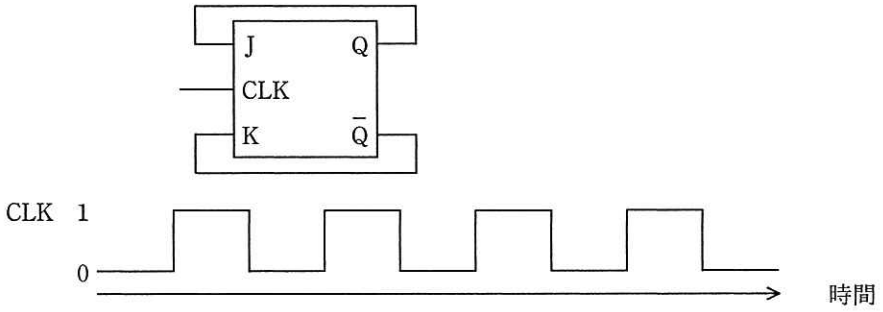


図３　JKフリップフロップの応用回路例

解答群

ア　0のままである　　イ　0→1と変化する

ウ　0→1，1→0と変化する　　エ　0→1，1→0の変化を繰り返す

設問2　表1の真理値表を基に，Q_1からQ_2への変化に着目し，そのときのJ_1，K_1との関係を表2にまとめ直した。表2中の［　　］に入れる正しい答えを，解答群の中から選べ。

表2　Qの値の変化とJ_1，K_1の値の関係

J_1	K_1	$Q_1 \rightarrow Q_2$
b		0→0
任意	1	1→0
1	任意	0→1
c		1→1

注記　任意：0又は1のいずれの値もあり得る。

b，cに関する解答群

ア	0	任意	イ	1	1	ウ	1	任意
エ	任意	0	オ	任意	1			

設問3　JKフリップフロップ1個を使って，図4のように動作する2進カウンタを構成する。ここで，2進カウンタとは，CLKの1周期ごとにQの値が変化するものである。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。

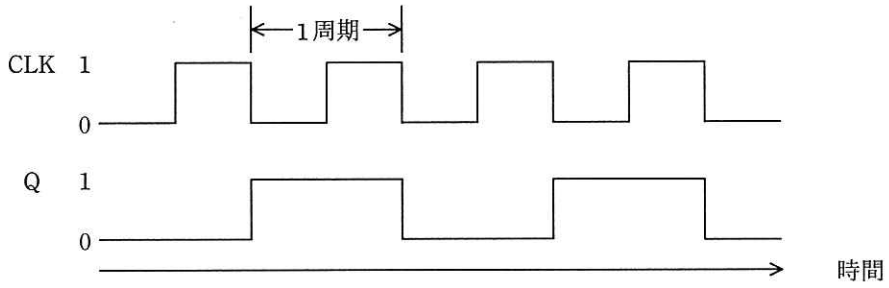


図４　２進カウンタの動作例

図４の例では，Ｑの値は，１回目の CLK の立ち下がりで０→１，２回目の CLK の立ち下がりで １→０ に変化し，以降も CLK の立ち下がりごとにこれを繰り返す。

表２から，１回目の CLK の立ち下がりのときの J，K，Q の値の組合せと，２回目の CLK の立ち下がりのときの J，K，Q の値の組合せが，表３のようであればよいことが分かる。

表３　CLK の立ち下がりのときの J，K，Q の値の組合せ

	J	K	Q
１回目の CLK の立ち下がり	1	任意$_K$	0
２回目の CLK の立ち下がり	任意$_J$	1	1

例えば，表３の任意$_J$の値を０，任意$_K$の値を１にするためには，$\overline{Q}$をＪの入力に，K の入力の値を常に １ にすればよい。（以下，（J，K）＝（$\overline{Q}$，1）と表記する）この構成例を図５に示す。

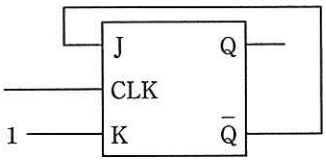


図５　２進カウンタ構成例

同様に，表3の任意$_J$と任意$_K$を組み合わせると，他の構成案として次の三つがある。

2進カウンタ構成案1　（J，K）＝（ d ）

2進カウンタ構成案2　（J，K）＝（ e ）

2進カウンタ構成案3　（J，K）＝（ f ）

d～fに関する解答群

ア　1，1　　イ　1，Q　　ウ　1，$\overline{Q}$　　エ　Q，1

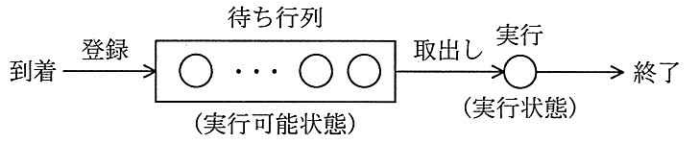
オ　Q，$\overline{Q}$　　カ　$\overline{Q}$，Q

問3　OS におけるプロセスのスケジューリングに関する次の記述を読んで，設問 1～3 に答えよ。

OS の役割の一つとして，プロセスに CPU を割り当てることがある。そして，プロセスに CPU を割り当てる順序（以下，実行順序という）を決定する方式には，次のようなものがある。ここで，プロセスが実行されるコンピュータの CPU は一つであり，CPU は同時に一つのプロセスしか実行できない。

(1)　到着順方式

プロセスを到着順に待ち行列の末尾に登録する。実行中のプロセスが終了すると，待ち行列の先頭からプロセスを一つ取り出して CPU を割り当て，実行を開始する。到着順方式を図 1 に示す。待ち行列に登録されているプロセスの状態を実行可能状態，実行中のプロセスの状態を実行状態と呼ぶ。

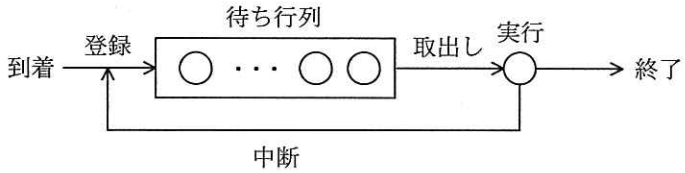


注記　○はプロセスを表す。

図 1　到着順方式

(2)　ラウンドロビン方式

プロセスを到着順に待ち行列の末尾に登録する。実行中のプロセスが終了すると，待ち行列の先頭からプロセスを一つ取り出して CPU を割り当て，実行を開始する。また，プロセスの実行中に一定時間（以下，タイムクウォンタムという）が経過したら，実行を中断して，待ち行列の末尾に再登録する。そして，待ち行列の先頭からプロセスを一つ取り出して CPU を割り当て，実行を開始する。ラウンドロビン方式を図 2 に示す。



注記　○はプロセスを表す。

図 2　ラウンドロビン方式

設問1　プロセス X を到着順方式で実行した場合のプロセスの状態の遷移について考える。プロセスの状態の遷移を図3に示す。図3において，(a)～(d)の矢印は状態の遷移を示す。プロセスXの処理順序が図4に示す①～⑤の場合，図3に示す(a)～(d)の各遷移の発生する回数の組合せとして正しい答えを，解答群の中から選べ。

なお，プロセスの実行中にデータの入出力が発生した場合，その実行を中断して待ち状態となり，データの入出力が完了したら実行可能状態となる。ここで，待ち状態のプロセスにはCPUを割り当てないものとする。

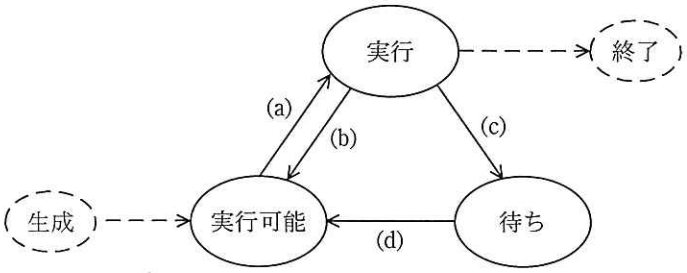


図3　プロセスの状態の遷移

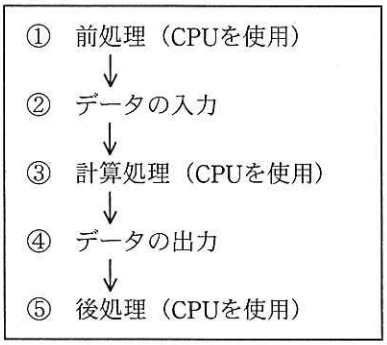


図4　プロセスXの処理順序

解答群

	(a)	(b)	(c)	(d)
ア	3	0	1	1
イ	3	0	1	2
ウ	3	0	2	2
エ	3	1	1	1
オ	3	1	1	2
カ	3	1	2	2

設問 2　ラウンドロビン方式についての，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。

四つのプロセス A～D があり，各プロセスの到着時刻と処理時間を表 1 に示す。表 1 において，到着時刻とは，プロセス A が到着して最初に待ち行列に登録された時刻からプロセス B～D が到着して最初に待ち行列に登録されるまでの経過時間である。処理時間とは，各プロセスの処理が完了するために必要な CPU の割当時間である。

表 1　プロセスの到着時刻と処理時間

プロセス名	到着時刻（ミリ秒）	処理時間（ミリ秒）
A	0	120
B	10	90
C	30	60
D	50	30

タイムクウォンタムが 20 ミリ秒の場合，プロセス D が最初に実行状態になったときには，待ち行列の先頭から［　　］の順にプロセスが登録されている。ここで，プロセス A が最初に待ち行列に登録されたとき，実行可能状態，実行状態及び待ち状態にあるプロセスはないものとし，プロセス A～D は，実行中にタイムクウォンタムの経過以外で中断することはないものとする。

なお，プロセスの登録と取出し，及び中断の処理に掛かる時間は考えないものとする。

解答群

ア A, B, C	イ A, C, B	ウ B, A, C
エ B, C, A	オ C, A, B	カ C, B, A

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。

プロセスの実行順序を決定する別の方式に残余処理時間順方式がある。残余処理時間順方式は，一定の時間ごとに実行状態と実行可能状態にあるプロセスの残余処理時間を比較し，その時間が最も短いプロセスに CPU を割り当てる方式である。ここで，残余処理時間とは，プロセスの残りの処理が完了するまでに必要な CPU の割当時間である。

表 1 に示すプロセス A～D を，残余処理時間順方式で実行した場合のターンアラウンドタイムについて考える。ターンアラウンドタイムとは，プロセスが最初に待ち行列に登録されてから処理が完了するまでの時間である。

プロセス A が登録されてから 10 ミリ秒ごとに残余処理時間を比較しながら実行したとき，プロセス A，B，C のターンアラウンドタイムはそれぞれ［ a ］ミリ秒，［ b ］ミリ秒，［ c ］ミリ秒である。図 5 には，プロセス A が登録されてから 90 ミリ秒までのプロセスの状態の遷移を記入してある。ここで，プロセス A が最初に待ち行列に登録されたとき，実行可能状態，実行状態及び待ち状態にあるプロセスはないものとし，プロセス A～D は，実行中に残余処理時間の比較結果以外で中断することはないものとする。また，プロセスが到着して待ち行列に登録された時点で，残余処理時間の比較対象となっているものとする。

なお，プロセスの登録，取出しと中断の処理，及び残余処理時間の比較処理に掛かる時間は考えないものとする。

	30	60	90	120	150	180	210	240	(ミリ秒)
A	○－－	－－－	－－－						
B	○○	－－－	－－－						
C		○○－	－－○						
D		○	○○						

注記　“○”は10ミリ秒間の実行状態を表す。
“－”は10ミリ秒間の実行可能状態を表す。

図5　残余処理時間順方式におけるプロセスの状態の遷移

a～cに関する解答群

ア　60　　イ　80　　ウ　90

エ　100　　オ　120　　カ　180

キ　300

問4　書籍を管理する関係データベースの設計及び運用に関する次の記述を読んで，設問1～4に答えよ。

ある会社の資料室では，社員（以下，利用者という）と窓口担当者の利便性を向上する目的で，所蔵する書籍を管理するデータベースを再構築することにした。この会社の資料室では，業務に関連する書籍を管理しており，利用者への貸出しも実施している。

従来のデータベースは，図1に示すとおり，所蔵する書籍と現在の貸出状況を管理する書籍表で構成されている。下線付きの項目は，主キーを表す。

書籍表

書籍番号	書籍名	著者	出版社	貸出日	返却予定日	社員番号
14030101	情報処理技術の歴史	情報太郎	情報社	null	null	null
14030102	ITで創る未来	東京都子	駒込出版	2014-09-08	2014-09-12	110028

図1　書籍表のデータ格納例

〔書籍表に関する説明〕

(1)　利用者に貸出中の書籍には，貸出日と返却予定日が格納されている。これらの項目がnullの書籍は，貸し出されていないことを表す。

(2)　社員番号には，その書籍を貸し出している利用者の社員番号が格納されている。人事系のシステムで管理している社員表と結合することで，氏名と連絡先が検索できる。

(3)　返却されたら，貸出日，返却予定日及び社員番号にはnullを設定する。

設問1　データベースの再構築に当たり，利用者と窓口担当者から要望を提出してもらった。解答群に示した要望のうち，従来の書籍表（図1）及び社員表からは**検索できない情報**を二つ選べ。

解答群

ア　現在，貸出中でない書籍の一覧

イ　現在，貸出中の書籍の一覧

ウ　現在，書籍を貸出中の利用者の連絡先

エ　書籍ごとの累積貸出回数

オ　返却予定日を過ぎても貸出中の書籍の一覧

カ　利用者ごとの貸出履歴

設問2　利用者と窓口担当者からの要望を踏まえ，データベースを図2に示す表構成で再構築して，運用を始めた。運用開始後に延滞したことがある利用者の社員番号と，延滞した書籍名を，社員番号の昇順に表示したい。次の SQL 文の［　　］に入れる正しい答えを，解答群の中から選べ。ここで，SQL 文中の CURRENT_DATE 値関数は，現在の日付を DATE 型で返却する。

書籍表

書籍番号	書籍名	著者	出版社
14030101	情報処理技術の歴史	情報太郎	情報社
14030102	ITで創る未来	東京都子	駒込出版

貸出表

貸出番号	書籍番号	社員番号	貸出日	返却予定日	返却日
140158	14030101	080172	2014-09-05	2014-09-08	2014-09-10
140159	14030102	110028	2014-09-08	2014-09-12	null

図2　再構築したデータベースの表構成とデータ格納例

〔貸出表に関する説明〕

(1)　1冊の貸出しに対して一意の貸出番号を付与し，貸出表に情報を記録する。

(2)　返却日には，返却された日付を格納する。返却日がnullの書籍は，貸出中であることを表す。

(3)　書籍が返却された後も，貸出表に記録された情報は残す。

(4)　貸出日，返却予定日及び返却日はDATE型である。

```
SELECT 貸出表.社員番号, 書籍表.書籍名
   FROM 書籍表, 貸出表
   WHERE [ a ]
   ORDER BY 貸出表.社員番号
```

解答群

ア
```
書籍表.書籍番号 = 貸出表.書籍番号 AND
(貸出表.返却日 > 貸出表.返却予定日 OR
 (貸出表.返却日 IS NULL AND 貸出表.返却予定日 < CURRENT_DATE))
```

イ
```
書籍表.書籍番号 = 貸出表.書籍番号 AND
(貸出表.返却日 BETWEEN 貸出表.貸出日 AND 貸出表.返却予定日 OR
 貸出表.返却日 IS NULL OR
 貸出表.返却予定日 < CURRENT_DATE)
```

ウ
```
書籍表.書籍番号 = 貸出表.書籍番号 AND
貸出表.返却日 != 貸出表.返却予定日
```

エ
```
書籍表.書籍番号 = 貸出表.書籍番号 AND
貸出表.返却日 IS NOT NULL
```

設問3　貸出回数が多い順に書籍番号，書籍名及び貸出回数を表示したい。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 書籍表.書籍番号, 書籍表.書籍名, [ b ] AS 貸出回数
   FROM 書籍表, 貸出表
   WHERE 書籍表.書籍番号 = 貸出表.書籍番号
   GROUP BY 書籍表.書籍番号, 書籍表.書籍名
   ORDER BY 貸出回数 DESC
```

解答群

ア　COUNT(*)　　イ　MAX(書籍表.書籍番号)

ウ　SUM(貸出表.貸出番号)　　エ　貸出表.貸出番号

オ　書籍表.書籍番号

設問4　資料室に設置されている端末からだけでなく，利用者が自席の PC からも書籍を検索できるようにしたところ，貸出実績の増加と利用者からのアクセスの急増に伴い，書籍名を入力して貸出中か否かを表示する処理で，レスポンスの低下が顕在化した。レコード件数を確認したところ，書籍表が 865 件，貸出表が 10,382 件だった。

そこで，インデックスを設定して検索性能の向上を図ることにした。インデックスの設定によって最も効果が期待できる項目として適切なものを，解答群の中から選べ。

解答群

ア　**書籍表.書籍名**　　イ　**貸出表.書籍番号**

ウ　**貸出表.社員番号**　　エ　**貸出表.貸出日**

オ　**貸出表.返却日**

問5　共通ライブラリのオブジェクト指向設計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の T 社は，自社で開発するソフトウェアの品質の安定化，開発の生産性の向上などを目的として，オブジェクト指向を用いた共通ライブラリの設計に取り組んでいる。

これまで T 社では，システム開発において社員検索機能を数多く開発してきた。この機能は，組織情報から特定の社員を探すために使用される。図 1 に示すように，左側に階層表示されている組織の一つを選択すると，右側にその組織に属する社員及び下位の組織の情報が表示されるので，目的とする社員が見つかるまで下位の組織を選択して検索する。図 1 は，左側で金融システム部を選択し，右側に金融システム部に属する社員及び下位の組織を表示している例である。

T 社では，この組織の階層構造情報（以下，組織階層という）を，デスクトップアプリケーションソフトウェア，Web ページなどのユーザインタフェースに出力するシステムを開発してきた。

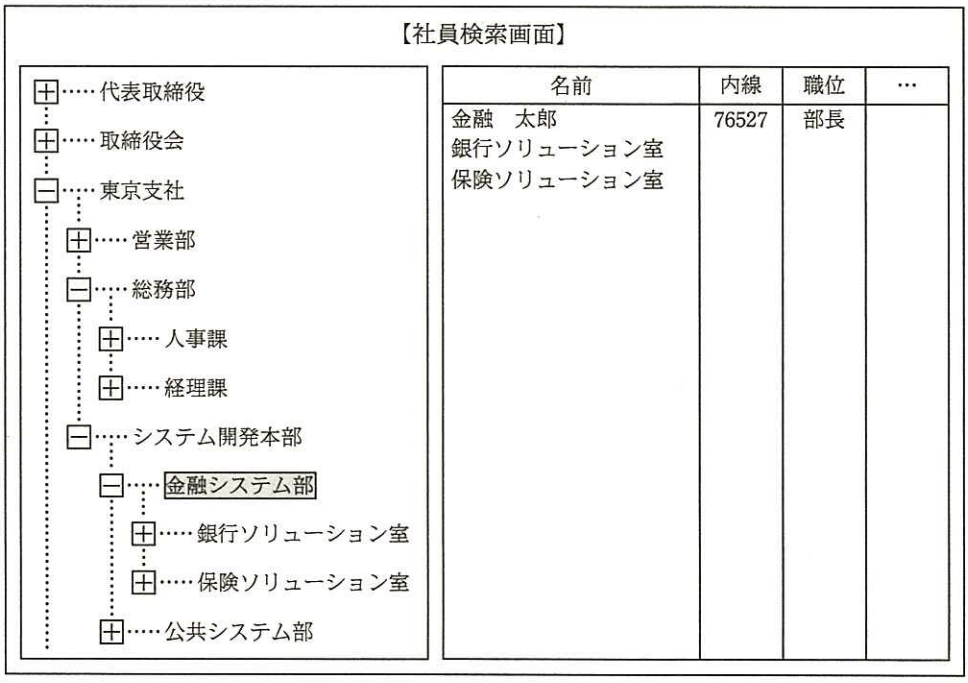


図 1　社員検索機能の表示例

組織階層の扱いは，どの開発プロジェクトにおいても類似するものが多い。そこで，組織階層を扱う機能をユーザインタフェースから切り離して，様々な開発プロジェクトで再利用できるように，共通ライブラリとして設計することにした。共通ライブラリは，オブジェクト指向で設計し，UML のクラス図で表現する。ここで，共通ライブラリの設計に当たり，共通ライブラリを利用する側のソフトウェアを一律にクライアントと呼ぶことにする。

まず，分析のためのクラス図を作成した。

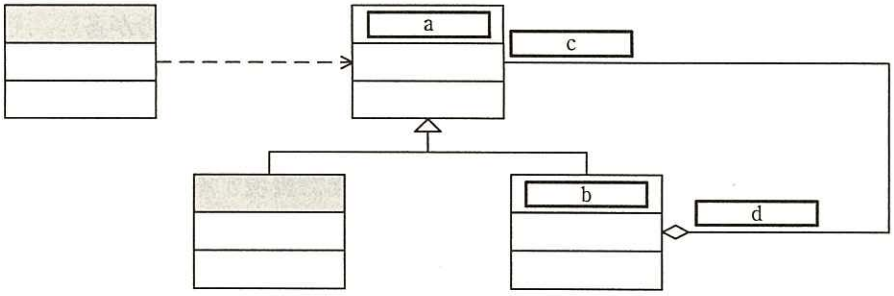
〔分析のためのクラス図の説明〕

組織階層において，ある組織が親の組織（以下，親組織という）をもつ場合，親組織は必ず一つであり，子の組織（以下，子組織という）をもつ場合，子組織の数に制限はない。また，親組織をもたない組織，子組織をもたない組織，親組織も子組織ももたない組織もある。社員は必ず一つの組織だけに属する。

(1) 組織階層は木構造で管理する。

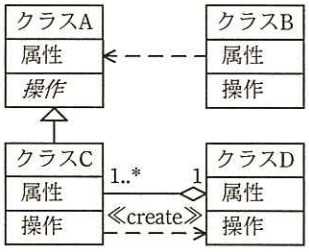
(2) クライアントが組織階層を参照する際，組織と社員を共通に扱えるようにするために，どちらも組織エントリとして管理する。

図2は，分析のためのクラス図である。



注記１　網掛けの部分は表示していない。
注記２　クラスに定義する属性，操作は記載していない。

（凡例）



長方形はクラスを表す。長方形の上段にはクラス名，中段には属性名の並び，下段には操作名の並びを記述する。属性名及び操作名は省略できる。

クラス名に≪interface≫が記載されている場合は，それがインタフェースであることを表す。

斜体の操作名は，それが抽象操作であることを表す。

クラス間を結ぶ線は関連を表す。

関連の端には，クラスのインスタンス間の多重度の範囲をx..yで表す。この凡例では，クラスDの1個のインスタンスが，クラスCの1個以上のインスタンスと関連することを示している。

関連の端の“◇”は集約を表し，凡例ではクラスCはクラスDの部分であることを示している。

“—▷”は汎化関係を表し，凡例では上位の一般的なクラスAと下位のより特殊的なクラスCとの間の分類関係を示している。

“--▷”は，インタフェースの実装を表す。

“->”はクラス間の依存関係を表し，凡例では，クラスBがクラスAに依存することを示している。

依存関係に≪create≫が記載されている場合は，インスタンスを生成することを表し，凡例ではクラスCがクラスDのインスタンスを生成することを示している。

図２　分析のためのクラス図

次に，組織エントリを管理するクラスの，設計のためのクラス図を作成した。

〔設計のためのクラス図の説明〕

組織は自身に属する組織エントリのリスト（以下，組織エントリリストという）を，属性として保持する。

この組織エントリリストを設計するためのクラス図を図３に，組織エントリリストに定義する属性と操作の説明を表１に示す。以下，操作をメソッドという。

(1)　組織エントリリストは，組織エントリをサイズが可変の配列で管理する。

(2)　クライアントは，組織エントリリストに表１のメソッドでアクセス（組織エントリの追加や削除，取出し，走査など）する。

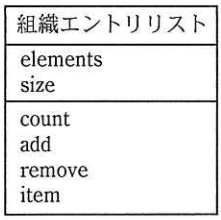


図３　組織エントリリストを設計するためのクラス図

表１　組織エントリリストに定義する属性とメソッドの説明

名前	説明
elements	組織エントリを格納するための配列。配列における要素の位置は要素番号で指定する。
size	elements の大きさ（配列の要素数）を表す。
count	組織エントリリストに登録されている組織エントリの数を返す。
add	組織エントリと要素番号を受け取り，受け取った要素番号の位置に組織エントリを挿入する。要素番号は省略可能であり，省略された場合は，組織エントリリストの最後に組織エントリを追加する。
remove	要素番号を受け取り，その要素番号の組織エントリを組織エントリリストから削除する。
item	要素番号を受け取り，その要素番号にある組織エントリを返す。

設問１　図 2 の分析のためのクラス図の中の［　　］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　親組織　　イ　クライアント　　ウ　子組織　　エ　社員

オ　組織　　カ　組織エントリ　　キ　組織階層

c，dに関する解答群

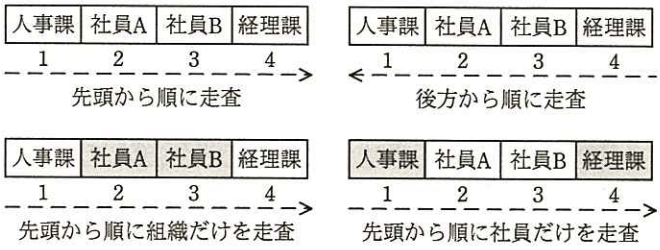
ア　0　　イ　0..1　　ウ　0..*　　エ　1　　オ　1..*

設問２　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。

共通ライブラリの社内レビューを実施したところ，組織エントリリストのクラス設計に対して次の指摘が挙がった。

〔指摘の内容〕

組織エントリリストから組織エントリを取り出すとき，クライアントが要素番号を指定する必要がある。全ての組織エントリを取り出すときや，特定条件を満たす組織エントリだけを取り出したいときなど，クライアント側の実装依存が大きくなる。図4にクライアントの走査の例を示す。



注記　網掛けの部分は読み飛ばすことを意味している。

図4　クライアントの走査の例

共通ライブラリは，組織エントリリストの内部構造をクライアントに意識させず，組織エントリリストの走査を拡張できるように改善すべきである。

この指摘を受け，組織エントリリストのクラス設計を次の設計方針に従って見直すことにした。

〔設計方針〕

(1)　組織エントリリストは，より汎用的にするために，任意のオブジェクトを管理できるようにする。

(2)　リストを走査する処理は，別のクラス（以下，イテレータという）に与える。リストでは，このイテレータを生成して［　e　］をイテレータに登録することでイテレータがリストを走査できるようにする。

(3)　共通ライブラリで提供するイテレータは，先頭から順に全ての要素を走査するメソッドを提供する。

(4)　アクセス方法の統一を目的として，必要なメソッドをインタフェースとして定義する。クライアントは，必ずこのインタフェースを用いてリストを走査することとする。

(5)　クライアントは，共通ライブラリで提供するメソッドと異なる走査をした

い場合は，［　f　］する。

図5は見直し後の設計のためのクラス図，表2は図5のクラス図の説明である。

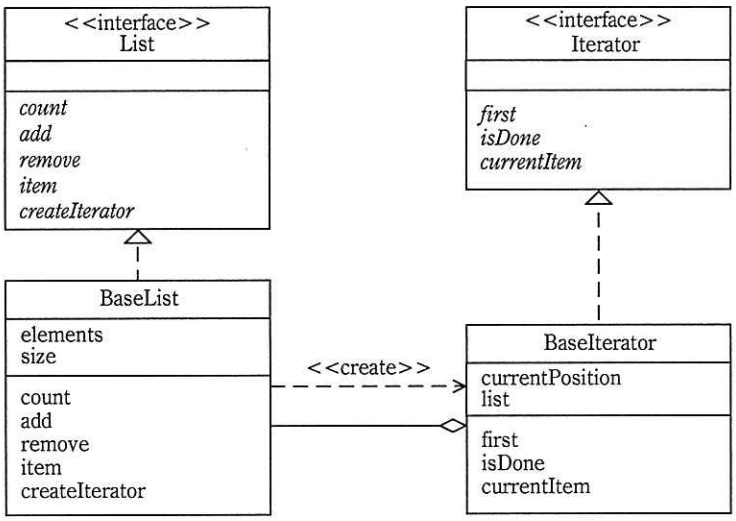


図5　見直し後の設計のためのクラス図

表2　図5のクラス図の説明

名前	説明
List	元の組織エントリリストに定義していたメソッド及びイテレータを生成するためのメソッド createIterator を定義する。
BaseList	元の組織エントリリストである。リストを表現するクラスに共通となる属性とメソッドを提供する。 createIterator は，リストを走査するイテレータを生成して，クライアントに返す。
Iterator	リストの要素へのアクセスや，リストを走査するためのメソッド first，isDone，currentItem を定義する。
BaseIterator	リストの先頭から順に全ての要素を走査する機能を提供する。走査対象のリスト list 及び走査中の位置 currentPosition を保持し，クライアントからの要求に応じて適切な要素を返す。 first は currentPosition を先頭に移動する。isDone はまだ走査対象が存在するかどうかをクライアントに返す。currentItem は currentPosition の位置にある要素をクライアントに返した後，currentPosition を走査方向に一つ移動する。

eに関する解答群

ア　クライアント　　イ　属性　　ウ　メソッド　　エ　リスト自身

fに関する解答群

ア　BaseListとBaseIteratorを継承して処理をオーバライド

イ　BaseListとBaseIteratorを継承してメソッドを追加

ウ　BaseListとBaseIteratorを実装して処理をオーバライド

エ　BaseListとBaseIteratorを実装してメソッドを追加

オ　ListとIteratorを継承して処理をオーバライド

カ　ListとIteratorを継承してメソッドを追加

キ　ListとIteratorを実装して処理をオーバライド

ク　ListとIteratorを実装してメソッドを追加

問6　サービスデスクにおける問合せ対応に関する次の記述を読んで，設問1～3に答えよ。

S社では，販売情報システムを更新する（以下，システム更新という）とともに，自社のサービスデスクを利用して販売情報システムに関する社員からの問合せに対応する。サービスデスクは，社員からの問合せの内容を表1に示す区分に従って分類し，ログインと操作に関する問合せについては自ら回答する。ログインと操作以外の問合せについては，受付日時などの記録を行い，担当部門に引き継ぐ。引継ぎを受けた担当部門は問い合わせた社員に回答するとともに，回答の内容や回答を完了した日時をサービスデスクに報告する。

問合せの受付日時，回答完了日時，回答内容などについては，サービスデスクが問合せ台帳で管理する。

表1　問合せの区分及び担当部門

問合せの内容	区分	担当部門
ログインIDとパスワードに関する問合せ	ログイン	サービスデスク
メニュー，ボタンなどの操作方法に関する問合せ	操作	サービスデスク
販売情報システムの起動に関する問合せ	接続	システム課
販売目標データに関する問合せ	目標管理	企画課
販売実績データに関する問合せ	販売実績	営業課
上記以外の問合せ	その他	営業課

サービスデスクでは，過去の類似の事例を参考にして，システム更新後1週間の問合せ総数を300，そのうちサービスデスクが回答する問合せ（以下，サービスデスク問合せという）数を90と見積もり，問合せ対応のために臨時に要員を1名増員することにした。過去の類似の事例から，問合せ総数は時間の経過とともになだらかに減少することが分かっているので，1週間の問合せ総数が150以下かつ1週間のサービスデスク問合せ数が30以下になった場合（以下，問合せ数条件という）は，その翌週から要員を1名減らして，元の要員数に戻すことにした。ただし，システム更新後第5週に新機能の追加導入（以下，システム追加導入という）を予定しており，それに伴って第5週の問合せ数が増えると想定されるので，第4週になって初めて問合せ数条件を満たしたとしても第5週は第4週と同じ要員数とし，第5週以降に問合せ数条件を満たした段階でその翌週に元の要員数に戻すことにした。

また，サービスデスクでは，サービスデスク問合せについては問合せの受付から 30 分以内で回答を完了することを目標（以下，サービスデスク目標という）とした。

設問 1　表 2 は，システム更新後のある日の午前中にあった販売情報システムに関する問合せを対象に，問合せ台帳から抽出したものである。表 2 のサービスデスク目標の達成率（サービスデスク目標を達成したサービスデスク問合せ数÷サービスデスク問合せ数）を，解答群の中から選べ。ここで，達成率は小数第 3 位を四捨五入し，パーセントで表している。

表 2　販売情報システムに関する問合せ

受付番号	受付日時	回答完了日時	区分	問合せ内容	担当部門	…
11842	2014/10/4 8:35	2014/10/4 8:38		パスワードを変更するにはどうすればよいか		…
11844	2014/10/4 8:44	2014/10/4 8:53		販売情報システムが起動できない		…
11860	2014/10/4 10:24	2014/10/4 10:50	操作	メニュー画面への戻り方が分からない		…
11863	2014/10/4 11:02	2014/10/4 11:41	目標管理	販売目標データが最新になっていない		…
11865	2014/10/4 11:34	2014/10/4 12:25	ログイン	ログイン ID を他の社員と共有して使ってよいか		…
11866	2014/10/4 11:55	2014/10/4 12:07		販売情報システムにログインしようとしたが，パスワードを忘れた	サービスデスク	…

注記　網掛けの部分は表示していない。

解答群

ア　25　　イ　33　　ウ　50　　エ　67　　オ　75　　カ　100

設問 2　表 3 に示すシステム更新後の問合せ数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。ここで，ログインと操作の行にある下段の括弧内の数字は，ログインと操作の当該週における問合せ数のうち，サービスデスク目標を達成した数である。

表３　販売情報システムに関する問合せ数

区分	第１週	第２週	第３週	第４週
ログイン	40 (28)	20 (15)	5 (4)	3 (3)
操作	25 (20)	22 (17)	18 (16)	20 (17)
接続	12	8	2	12
目標管理	90	45	55	48
販売実績	82	38	10	4
その他	15	11	9	8
合計	264	144	99	95

システム更新の当初は問合せ数が多かったものの，第 4 週の問合せ数が第 1 週と比べて半数以下となった区分は，［　a　］である。

担当部門別に見ると，第４週の問合せ数が最も多いのは［　b　］である。

販売情報システムに関する問合せのためのサービスデスクの要員は，［　c　］。

また，週別，区分別に計算したサービスデスク目標の達成率は，［　d　］。

aに関する解答群

ア　操作と販売実績　　イ　操作と目標管理

ウ　目標管理と販売実績　　エ　ログインと操作

オ　ログインと販売実績　　カ　ログインと目標管理

bに関する解答群

ア　営業課　　イ　企画課　　ウ　サービスデスク　　エ　システム課

cに関する解答群

ア　第３週から元の要員数に戻している　　イ　第４週から元の要員数に戻している

ウ　第５週から元の要員数に戻す　　エ　第５週も１名増員したままである

dに関する解答群

ア　ログイン，操作とも毎週上がっている

イ　ログイン，操作とも週によって上がり下がりがある

ウ　ログインは週によって上がり下がりがあるものの，操作は毎週上がっている

エ　ログインは毎週上がっているものの，操作は週によって上がり下がりがある

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。

サービスデスクでは，回答を完了するまでに掛かる時間（以下，回答完了時間という）を短縮するために，サービスデスク問合せ以外の問合せで過去に同種の問合せがあった場合は，担当部門に連絡するのではなく，問合せ台帳を基にサービスデスクで回答すること（以下，サービスデスク改善という）を検討している。表4は，第4週の問合せ数と，そのうちで過去に同種の問合せがあった数を示している。

表4　第4週の問合せ数と，そのうちで過去に同種の問合せがあった数

区分	第4週の問合せ数	過去に同種の問合せがあった数
ログイン	3	1
操作	20	12
接続	12	5
目標管理	48	16
販売実績	4	4
その他	8	2
合計	95	40

サービスデスクでは，システム追加導入後の第5週の問合せの状況を次のように仮定した。

(1)　第5週の操作，目標管理及び販売実績に区分される問合せ数は第4週の25%増である。

(2)　第5週のログイン，接続及びその他に区分される問合せ数は第4週と同数である。

(3) 過去に同種の問合せがあった数の割合は第5週と第4週で同じである。

(4) サービスデスク改善によって，サービスデスクで回答することになる問合せについて，1件当たりの回答完了時間を平均10分短縮できる。

このとき，第5週の想定される問合せ総数は［ e ］であり，第5週の回答完了時間はサービスデスク改善によって［ f ］分短縮できる。

サービスデスクでは，よくある問合せとその解決策をFAQとして整備し，社員に対して利用を推奨することにした。第5週に想定される過去と同種の問合せの50%がFAQの対象として整備されたものに該当し，さらにFAQとして整備されたものは問合せ数が50%に減らせると仮定すると，FAQの利用によって第5週の問合せ総数のうち，減らせる数は［ g ］であると想定できる。

eに関する解答群

ア 95　イ 108　ウ 111　エ 113　オ 116　カ 119

fに関する解答群

ア 320　イ 330　ウ 400　エ 470　オ 480　カ 500

gに関する解答群

ア 8　イ 12　ウ 16　エ 24　オ 32　カ 48

問7　受発注システムの改修に関する次の記述を読んで，設問1，2に答えよ。

事務用品卸売業のC社では，取引先から注文を受けたときに指定された納期（以下，要求納期という）どおりに納品できない場合があり，実際に納品する日程（以下，約束納期という）の調整を行っている。そこで，受発注システム（以下，システムという）を改修し，要求納期どおりに納品できる商品の割合（以下，要求納期遵守率という）を高めて，顧客満足度を上げることにした。現状のシステムを利用した受注，発注及び入庫に関する処理は次のとおりである。

〔受注に関する処理〕

(1) 営業担当は取引先からの注文を受け，受け付けた順に取引先，受注日，品名，数量，要求納期などの受注情報をシステムに入力する。受注日から要求納期までの期間を，納入リードタイムという。

(2) システムは，受注情報に対し，通し番号で受注番号を自動採番し，受注情報の品名と数量を用いて，引当可能在庫に引当て（以下，在庫引当という）を行う。

(3) 在庫引当は，引当可能在庫の数量から受注情報の数量を減らして，引当可能在庫の数量を更新する。

(4) 引当可能在庫の数量が受注情報の数量に満たない場合は，システムは引当可能在庫の数量だけを引き当て，不足した数量は，仕入先から新たに商品が入庫され，入庫処理を行った後，受注情報を入力した順に在庫引当を行う。

(5) 引き当てられた商品は，引き当てられたその日のうちに取引先に納品することができる。

(6) 要求納期どおりに納品できる場合は要求納期に，調整を行った場合は約束納期に，取引先に納品する。

(7) 営業担当が取引先から注文を受け付ける時間（以下，注文受付時間という）は9時から16時までであり，注文受付時間内に，営業担当は受注情報を入力し，システムは在庫引当を行う。

〔発注に関する処理〕

(1) 商品の発注方式は定量発注方式で行っている。定量発注方式は，引当可能在庫

の数量があらかじめ設定した数量（以下，発注点という）以下になったときに，一定数量（以下，発注量という）を発注して在庫を補充する方法である。商品ごとに発注点と発注量をシステムに設定し，引当可能在庫の数量を管理している。

(2) システムは品名，発注量などの情報を用いて発注処理を行う。発注処理は，注文受付時間の終了後に行われる。

(3) 発注処理によって，仕入先に発注書が自動で送信される。

(4) 発注書を送信した日から商品の入庫日までの期間（以下，調達リードタイムという）は，あらかじめ，商品ごとに仕入先と取り決めている。仕入先は十分な商品在庫をもち，欠品による納期遅れはないものとする。

〔入庫に関する処理〕

(1) 入庫処理は注文受付時間内に行う。倉庫担当は，仕入先から入庫された商品の品名，数量などの情報をシステムに入力して入庫処理を行う。

(2) システムは，引当可能在庫の数量に入庫された数量を加算して，引当可能在庫の数量を更新する。

設問1 C社の業務に関する説明として適切な答えを，解答群の中から選べ。

解答群

ア 仕入先への発注は，商品ごとに一定の日にち間隔で行われる。

イ 受注情報を入力すると，納入リードタイムに関係なく，システムは在庫引当を行う。

ウ 引当可能在庫の数量がゼロの場合，受注情報を入力することはできない。

エ 引当可能在庫の数量はマイナスになることがある。

設問2 システムの改修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。

C社では，システムの改修に当たり，従来の定量発注方式に加えて，新たな発注方式を追加することによって，要求納期遵守率を上げることにした。

この実現方法を検討するために，商品 X の受注情報と引当可能在庫の数量を分析した。商品 X は，C 社の代表的な商品であり，商品 X の分析結果は，他の商品にも適用できるものとする。

商品 X の 7 月の受注情報及び引当可能在庫の数量を表 1 に示す。

表 1　商品 X の 7 月の受注情報及び引当可能在庫の数量

受注番号	受注日	数量（個）	要求納期	引当可能在庫の数量（在庫引当後）
130121	7 月 3 日	35	7 月 11 日	235
130127	7 月 7 日	30	7 月 11 日	205
130143	7 月 8 日	30	7 月 10 日	175
130144	7 月 12 日	40	7 月 20 日	
130148	7 月 17 日	25	7 月 24 日	
130152	7 月 19 日	35	7 月 23 日	
130156	7 月 23 日	45	7 月 25 日	
130168	7 月 23 日	40	7 月 25 日	
130169	7 月 24 日	35	7 月 25 日	
130175	7 月 26 日	25	7 月 27 日	

注記　網掛けの部分は表示していない。

〔商品 X の情報〕

(1)　調達リードタイムは 7 日間である。

(2)　発注点は 90 個，発注量は 300 個である。

(3)　6 月末の引当可能在庫の数量は 270 個である。

〔商品 X の受注情報の分析結果〕

(1)　定量発注方式による発注処理は［　a　］に行われた。

(2)　取引先の要求納期に納品できなかった受注情報の受注番号は［　b　］である。

(3)　納入リードタイムが調達リードタイム以上の受注情報は 3 件ある。

現状のシステムは，納入リードタイムが調達リードタイム以上である場合でも在庫引当が行われるので，要求納期遵守率を下げる場合があることが分かった。

例えば，引当可能在庫の数量が100個のときに，納入リードタイムが9日で数量が100個の受注があった場合，受注した時点で仕入先に発注すれば，7日後に倉庫に入庫された商品を要求納期に納品することができる。しかし，現状のシステムでは，引当可能在庫の数量はゼロとなり，［　c　］。

そこで，受注情報の納入リードタイムが調達リードタイム以上である場合，次のような処理ができるようにシステムを改修する。

(1)　［　d　］。

(2)　システムは，受注情報の数量を，新たな発注方式で発注処理する。

(3)　システムは，個別に発注処理されて仕入先から入庫された商品を引当可能在庫に加算せず，該当する受注情報の取引先に納品する。

システム改修後の，商品Xの7月の受注情報及び引当可能在庫の数量の試算結果を表2に示す。

ただし，このシステム改修に当たっては，受注情報の納入リードタイムが調達リードタイム以上である場合が長期にわたって連続すると，引当可能在庫があるにもかかわらず在庫引当されないという課題があるので，8月以降も受注情報と引当可能在庫の数量を分析した上でシステム改修の判断をするものとした。

表2　商品Xの7月の受注情報及び引当可能在庫の数量の試算結果

受注番号	受注日	数量（個）	要求納期	引当可能在庫の数量（在庫引当後）
130121	7月3日	35	7月11日	270
130127	7月7日	30	7月11日	240
130143	7月8日	30	7月10日	210
130144	7月12日	40	7月20日	
130148	7月17日	25	7月24日	
130152	7月19日	35	7月23日	
130156	7月23日	45	7月25日	
130168	7月23日	40	7月25日	［　e　］
130169	7月24日	35	7月25日	
130175	7月26日	25	7月27日	

注記　網掛けの部分は表示していない。

aに関する解答群

ア　7月17日　　イ　7月18日

ウ　7月19日　　エ　7月20日

bに関する解答群

ア　130168と130169　　イ　130168と130169と130175

ウ　130168と130175　　エ　130169と130175

cに関する解答群

ア　倉庫にある商品の数量もゼロになってしまう

イ　引き当てられた100個の商品は，倉庫に商品があるにもかかわらず，発注した商品の入庫日よりも前の要求納期をもつ別の受注に在庫引当ができない

ウ　引き当てられた100個の商品は，その日に納品されてしまう

エ　引き当てられた100個の商品は，別の受注に在庫引当ができてしまう

dに関する解答群

ア　営業担当が受注情報を入力しても，システムは在庫引当を行わない

イ　営業担当が受注情報を入力すると，システムは受注情報の数量を引当可能在庫の数量に加算する

ウ　営業担当が引当可能在庫の数量から受注情報の数量を減らし，引当可能在庫の数量を更新する

エ　システムが在庫引当を行わないように，営業担当は受注情報を入力しない

eに関する解答群

ア　0　　イ　65

ウ　90　　エ　125

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1，2に答えよ。

二つの文字列の差異を測る指標に編集距離がある。編集距離の概念は，文書比較や検索キーワードの候補の提示などに用いられている。編集距離とは，1文字の追加操作又は削除操作を繰り返し適用し，ある文字列を別の文字列に変換するのに必要な最小の操作回数である。例えば文字列"abcabba"を文字列"cbabac"に変換する場合，図1に示す操作1～5によって変換が完了する。図1は，最小の操作回数で変換する一例を示しており，編集距離は5となる。

```
abcabba  →  bcabba（1文字目のaを削除）          操作1
         →  cabba  （1文字目のbを削除）          操作2
         →  cbabba（1文字目のcの後ろにbを追加）  操作3
         →  cbaba  （5文字目のbを削除）          操作4
         →  cbabac（5文字目のaの後ろにcを追加）  操作5
```

図1　文字列"abcabba"を文字列"cbabac"に変換する場合の例

関数CalcEditDistanceは，二つの文字列間の編集距離を返す関数である。

(1)　変換元の文字列をStr1[]，変換先の文字列をStr2[]とする。また，配列の添字は0から始まり，文字列Str1[]のi番目の文字はStr1[i − 1]と表記する。したがって，Str1[] = "abcabba"の場合，1番目の文字 = Str1[0] = "a"，2番目の文字 = Str1[1] = "b"となる。Str2[]についても同様である。

(2)　関数CalcEditDistanceは，エディットグラフと呼ばれるグラフの最短距離取得問題の考え方に基づいて，編集距離を求めている。編集距離の求め方は次のとおりである。

①　次の手順で，x y 平面上にエディットグラフを作成する。ここで，Str1LenはStr1[]の文字数，Str2LenはStr2[]の文字数である。

(a)　0 ≦ X ≦ Str1Lenを満たす全ての整数Xに対して，点(X, 0)から点(X, Str2Len)に線分を引く。

(b) 0 ≦ Y ≦ Str2Len を満たす全ての整数 Y に対して，点 (0, Y) から点 (Str1Len, Y) に線分を引く。

(c) 0 ≦ X ＜ Str1Len，0 ≦ Y ＜ Str2Len を満たす全ての整数 X，Y の組に対して，Str1[X] と Str2[Y] が同一の文字の場合，点 (X, Y) から点 (X + 1, Y + 1) に線分を引く。

② エディットグラフを構成する線分をたどって，点 (0, 0) から点 (Str1Len, Str2Len) へ移動する経路を考える。点 (X, Y) から点 (X + 1, Y) 又は点 (X, Y + 1) への移動距離を 1，点 (X, Y) から点 (X + 1, Y + 1) への移動距離を 0 としたときの，点 (0, 0) から点 (Str1Len, Str2Len) までの最短移動距離が編集距離となる。

(3) Str1[] ＝ "abcabba"，Str2[] ＝ "cbabac" の場合のエディットグラフを図 2 の左に示す。この場合に，最短移動距離となる経路の一つを図 2 の右に → で示す。

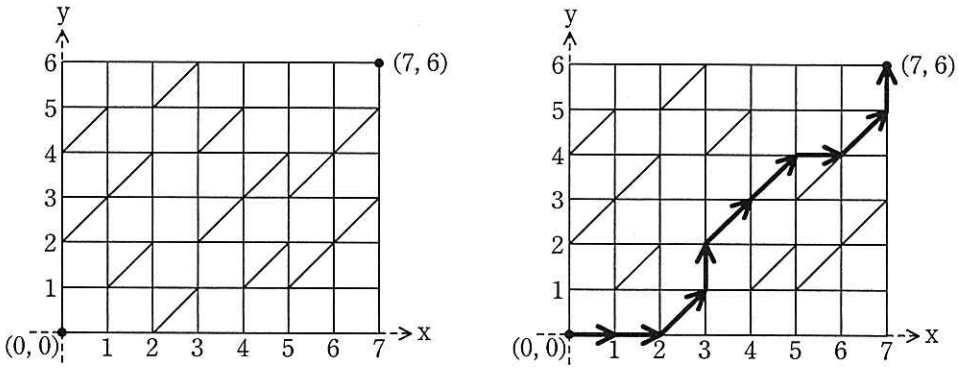


図 2　Str1[] ＝ "abcabba"，Str2[] ＝ "cbabac"の場合のエディットグラフ（左）と最短移動距離となる経路の例（右）

(4) 関数 CalcEditDistance では，点 (0, 0) から点 (X, Y) への最短移動距離を D[X, Y] に求めている。D[X, Y] は，既に算出されている D[X − 1, Y − 1]，D[X, Y − 1]，D[X − 1, Y] を用いて求めることができる。これによって編集距離を算出している。

〔関数 CalcEditDistance の引数と返却値〕

関数 CalcEditDistance の引数の仕様は，次のとおりである。各配列の添字は，0から始まる。

引数名／返却値	データ型	意味
Str1[]	文字型	変換元の文字列が格納されている1次元配列
Str1Len	整数型	変換元の文字列の長さ（1以上）
Str2[]	文字型	変換先の文字列が格納されている1次元配列
Str2Len	整数型	変換先の文字列の長さ（1以上）
返却値	整数型	変換元と変換先の文字列間の編集距離

関数 CalcEditDistance では，次の関数 Min を使用している。

〔関数 Min の仕様〕

引数として与えられた二つ以上の整数値の中で最も小さい値を返却値とする。

〔プログラム〕

```
○整数型: CalcEditDistance(文字型: Str1[], 整数型: Str1Len,
                           文字型: Str2[], 整数型: Str2Len)
○整数型: D[Str1Len + 1, Str2Len + 1], X, Y

■ X: 0, X ≦ Str1Len, 1
  • D[X, 0] ← X
■
■ Y: 0, Y ≦ Str2Len, 1
  • D[0, Y] ← Y
■
■ X: 1, X ≦ Str1Len, 1
  ■ Y: 1, Y ≦ Str2Len, 1
      ┌ a ┐
      • D[X, Y] ← Min(D[X − 1, Y − 1],
                      D[X, Y − 1] + 1, D[X − 1, Y] + 1) ◀━━ α
      ─────
      • D[X, Y] ← Min(D[X, Y − 1] + 1, D[X − 1, Y] + 1) ◀━━ β
  ■
■
• return ( ┌ b ┐ )      /* 編集距離を返却値として返す */
```

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　Str1[X − 1] = Str2[Y − 1]

イ　Str1[X − 1] ≠ Str2[Y − 1]

ウ　Str1[X] = Str2[Y]

エ　Str1[X] ≠ Str2[Y]

オ　Str1[X − 1] = Str1[X] and Str2[Y − 1] = Str2[Y]

カ　Str1[X − 1] ≠ Str1[X] and Str2[Y − 1] ≠ Str2[Y]

b に関する解答群

ア　D[0, Str2Len]

イ　D[Str1Len, 0]

ウ　D[Str1Len, Str2Len]

エ　D[Str1Len, Str2Len] + 1

オ　D[Str1Len − 1, Str2Len − 1]

カ　D[Str1Len − 1, Str2Len − 1] + 1

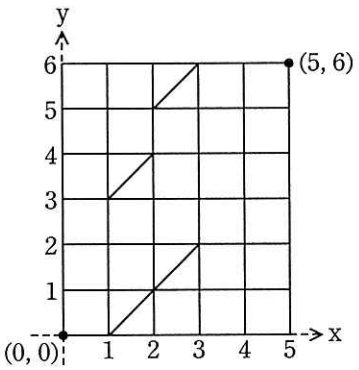
設問2　次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。

Str1[] = "peace", Str2[] = "people"の場合のエディットグラフは，［ c ］となる。

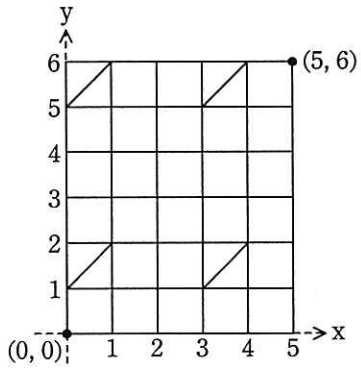
CalcEditDistance("peace", 5, "people", 6) を実行した場合，関数 CalcEditDistance が終了するまでに行 α は［ d ］回実行され，行 β は［ e ］回実行される。また，返却値は［ f ］となる。ここで，関数 CalcEditDistance 中の［ a ］，［ b ］には正しい答えが入っているものとする。

cに関する解答群

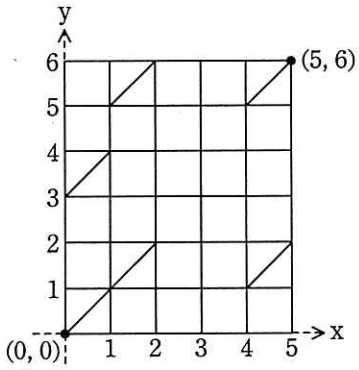
ア



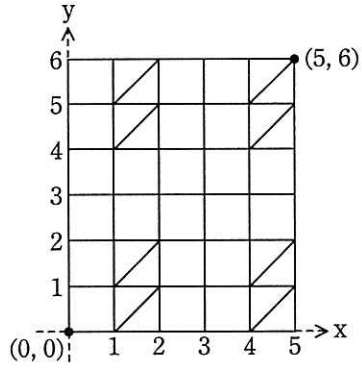
イ



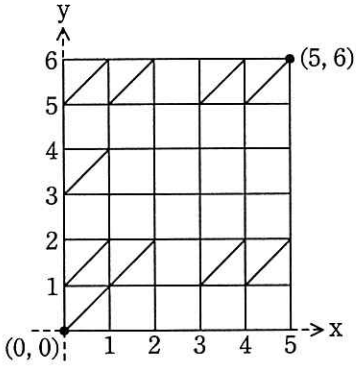
ウ



エ



オ



dに関する解答群

ア 2	イ 4	ウ 6	エ 8
オ 10	カ 12	キ 14	ク 16

eに関する解答群

ア 14	イ 16	ウ 18	エ 20
オ 22	カ 24	キ 26	ク 28

fに関する解答群

ア 4	イ 5	ウ 6	エ 7
オ 8	カ 9	キ 10	ク 11

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。

B社では，セキュリティ管理のために，システムファイルから定期的に，システムの運用・保守用の利用者ID一覧を作成し，ファイルで保管している。

これらの利用者IDに付加できる特権には，システムファイルの更新などができるシステム特権（以下，特権Sという）及びバックアップ作業などのために全てのファイルを参照できるオペレーション特権（以下，特権Oという）の2種類がある。

B社では，セキュリティ管理強化の一環で，利用者IDの追加・削除や特権の付加・解除が，申請に基づいて正しくシステムに反映されているかどうかを検証することになった。そのために，最新及び1世代前の利用者ID一覧を比較して，この間の利用者ID及び特権の登録内容の差異を印字するプログラムを作成する。

〔プログラムの説明〕

(1)　最新の利用者ID一覧をファイル`NewFile`から，1世代前の利用者ID一覧をファイル`OldFile`から，それぞれ読み込む。レコードは利用者IDの昇順に整列されている。

(2)　1レコードは，利用者ID（1～8桁），利用者名（1～10桁），属性（1桁）及び最終使用日（8桁）の4項目から成る。各項目は，空白文字で区切られている。

①　利用者ID及び利用者名は，英数字から成る文字列である。

②　属性は，次に示す内容の8ビット長のビット列 | 0 | 1 | 0 | 0 | s | o | g | r | である。

上位4ビット：　固定値0100
ビットs：　特権Sが付加されていれば1，付加されていなければ0
ビットo：　特権Oが付加されていれば1，付加されていなければ0
ビットg及びr：　その他の属性

③　最終使用日は，数字から成る文字列で，その利用者IDで最後にログインした年月日を表す。登録後，一度もログインしていない場合は，全桁が“0”である。

(3) 利用者 ID 及び特権の登録内容の差異は，次のように印字する。

① NewFile 中にあって OldFile 中にない利用者 ID の場合

利用者 ID，利用者名の後に“利用者 ID 追加”と印字する。この利用者 ID に特権 S が付加されていれば“特権 S 付加”，特権 O が付加されていれば“特権 O 付加”を追加印字する。

② OldFile 中にあって NewFile 中にない利用者 ID の場合

利用者 ID，利用者名の後に“利用者 ID 削除”と印字する。この利用者 ID に特権 S が付加されていれば“特権 S 解除”，特権 O が付加されていれば“特権 O 解除”を追加印字する。

③ NewFile 及び OldFile の両方にあり，特権 S と特権 O の少なくとも一方の付加状況が変わった利用者 ID の場合

利用者 ID，利用者名の後に“特権 S 付加”，“特権 S 解除”，“特権 O 付加”，“特権 O 解除”の該当する全てを印字する。

(4) 入力ファイル NewFile 及び OldFile のデータ例を図 1 に，図 1 のデータ例を用いた実行結果を図 2 に，それぞれ示す。

ファイル NewFile のデータ例

利用者 ID	利用者名	属性	最終使用日
AE001	UserE1	41	20140419
AE002	UserE2	48	20141014
AE003	UserE3	48	20140716
AP005	UserP5	46	20141015
AP006	UserP6	44	00000000

ファイル OldFile のデータ例

利用者 ID	利用者名	属性	最終使用日
AE001	UserE1	40	20140419
AE002	UserE2	44	20140712
AE003	UserE3	4C	20140716
AP004	UserP4	45	20140715
AP005	UserP5	44	20140713

注記 1　属性の値（網掛け部分）は，16 進数で表示している。
注記 2　見出し行は，各ファイルには含まれないものとする。

図 1　入力ファイル NewFile 及び OldFile のデータ例

利用者 ID	利用者名	登録内容の差異
AE002	UserE2	特権 S 付加　特権 O 解除
AE003	UserE3	特権 O 解除
AP004	UserP4	利用者 ID 削除　特権 O 解除
AP006	UserP6	利用者 ID 追加　特権 O 付加

注記　見出し行は，事前に印字されているものとする。

図 2　図 1 のデータ例を用いた実行結果

(5) 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。

〔プログラム〕

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

#define  BitS  0x08
#define  BitO  0x04
#define  BitR  0x01

FILE          *NewFile,    *OldFile;
int            NewEof,      OldEof;
char           NewID[9],    OldID[9];
char           NewName[11],OldName[11];
unsigned char  NewAttr,     OldAttr;
char           NewDate[9],  OldDate[9];

void ReadNewRecord() {

   fscanf(NewFile, "%s %s %c %s", NewID, NewName, &NewAttr, NewDate);
   if (feof(NewFile) != 0) {
      NewEof = EOF;                  /* EOF:  負の整数定数 */
      strcpy(NewID, "\xFF");         /* \xFF: 8 ビット符号の最大値 */
   }
}

void ReadOldRecord() {

   fscanf(OldFile, "%s %s %c %s", OldID, OldName, &OldAttr, OldDate);
   if (feof(OldFile) != 0) {
      OldEof = EOF;
      strcpy(OldID, "\xFF");
   }
}

void main() {

   NewFile = fopen("NewFile", "rb");
   OldFile = fopen("OldFile", "rb");
   NewEof = 0;
   OldEof = 0;
   ReadNewRecord();
   ReadOldRecord();
```

```
   while ( [ a ] ) {
      if (strcmp(NewID, OldID) == 0) {
         if ( [ b ] ) {
            printf("\n%-8s  %-10s", NewID, NewName);
            if ((NewAttr & BitS) > (OldAttr & BitS))
               printf("  特権 S 付加");
            if ((NewAttr & BitS) < (OldAttr & BitS))
               printf("  特権 S 解除");
            if ((NewAttr & BitO) > (OldAttr & BitO))
               printf("  特権 O 付加");
            if ((NewAttr & BitO) < (OldAttr & BitO))
               printf("  特権 O 解除");
         }
         [ c ]
α    }
      else {
         if ( [ d ] ) {
            printf("\n%-8s  %-10s  利用者 ID 追加", NewID, NewName);
            if ((NewAttr & BitS) == BitS)  printf("  特権 S 付加");
            if ((NewAttr & BitO) == BitO)  printf("  特権 O 付加");
            ReadNewRecord();
         }
         else {
            printf("\n%-8s  %-10s  利用者 ID 削除", OldID, OldName);
            if ((OldAttr & BitS) == BitS)  printf("  特権 S 解除");
            if ((OldAttr & BitO) == BitO)  printf("  特権 O 解除");
            ReadOldRecord();
         }
      }
   }
   fclose(OldFile);
   fclose(NewFile);
}
```

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　`(NewEof != EOF) && (OldEof != EOF)`

イ　`(NewEof != EOF) || (OldEof != EOF)`

ウ　`NewEof != OldEof`

エ　`NewEof == OldEof`

bに関する解答群

ア　`((NewAttr & OldAttr) & (BitS + BitO)) != 0x00`

イ　`((NewAttr | OldAttr) & (BitS + BitO)) != 0x00`

ウ　`(NewAttr & (BitS + BitO)) != (OldAttr & (BitS + BitO))`

エ　`(NewAttr | (BitS + BitO)) != (OldAttr | (BitS + BitO))`

cに関する解答群

ア
```
ReadNewRecord();
```

イ
```
ReadNewRecord();
ReadOldRecord();
```

ウ
```
ReadOldRecord();
```

dに関する解答群

ア　`(NewAttr & (BitS + BitO)) != 0x00`　　イ　`NewAttr > OldAttr`

ウ　`strcmp(NewID, OldID) < 0`　　エ　`strcmp(NewID, OldID) > 0`

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。

利用者 ID が有効に使用されているかどうかを検証するために，一定期間未使用，又は現在使用不可の利用者 ID 一覧を印字するプログラムを作成する。

(1)　最新の利用者 ID 一覧をファイル `NewFile` から，一定期間前の世代の利用者 ID 一覧をファイル `OldFile` から，それぞれ読み込む。

(2)　次の①，②の少なくとも一方に該当する利用者 ID について，利用者 ID，利用者名の後に，①に該当する場合は“現在使用不可”を，②に該当する場合は“期間中未使用”を印字する。

①　`NewFile` 中にあって，属性のビット `r` が 1 である利用者 ID

ここで，属性のビットrは，利用者IDが使用可能なら0，使用不可なら1である。

② NewFile 及び OldFile の両方にあって，最終使用日の値が等しい利用者ID（全桁が“0”同士で等しい場合を含む）

(3) 図3に，図1のデータ例を用いた実行結果を示す。

```
利用者ID  利用者名    使用状況
AE001     UserE1     現在使用不可  期間中未使用
AE003     UserE3     期間中未使用
```

注記　見出し行は，事前に印字されているものとする。

図3　図1のデータ例を用いた使用状況の印字結果

この処理を実装するためには，プログラム中の while 文のブロック内（αで示した部分）を次のように変更すればよい。ここで，プログラム中の［　a　］には，正しい答えが入っているものとする。

```
if (strcmp(NewID, OldID) <= 0) {
   if ((［　　e　　］) || (［　　f　　］)) {
      printf("\n%-8s  %-10s", NewID, NewName);
      if (［　　e　　］)
         printf("  現在使用不可");
      if (［　　f　　］)
         printf("  期間中未使用");
   }
   ReadNewRecord();
}
else
   ReadOldRecord();
```

e，fに関する解答群

ア　(NewAttr & BitR) != (OldAttr & BitR)

イ　(NewAttr & BitR) == BitR

ウ　strcmp(NewDate, OldDate) == 0

エ　strcmp(NewID, OldID) == 0

オ　strcmp(NewID, OldID) == 0 && (NewAttr & BitR) == BitR

カ　strcmp(NewID, OldID) == 0 && strcmp(NewDate, OldDate) == 0

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〕

X 社では，特定の商品の売上傾向を分析するために，2 年分の売上についてグラフ化することにした。このプログラムは，分析する商品の商品コードをパラメタで受け取り，全ての商品の売上データが記録されている売上ファイルから当該商品の売上を月ごとに集計して，結果を図 1 に示すように印字する。

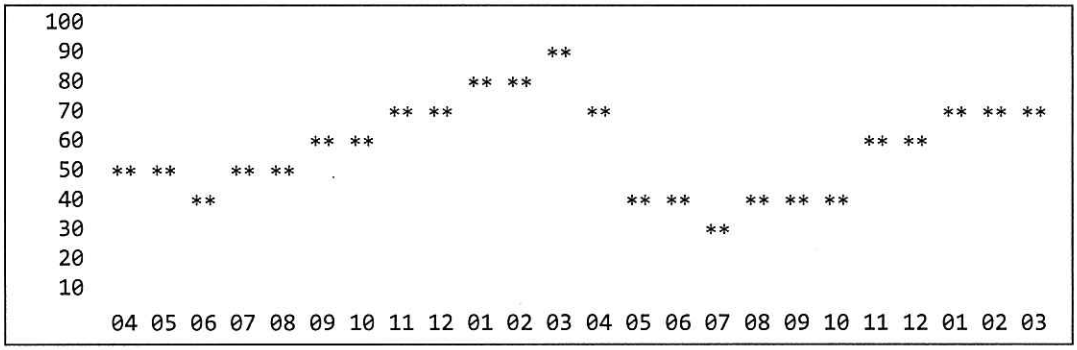


図1　売上グラフの例

(1)　売上グラフは，2012 年 4 月～2014 年 3 月の各月の売上を表す。

①　縦軸の数値は売上金額（単位は万円）で，下段から，1～100,000 円，100,001～200,000 円，…，900,001～1,000,000 円を表す。

②　横軸の数値は月で，左端から，2012 年 4 月，2012 年 5 月，…，2014 年 3 月を表す。

③　縦軸と横軸の数値は，用紙にあらかじめ印字されている。

(2)　売上ファイルは，図 2 に示すレコード様式の順ファイルで，X 社が扱う全ての商品の 2012 年 4 月～2014 年 3 月の各営業日の売上データが格納されている。レコードは売上日の昇順に整列されている。

売上日 8桁	商品コード 5桁	個数 4桁	売上金額 8桁	取引先コード 4桁

図2　売上ファイルのレコード様式

① 売上日には，年，月，日が，それぞれ 4 桁，2 桁，2 桁の西暦で格納されている。

② 商品コードには，販売した商品のコードが格納されている。商品コードは，商品ごとに一意に割り当てられている。

③ 取引先コードには，商品を販売した取引先のコードが格納されている。取引先コードは，取引先ごとに一意に割り当てられている。

④ 各商品の月ごとの売上金額は，1～999,999 円とする。

〔プログラム〕

（行番号）

```
 1   DATA DIVISION.
 2   FILE SECTION.
 3   FD S-FILE.
 4   01 S-REC.
 5      02 S-DATE.
 6         03 S-YYYY      PIC 9(4).
 7         03 S-MM        PIC 9(2).
 8         03 S-DD        PIC 9(2).
 9      02 S-GOODS     PIC 9(5).
10      02 S-QUANTITY  PIC 9(4).
11      02 S-AMOUNT    PIC 9(8).
12      02 S-CUSTOMER  PIC 9(4).
13   FD P-FILE.
14   01 P-REC          PIC X(80).
15   WORKING-STORAGE SECTION.
16   77 EOF-FLAG     PIC X(1).
17      88 S-INIT    VALUE SPACE.
18      88 S-EOF     VALUE "E".
19   77 IX-LINE      PIC 9(2).
20   77 IX-COL       PIC 9(2).
21   77 CR-MONTH     PIC 9(2).
22   01 AMOUNT-DATA.
23      02 AMOUNT-MONTH OCCURS 24 PIC 9(6).
24   01 PRINT-TABLE.
25      02 PRINT-LINE  OCCURS 10.
26         03              PIC X(6).
27         03 PRINT-ELM   OCCURS 24.
28            04 PRINT-MK   PIC X(2).
29            04           PIC X(1).
30   LINKAGE SECTION.
31   77 PRM-GOODS    PIC 9(5).
32   PROCEDURE DIVISION USING PRM-GOODS.
33   MAIN-PROC.
34       SET S-INIT TO TRUE.
35       INITIALIZE AMOUNT-DATA.
36       MOVE 4 TO CR-MONTH.
```

```
37       MOVE 1 TO IX-COL.
38       OPEN INPUT  S-FILE
39            OUTPUT P-FILE.
40       PERFORM UNTIL S-EOF
41          READ S-FILE
42             AT END     SET S-EOF TO TRUE
43             NOT AT END IF S-GOODS = PRM-GOODS THEN
44                           PERFORM ADD-PROC
45                        END-IF
46          END-READ
47       END-PERFORM.
48       PERFORM PRINT-PROC.
49       CLOSE S-FILE P-FILE.
50       EXIT PROGRAM.
51   ADD-PROC.
52       IF CR-MONTH NOT = S-MM THEN
53          [          a          ]
54          ADD 1 TO IX-COL
55       END-IF.
56       [          b          ].
57   PRINT-PROC.
58       MOVE SPACE TO PRINT-TABLE.
59       PERFORM VARYING IX-COL FROM 1 BY 1 UNTIL IX-COL > 24
60          COMPUTE IX-LINE = (AMOUNT-MONTH(IX-COL) + 99999) / 100000
61          MOVE ALL "*" TO [          c          ]
62       END-PERFORM.
63       PERFORM VARYING IX-LINE FROM 10 BY -1 UNTIL IX-LINE = 0
64          WRITE P-REC FROM PRINT-LINE(IX-LINE)
65       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，bに関する解答群

ア　ADD 1 TO CR-MONTH

イ　ADD S-AMOUNT TO AMOUNT-MONTH(IX-COL)

ウ　MOVE 1 TO CR-MONTH

エ　MOVE S-AMOUNT TO AMOUNT-MONTH(CR-MONTH)

オ　MOVE S-MM TO CR-MONTH

カ　MOVE S-MM TO IX-LINE IX-COL

cに関する解答群

ア　PRINT-ELM(IX-COL, IX-LINE)　　イ　PRINT-ELM(IX-LINE, IX-COL)

ウ　PRINT-MK(IX-COL, IX-LINE)　　エ　PRINT-MK(IX-LINE, IX-COL)

設問2　図1に示すグラフでは，季節変動がある商品の中長期的な傾向の分析は難しいことが分かった。そこで，図3に示すZチャートを表示するようにプログラムを変更することにした。表1中の［　　］に入れる正しい答えを，解答群の中から選べ。ここで，表1中の［　c　］には設問1の正しい答えが入っているものとする。

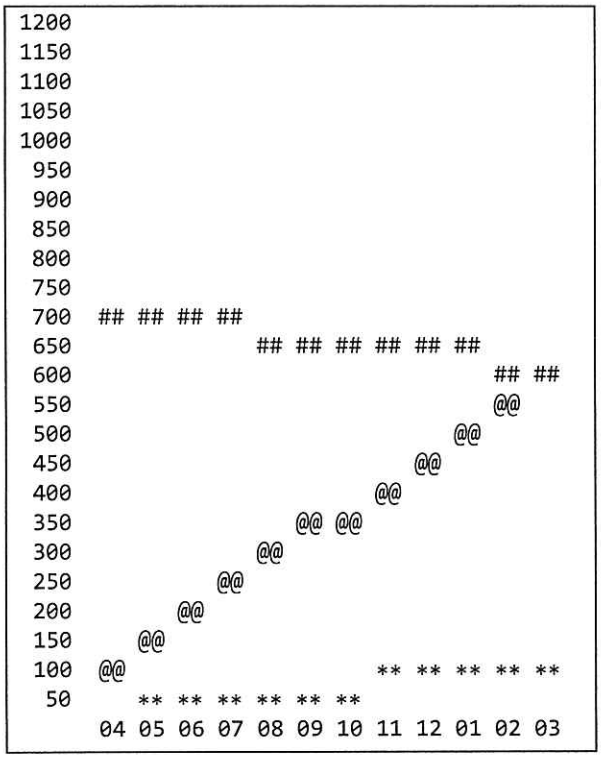


図3　Zチャートの例

〔Zチャートの説明〕

ある月を起点に，各月の売上，起点からの売上累計，その月を含む過去1年間の売上合計を表すグラフである。これら三つのグラフを重ね合わせるとZの形になることからZチャートと呼ばれ，季節変動のある商品の売上傾向を分析する際などに用いられる。

〔図3の説明〕

(1) 縦軸の数値は売上金額（単位は万円）で，下段から，1～500,000 円，500,001～1,000,000 円，…，11,500,001～12,000,000 円を表す。

(2) 横軸の数値は月で，左端から，2013 年 4 月，2013 年 5 月，…，2014 年 3 月を表す。

(3) 2013 年 4 月を起点として，各月の売上，起点からの売上累計，その月を含む過去 1 年間の売上合計を，それぞれ記号 "*"，"@"，"#" で印字する。ここで，印字が重なる場合は，優先順位（"#" ＞ "@" ＞"*"）の高い方を印字する。

(4) 縦軸と横軸の数値は，用紙にあらかじめ印字されている。

表1 プログラムの変更内容

処置	変更内容
行番号21と22の間に追加	77 IX-AMOUNT PIC 9(2). 77 Z-TOTAL PIC 9(8). 77 Z-YEAR PIC 9(8).
行番号24から29を変更	01 PRINT-TABLE. 02 PRINT-LINE OCCURS 24. 03 PIC X(6). 03 PRINT-ELM OCCURS 12. 04 PRINT-MK PIC X(2). 04 PIC X(1).
行番号59から65を変更	MOVE ZERO TO Z-TOTAL Z-YEAR. PERFORM VARYING IX-AMOUNT FROM 1 BY 1 UNTIL IX-AMOUNT > 12 [d] END-PERFORM. PERFORM VARYING IX-COL FROM 1 BY 1 UNTIL IX-COL > 12 COMPUTE IX-LINE = (AMOUNT-MONTH(IX-AMOUNT) + 499999) / 500000 MOVE ALL "*" TO [c] [e] COMPUTE IX-LINE = (Z-TOTAL + 499999) / 500000 MOVE ALL "@" TO [c] [f] COMPUTE IX-LINE = (Z-YEAR + 499999) / 500000 MOVE ALL "#" TO [c] ADD 1 TO IX-AMOUNT END-PERFORM. PERFORM VARYING IX-LINE FROM 24 BY -1 UNTIL IX-LINE = 0 WRITE P-REC FROM PRINT-LINE(IX-LINE) END-PERFORM.

d～fに関する解答群

ア　ADD AMOUNT-MONTH(IX-AMOUNT) TO Z-TOTAL

イ　ADD AMOUNT-MONTH(IX-AMOUNT) TO Z-YEAR

ウ　ADD Z-TOTAL TO Z-YEAR

エ
```
COMPUTE Z-YEAR = Z-YEAR
        + AMOUNT-MONTH(IX-AMOUNT) - AMOUNT-MONTH(IX-AMOUNT - 12)
```

オ　COMPUTE Z-YEAR = Z-YEAR + Z-TOTAL + AMOUNT-MONTH(IX-COL - 1)

カ　MOVE AMOUNT-MONTH(IX-COL) TO Z-TOTAL

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。
（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〕

D 君は，社内向けの応用プログラム（以下，アプリケーションという）で共通に使うライブラリを開発するチームに属している。ライブラリの次のバージョンで，時間に関するクラスを幾つか用意することになり，D 君は期間（時間間隔）を表すクラス Period の開発を担当することになった。このクラスは，“有効期間は，今週月曜日の午前 0 時から金曜日の終わりまで（土曜日の午前 0 時以降は，無効）”など，ある日付及び時刻（以下，日時という）から別の日時までの時間間隔を表すことを想定している。期間の基準になる日時を始点，終わりを表す日時を終点という。また，ある始点から過去に遡って期間を表すことも想定している。D 君が定めたクラス Period の外部仕様は，次のとおりである。

クラス Period は，始点から終点までの期間を表す。期間には，始点は含まれるが終点は含まれない。始点と終点が同じ日時の場合は，期間が空であると定義し，いかなる日時も含まないものとする。

(1) コンストラクタは，引数で与えられた始点（start）と終点（end）から期間を表すインスタンスを生成する。日時は，クラス Date のインスタンスで与えられ，協定世界時 1970 年 1 月 1 日午前 0 時（以下，この日時をエポックという）以降でなければならない。引数 start 又は end が null の場合は，NullPointerException を，日時がエポックよりも前の場合は，IllegalArgumentException を投げる。

(2) メソッド getStart：始点を返す。

(3) メソッド getEnd：終点を返す。

(4) メソッド getLength：期間の長さ（ミリ秒）を返す。終点が始点よりも前の場合は，負の値で返す。期間が空の場合は，0 を返す。

(5) メソッド isBackward：終点が始点よりも前の場合は，true を返す。それ以外は，false を返す。期間が空の場合は，false を返す。

(6) メソッド contains：引数で与えられた日時が期間に含まれる場合は，true を返す。それ以外は，false を返す。期間が空の場合は，false を返す。引数が

null の場合は，NullPointerException を投げる。

この仕様でチームの承認を得て，次のプログラム 1 を作成した。

〔プログラム 1〕

```
import java.util.Date;

public final class Period {
   private static final Date EPOCH = new Date(0L);
   private final Date start, end;

   public Period(Date start, Date end) {
      if (start == null || end == null) {
         throw new NullPointerException();
      }
      if (start.compareTo(EPOCH) < 0
          || end.compareTo(EPOCH) < 0) {
         throw new IllegalArgumentException();
      }
      this.start = start;
      this.end = end;
   }

   public Date getStart() { return start; }

   public Date getEnd() { return end; }

   public long getLength() {
      return end.getTime() - start.getTime();
   }

   public boolean isBackward() {
      return end.compareTo(start) < 0;
   }

   public boolean contains(Date time) {
      return (time.compareTo(start) >= 0
              && time.compareTo(end) < 0)
          || (time.compareTo(start) [ a ] 0
              && time.compareTo(end) [ b ] 0);
   }
}
```

次に，D 君は，テスト用にプログラム 2 を作成した。問題が検出されなければ，プログラム 2 は何も出力せずに終了する。問題を検出した場合は，RuntimeException を投げる。

〔プログラム2〕

```
import java.util.Date;

public class PeriodTest {
   private static final long DELTA = 60 * 60 * 1000L;

   // コンストラクタが，正しくない引数に対して仕様どおりに例外を投げること
   // を確認する。引数 expected は，期待される例外の型をクラスで指定する。
   private static void testException(Date start, Date end,
               Class<? extends RuntimeException> expected) {
      try {
         Period period = new Period(start, end);
      } catch (RuntimeException e) {
         if (e.getClass() == expected) {
            return;
         }
         throw new RuntimeException("unexpected exception", e);
      }
      throw new RuntimeException("no " + expected + " thrown");
   }

   // 期間の長さ及び方向（始点と終点の前後関係）の整合性を確認する。
   private static void testConsistency(Period period,
                                       long length) {
      if (period.getLength() != length) {
         throw new RuntimeException("invalid getLength() value");
      }
      if (period.isBackward() != (length [ c ] 0)) {
         throw new RuntimeException("isBackward failed");
      }
   }

   // メソッド contains が，期間に含まれるデータ及び含まれないデータ
   // に対して正しく判定することを確認する。また，メソッドの引数が
   // null のとき，NullPointerException を投げることを確認する。
   private static void testContains(Period period, long[] valid,
                                    long[] invalid) {
      for (long time : valid) {
         if (!period.contains(new Date(time))) {
            throw new RuntimeException("failed with valid: " + time);
         }
      }
      for (long time : invalid) {
         if (period.contains(new Date(time))) {
            throw new RuntimeException("failed with invalid: "
                                       + time);
         }
      }
      try {
         period.contains(null);
```

```
            throw new RuntimeException("no NPE thrown");
        } catch (NullPointerException e) {
        }
    }

    public static void main(String[] args) {
        testException(null, new Date(0L), [ d ]);
        testException(new Date(0L), null, [ d ]);
        testException(new Date(-1L), new Date(0L), [ e ]);
        testException(new Date(0L), new Date(-1L), [ e ]);

        long now = System.currentTimeMillis();
        Date start = new Date(now);
        Date end = new Date(now + DELTA);
        final Period period = new Period(start, end);
        testConsistency(period, DELTA);
        testContains(period,
            new long[] { now, now + 1, now + DELTA - 1 },
            new long[] { now - 1, now + DELTA, now + DELTA + 1 });

        Date backwardEnd = new Date(now - DELTA);
        final Period backwardPeriod = new Period(start, backwardEnd);
        testConsistency(backwardPeriod, -DELTA);
        testContains(backwardPeriod,
            new long[] { now, now - 1, now - DELTA + 1 },
            new long[] { now + 1, now - DELTA, now - DELTA - 1 });

        final Period nullPeriod = new Period(start, start);
        testConsistency(nullPeriod, 0);
        testContains(nullPeriod, new long[0],
                     new long[] { now - 1, now, now + 1 });
                                                    ←―― α
    }
}
```

設問1　プログラム 1 及び 2 中の［　　］に入れる正しい答えを，解答群の中から選べ。ただし，プログラム2を実行したとき，例外は発生せず正常に終了するものとする。

a～cに関する解答群

ア　!=	イ　<	ウ　<=
エ　==	オ　>	カ　>=

d，eに関する解答群

ア　IllegalArgumentException

イ　IllegalArgumentException.class

ウ　IllegalArgumentException.getClass()

エ　new IllegalArgumentException()

オ　new NullPointerException()

カ　NullPointerException

キ　NullPointerException.class

ク　NullPointerException.getClass()

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。

チーム内でのコードレビューも終わり，クラス Period を含むライブラリが評価版として社内にリリースされた。それからしばらくして，アプリケーション開発チームから，アプリケーションの実行中に Period のインスタンスが表す期間が変わってしまうという指摘を受けた。D君は，プログラムを見直したがクラスもフィールドも最終（final）なので，生成後にインスタンスの状態が変化することはないと考えた。そこで，D君は，アプリケーション開発チームと一緒にデバッグを行った。

その結果，アプリケーションでは，Period のメソッド getStart で始点を取得し，その Date インスタンスの表す日時を変更して別の Period のインスタンスを生成するときの引数に使用していることが分かった。これが原因で，先に生成済みの Period のインスタンスの始点が変更されていた。すなわち，Period インスタンス自体は不変でも，それから参照される Date インスタンスは可変なので，表している期間が変わってしまうことがある。したがって，Period のインスタンスで始点及び終点を保持する場合は，外部からの変更ができないようにする必要がある。また，クラス Date は，最終（final）ではないので，どのような下位クラスでも作成可能であり，上書きしたメソッドについて挙動の変更が可能である。これらは，データの改ざんを許すことになるので，セキュリティ上の問題でもある。これらの問題を修正するために，D君は，クラス Period に次

の変更を加えた。

(1) コンストラクタの処理において，this.start の代入文の右辺を new Date(start.getTime())に変更した。this.end の代入文についても同様の変更をした。この変更によって，フィールド start 及び end は，それぞれ引数と同じ値をもつ別の java.util.Date のインスタンスを参照する。

(2) メソッド getStart の返却値を［ f ］に変更した。メソッド getEnd についても同様の変更をした。

D君は，プログラム2のメソッド main の α で示した部分に次のプログラムを追加し，正しく修正されたことを確認した。

```
start.setTime(now - DELTA);
end.setTime(now + DELTA * 2);
testConsistency(period, [ g ]);
testContains(period,
   new long[] { now, now + 1, now + DELTA - 1 },
   new long[] { now - 1, now + DELTA, now + DELTA + 1 });
Date newStart = period.getStart();
newStart.setTime(newStart.getTime() - DELTA);
Date newEnd = period.getEnd();
newEnd.setTime(newEnd.getTime() + DELTA);
testConsistency(period, [ g ]);
testContains(period,
   new long[] { now, now + 1, now + DELTA - 1 },
   new long[] { now - 1, now + DELTA, now + DELTA + 1 });
```

fに関する解答群

ア (Date) start.clone()　　イ null

ウ start.clone()　　エ start.getTime()

オ start.setTime(start.getTime())

gに関する解答群

ア 0　　イ DELTA　　ウ DELTA * 2

エ DELTA * 3　　オ DELTA / 2　　カ DELTA / 3

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〕

n個の要素（大きさ1語）から成る配列に格納されているデータを，バブルソートによって昇順に並べ替える副プログラムSORTである。

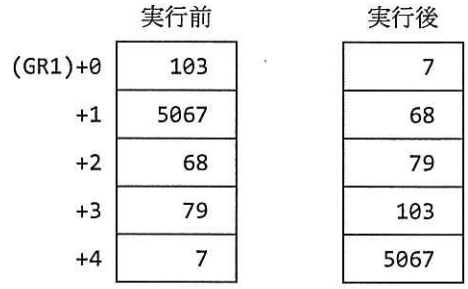


図1　配列内のデータの並べ替え（n＝5の例）

(1)　配列の先頭アドレスはGR1に設定されて，主プログラムから渡される。

(2)　配列の要素数n（n≧2）はGR2に設定されて，主プログラムから渡される。

(3)　配列に格納されているデータは，符号なしの整数とみなして並べ替える。

(4)　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1〕

(行番号)

```
 1  SORT    START
 2          RPUSH
 3          SUBA    GR2,=1          ; ループのカウンタ1を設定
 4  LOOP1   LD      GR3,GR2         ; ループのカウンタ2を設定
 5          LD      GR4,GR1         ; GR4←比較する要素のアドレス
 6  LOOP2   LD      GR5,0,GR4
 7          CPL     GR5,1,GR4       ; 二つの要素を比較
 8          [      a      ]
 9          LD      GR6,1,GR4       ; 二つの要素を入れ替え
10          ST      GR5,1,GR4
11          ST      GR6,0,GR4
12  CONT    ADDA    GR4,=1
13          SUBA    GR3,=1
14          JPL     LOOP2
```

```
15          SUBA    GR2,=1
16          [   b   ]
17          RPOP
18          RET
19          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。

解答群

ア	JMI	CONT	イ	JMI	LOOP1	ウ	JPL	CONT
エ	JPL	LOOP1	オ	JZE	CONT	カ	JZE	LOOP1

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配列の要素数nが4で，配列の内容が図2のとき，行番号7のCPL命令は［　c　］回実行され，行番号11のST命令は［　d　］回実行される。

(GR1)+0	2
+1	4
+2	1
+3	3

図2　主プログラムから渡された配列の内容

解答群

ア　1	イ　2	ウ　3
エ　4	オ　5	カ　6

設問3　m個の数字列を入力し，副プログラムSORTを使って，昇順に並べ替えて出力する主プログラムMAINを作成した。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。

(1) 入力する数字列は4桁以内とする。

(2) 2≦m≦100 とする。

(3) 出力する数字列は4桁で，左のゼロは空白で表示する。

入力数字列	出力数字列
'298'	' 9'
'30'	' 30'
'9'	'298'
'3240'	'508'
'508'	'3240'

図3　入力数字列と出力数字列（m＝5の例）

(4) 主プログラム MAIN は入力した数字列を，1語長の2進化10進数に変換して配列に格納し，並べ替えを行う。図4に，数字列'298'を変換した2進化10進数の例を示す。

0000	0010	1001	1000

図4　2進化10進数の例

〔プログラム2〕

```
MAIN    START
        LD      GR2,=0          ; 数字列のカウンタを初期化
LOOP3   IN      BUF,ILEN
        LD      GR0,ILEN
        JMI     EOF
        LD      GR3,=0          ; 配列に格納する内容を初期化
        LAD     GR4,BUF
LOOP4   LD      GR5,0,GR4       ; GR5←数字
        AND     GR5,=#000F      ; 数字の数値化
        SLL     GR3,4           ; 処理済の内容を桁上げ
        OR      GR3,GR5         ; 空いた右端4ビットに数値を格納
        ADDA    GR4,=1
        SUBA    GR0,=1
        JNZ     LOOP4
        ST      GR3,ARRAY,GR2   ; 2進化10進数にした数字列を格納
        ADDA    GR2,=1
        JUMP    LOOP3
EOF     LAD     GR1,ARRAY
```

```
        CALL    SORT
        LD      GR4,=' '        ; 空白文字を設定
LOOP5   LD      GR0,0,GR1
        LD      GR5,=3          ; ループのカウンタを設定
LOOP6   LD      GR3,GR0
        [    e    ]
        OR      GR3,=#0030      ; 数値の数字化
        ST      GR3,BUF,GR5     ; バッファに数字を設定
        SUBA    GR5,=1
        JMI     WRITE
        [    f    ]
        JNZ     LOOP6
LOOP7   ST      GR4,BUF,GR5     ; バッファに空白を設定
        SUBA    GR5,=1
        JMI     WRITE
        JUMP    LOOP7
WRITE   OUT     BUF,OLEN
        LAD     GR1,1,GR1
        SUBA    GR2,=1
        JNZ     LOOP5
        RET
BUF     DS      256
ILEN    DS      1
OLEN    DC      4
ARRAY   DS      100
        END
```

解答群

ア　AND　GR0,=#000F　　イ　AND　GR3,=#000F　　ウ　SLL　GR0,4

エ　SLL　GR3,4　　オ　SRL　GR0,4　　カ　SRL　GR3,4

〔 メ モ 用 紙 〕

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〕

鉄道会社 Y 社では，表計算ソフトを使って乗車駅と降車駅を入力すると鉄道路線における駅間の距離（以下，乗車距離という）と運賃を表示するプログラムを作成した。

(1) 鉄道路線は，隣接駅間の距離（以下，隣接駅間距離という）が 5 km を超えず，分岐をもたない全長 94.2 km の単一路線であり，起点駅から終点駅まで 32 の駅から成る。

(2) 各駅には“001”～“032”の駅コードが割り付けられている。

(3) ワークシート“鉄道運賃”に乗車駅の駅コード（セル B1）と降車駅の駅コード（セル B2）を入力すると，乗車距離（セル B3）と運賃（セル B4）が表示される。乗車駅と降車駅の駅コードは，鉄道路線の各駅に割り付けられた駅コードのリストから選択して入力するようになっている。ワークシート“鉄道運賃”の例を図 1 に示す。

	A	B
1	乗車駅（駅コード）	004
2	降車駅（駅コード）	001
3	乗車距離（km）	11.6
4	運賃（円）	220

図 1　ワークシート“鉄道運賃”の例

(4) 駅コードと隣接駅間距離は，起点駅から終点駅に向かってワークシート“路線情報”の列 A と列 B に順に格納されている。ワークシート“路線情報”の例を図 2 に示す。

	A	B	C
1	駅コード	隣接駅間距離（km）	
2	001	4.3	○
3	002	3.0	○
4	003	4.3	○
5	004	4.7	
⋮	⋮	⋮	⋮
31	030	2.4	
32	031	1.3	
33	032	0.0	

図 2　ワークシート“路線情報”の例

① ワークシートの各行の隣接駅間距離には，その行の駅コードに対応する駅から次行の駅コードに対応する駅までの距離（単位は km）が格納されている。例えば，駅コード“002”と駅コード“003”の駅間の距離は，セル B3 に格納されている 3.0 である。終点駅の駅コードに対応する行の隣接駅間距離（セル B33）には 0.0 が格納されている。

② 列 C は乗車距離計算のための作業領域に用いる。

(5) 運賃は，乗車距離に関係なく一律に定められている基本運賃 100 円と，乗車距離に応じて決まる距離運賃とを合計した額である。

(6) 距離運賃は，決められた四つの距離の範囲で乗車距離を区分し，決められた計算方法で区分ごとの運賃（以下，区分運賃という）を計算し，それらを合計した額である。

① 各区分には 1～4 の番号（以下，区分番号という）が割り付けられている。各区分の距離の範囲は表 1 のとおりである。

表 1　区分番号と距離の範囲

区分番号	距離の範囲
1	10 km 以下の範囲
2	10 km を超え 40 km 以下の範囲
3	40 km を超え 70 km 以下の範囲
4	70 km を超える範囲

② 乗車距離を，決められた四つの距離の範囲で区分し，区分ごとの距離（以下，区分距離という）に分ける。例えば，乗車距離が 11.6 km の場合，区分番号 1 の区分距離 10 km と，区分番号 2 の区分距離 1.6 km（11.6 km − 10 km）に分ける。

③ 区分ごとに，運賃計算の基準となる単位距離が決められている。区分，単位距離と単位運賃の関係を表 2 に示す。区分距離の端数は単位距離に切り上げ，単位距離ごとに単位運賃を加算する。

表２　単位距離と単位運賃

区分番号	単位距離（km）	単位運賃（円）
1	1	10
2	3	20
3	5	30
4	10	50

(7)　例えば，乗車距離が 11.6 km である駅コード“004”から駅コード“001”までの運賃は，次のように求める。ここで，計算式中の“切上げ(X, 0)”は，Xの小数点以下を切り上げた値を表す。

①　区分番号１の区分運賃は，1km までごとに 10 円を加算し，次の式で求める。

10 円 × 切上げ(10 km ÷ 1 km, 0) ＝ 100 円

②　区分番号２の区分運賃は，3km までごとに 20 円を加算し，次の式で求める。

20 円 × 切上げ((11.6 km − 10 km) ÷ 3 km, 0) ＝ 20 円

③　運賃は，基本運賃 100 円と距離運賃 120 円（①，②で求めた区分運賃の合計額）を合計した 220 円になる。

(8)　基本運賃と各区分の区分番号，距離の範囲の上限（以下，上限距離という），単位距離及び単位運賃を格納したワークシート“運賃情報”の例を図３に示す。

	A	B	C	D	E	F
1			基本運賃（円）	100		
2						
3	区分ごとの運賃体系表					
4	区分番号	上限距離 (km)	単位距離 (km)	単位運賃 (円)	区分距離 (km)	区分運賃 (円)
5	1	10	1	10	10.0	100
6	2	40	3	20	1.6	20
7	3	70	5	30	0.0	0
8	4	999999	10	50	0.0	0

図３　ワークシート“運賃情報”の例

① セルD1には，基本運賃が格納されている。

② 区分ごとの運賃体系表（セル A4～D8）のセル A5～D8 には，各区分の区分番号，上限距離，単位距離及び単位運賃が区分番号の昇順に格納されている。

③ 区分番号4の上限距離には，999999が格納されている。

④ 列E，Fには，各区分の区分距離と区分運賃が表示される。

〔乗車距離の計算〕

乗車駅から降車駅までの間（以下，乗車区間という）にある隣接駅間距離を選び出し，それらを合計したものを乗車距離とする。乗車距離（ワークシート“鉄道運賃”のセル B3）を求めるために，ワークシート“路線情報”と“鉄道運賃”に(1)～(3)の手順で式の入力を行う。

(1) 乗車区間の判別に当たり，ワークシート“路線情報”のセル C34 とセル C35 を使用する。乗車駅及び降車駅の駅コードを利用するために，ワークシート“路線情報”のセル C34 に次の式を入力し，セル C35 に複写する。

照合一致(鉄道運賃!B1, A$2～A$33, 0)

(2) ワークシート“路線情報”のセル C2 に次の式を入力し，セル C3～C32 に複写する。

IF([a] ([b] , [c]), '○', null)

(3) 乗車距離を求めるために，ワークシート“鉄道運賃”のセル B3 に次の式を入力する。

条件付合計(路線情報!C2～C32, ＝'○', 路線情報!B2～B32)

〔運賃の計算〕

区分距離と区分運賃を計算し，基本運賃と全ての区分運賃を合計したものを運賃とする。運賃（ワークシート“鉄道運賃”のセル B4）を求めるために，ワークシート“運賃情報”と“鉄道運賃”に(1)～(4)の手順で式の入力を行う。

(1) 区分番号1の区分距離を求めるために，ワークシート“運賃情報”のセル E5 に次の式を入力する。

IF(鉄道運賃!B3≦B5, 鉄道運賃!B3, B5)

(2) 区分番号 2～4 の区分距離を求めるために，ワークシート“運賃情報”のセル

E6 に次の式を入力し，セル E7，E8 に複写する。

IF(鉄道運賃!B$3≦B6, 鉄道運賃!B$3, B6)－［ d ］

(3) 区分運賃を求めるために，ワークシート“運賃情報”のセル F5 に次の式を入力し，セル F6～F8 に複写する。

［ e ］

(4) 乗車区間の運賃を求めるために，ワークシート“鉄道運賃”のセル B4 に次の式を入力する。

運賃情報!D1＋合計(運賃情報!F5～F8)

設問 1 乗車距離の計算，運賃の計算に関する説明文中の［　　］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 論理積　　イ 論理和

b に関する解答群

ア 照合一致(A2, A$2～A$33, 0)＜最小(C$34～C$35)

イ 照合一致(A2, A$2～A$33, 0)≦最小(C$34～C$35)

ウ 照合一致(A2, A$2～A$33, 0)＞最小(C$34～C$35)

エ 照合一致(A2, A$2～A$33, 0)≧最小(C$34～C$35)

c に関する解答群

ア 照合一致(A2, A$2～A$33, 0)＜最大(C$34～C$35)

イ 照合一致(A2, A$2～A$33, 0)≦最大(C$34～C$35)

ウ 照合一致(A2, A$2～A$33, 0)＞最大(C$34～C$35)

エ 照合一致(A2, A$2～A$33, 0)≧最大(C$34～C$35)

d に関する解答群

ア B5　　イ E5　　ウ 合計(B$5～B5)

エ 合計(B$5～B$8)　　オ 合計(E5～E$8)　　カ 合計(E$5～E5)

eに関する解答群

ア　切上げ(B5, 0)／C5 * D5　　イ　切上げ(B5／C5, 0) * D5

ウ　切上げ(E5, 0)／C5 * D5　　エ　切上げ(E5／C5, 0) * D5

オ　切捨て(B5／C5 + 1, 0) * D5　　カ　切捨て(E5／C5 + 1, 0) * D5

設問2　Y社では，区分運賃を見直すことによる運賃改定を行うことにした。ただし，区分ごとの単位距離と単位運賃は変更せず，距離の範囲だけを見直す。これに伴い，ワークシート“運賃情報”の区分番号1～3の上限距離（セルB5～B7）の値を変更するためのマクロを作成し，ワークシート“運賃情報”に格納した。マクロ Change_SectionInfo に関する説明を読んで，マクロ中の［　　］に入れる正しい答えを，解答群の中から選べ。ここで，g1 と g2 に入れる答えは，g に関する解答群の中から組合せとして正しいものを選ぶものとする。

〔距離の範囲見直しの説明〕

(1)　少なくとも隣接駅間距離は，区分番号1の距離の範囲に入るようにする。

①　隣接駅間距離の中で最長の距離を求める。

②　求めた距離の小数点以下を切り上げた値を，区分番号1の新たな上限距離とする。

(2)　五つ離れた駅までの距離は，区分番号2の距離の範囲に入るようにする。

①　鉄道路線上のそれぞれの駅から五つ離れた駅までの距離を計算し，その中で最長の距離を求める。

②　区分番号1の新たな上限距離に区分番号2の単位距離の倍数を加えた値で，かつ，①で求めた距離以上となる最小値を，区分番号2の新たな上限距離とする。

(3)　鉄道路線の全区間（起点駅から終点駅まで）の距離の 50％以下の距離は，区分番号3の距離の範囲に入るようにする。

①　鉄道路線の全区間の50％の距離を求める。

②　区分番号2の新たな上限距離に区分番号3の単位距離の倍数を加えた値で，かつ，①で求めた距離以上となる最小値を，区分番号3の新たな上限距離とする。

表計算

〔マクロ：Change_SectionInfo〕

```
○マクロ：Change_SectionInfo
○数値型：I, J
○数値型：Dist1, Dist2, Dist3, Sum2
• Dist1 ← 最大(路線情報!B2～B32)
• Dist2 ← 0
■ I：0, I ≦ 26, 1
  • Sum2 ← 相対(路線情報!B2, I, 0)
  ■ [  f  ], 1
    • Sum2 ← Sum2 ＋ 相対(路線情報!B2, I ＋ J, 0)
  ■
  ▲ Sum2 ＞ Dist2
    • Dist2 ← Sum2
  ▼
■
• Dist3 ← 合計(路線情報!B2～B32) ＊ 0.5
• B5 ← 切上げ(Dist1 ／ C5, 0) ＊ C5
• B6 ← 切上げ([ g1 ] ／ C6, 0) ＊ C6 ＋ B5
• B7 ← 切上げ([ g2 ] ／ C7, 0) ＊ C7 ＋ B6
```

fに関する解答群

ア　J：0, J ≦ 4　　イ　J：1, J ≦ 4　　ウ　J：0, J ≦ 5

エ　J：1, J ≦ 5　　オ　J：0, J ≦ 31　　カ　J：1, J ≦ 31

gに関する解答群

	g1	g2
ア	Dist2	Dist3
イ	(Dist2 － B5)	(Dist3 － B6)
ウ	(Dist2 － Dist1)	(Dist3 － Dist2)
エ	(Dist2 － B5 ＋ 1)	(Dist3 － B6 ＋ 1)
オ	(Dist2 － Dist1 ＋ 1)	(Dist3 － Dist2 ＋ 1)

■ Java プログラムで使用する API の説明

java.util

public class Date

クラス Date は，特定の日時をミリ秒単位で表す。日時は，協定世界時 1970 年 1 月 1 日午前 0 時（エポック）からの相対時間で表す。

コンストラクタ

public Date(long date)

引数で与えられた日時を表すインスタンスを生成する。

引数：date — 日時（エポックからのミリ秒単位の相対時間）

メソッド

public int compareTo(Date anotherDate)

順序付けのために 2 個の Date インスタンスを比較する。

引数： anotherDate — 比較対象の Date インスタンス

戻り値：この Date が引数の Date より前の日時の場合は，負の値

この Date と引数の Date が同じ日時の場合は，0

この Date が引数の Date より後の日時の場合は，正の値

public long getTime()

この Date インスタンスが表す日時をエポックからの相対時間（ミリ秒）で返す。

戻り値：この Date インスタンスが表す日時（エポックからのミリ秒単位の相対時間）

public void setTime(long time)

この Date インスタンスを引数が表す日時に設定する。

引数：time — 日時（エポックからのミリ秒単位の相対時間）

public Object clone()

この Date インスタンスの複製を返す。

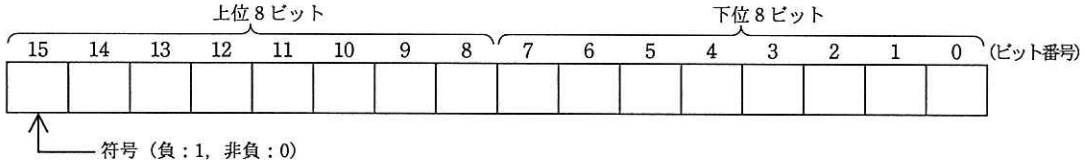
戻り値：この Date インスタンスの複製

■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
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。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”のセル範囲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，セル範囲 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，セル範囲 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，セル範囲 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(－314.159,2) は，－314.16 を返す。 ［例2］切上げ(314.159，－2) は，400 を返す。 ［例3］切上げ(314.159,0) 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，セル範囲 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，セル範囲 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，セル範囲 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 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 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 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 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”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(B5, row, 0)  ←  順位(相対(C5, row, 0), G5～G9, 0)
■
```

例は，セル C5，C6，…，C9 の各値に対して，セル範囲 G5 ～ G9 の中での順位を調べ，その順位をセル B5，B6，…，B9 に順に代入する。

〔 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（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。