FE

平成29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）

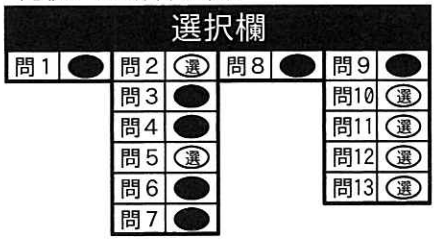
注意事項

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〕



   (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。

正　誤　表

平成29年10月15日実施

基本情報技術者試験　午後　問題

ページ	問題番号	行	誤	正	訂正の内容
22	5	上から5行目	入金は，買上げ前に行うことができる。	得意先は，入金を，買上げ前にも，買上げ後にも行うことができる。	下線部分を訂正する。

〔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	SSH による通信

●問 2～問 7（6 問中 4 問選択）

問題番号	出題分野	テーマ
問 2	ソフトウェア	プロセスの排他制御
問 3	データベース	会員制通信販売事業者における会員販売データ管理
問 4	ネットワーク	コールセンタ設備の構成案及び必要となるオペレータ数の検討
問 5	ソフトウェア設計	買上げ・入金管理システムを用いた月次集計処理
問 6	サービスマネジメント	情報システム運用サービスの予算策定と提示価格の計算
問 7	システム戦略	購買管理システムの導入による業務改善効果

●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	文字列の誤りの検出

●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）	回文の探索と表示
問 10	ソフトウェア開発（COBOL）	駐車場の自動精算システム
問 11	ソフトウェア開発（Java）	文字列の整列
問 12	ソフトウェア開発（アセンブラ）	ビット列の検索・置換
問 13	ソフトウェア開発（表計算）	サーバのアクセスログの分析

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　SSHによる通信に関する次の記述を読んで，設問1～4に答えよ。

SSH は遠隔ログインのための通信プロトコル及びソフトウェアであり，通信データの盗聴対策や，通信相手のなりすましを防ぐ仕組みを備えている。SSH では，サーバにログインしてデータをやり取りする通信（以下，ログインセッションという）に先立って，安全な通信経路の確立と利用者認証を行う必要がある。安全な通信経路の確立，利用者認証及びログインセッションを合わせて SSH セッションと呼ぶ。その流れを，図1に示す。

図1　SSHセッションの流れ

〔安全な通信経路の確立の概要〕

安全な通信経路の確立は，次のようにして行う。

(1) クライアントがサーバにアクセスする。

(2) サーバとクライアントが，SSH セッションで使用する暗号アルゴリズムについて合意する。

(3) サーバとクライアントが，通信データの暗号化に使用するセッション鍵と，他の SSH セッションと区別するためのセッション識別子について合意する。

(4) ①クライアントがサーバ認証を行う。サーバ認証では，クライアントがあらかじめ入手して正当性を確認しておいた［　a　］を用い，サーバによるセッション識別子へのディジタル署名が正しいかどうかを検証する。

(5) 合意した暗号アルゴリズムとセッション鍵を用いて，②共通鍵暗号方式による通信データの暗号化を開始する。これ以降の通信は，全て暗号化される。

〔利用者認証の概要〕

クライアントからサーバへのログインでは，サーバは利用者認証を行う。SSH の利用者認証の方式には，ディジタル署名を用いる“公開鍵認証”とパスワードを用いる“パスワード認証”がある。

“公開鍵認証”では，クライアントの公開鍵を事前にサーバに登録しておき，この登録されている公開鍵に対応する秘密鍵をクライアントがもっていることをサーバが確認する。この確認では，クライアントがセッション識別子などに対するディジタル署名をサーバに送信し，サーバが［ b ］を用いてディジタル署名を検証する。

“パスワード認証”では，クライアントが利用者 ID とパスワードを送信し，サーバは受け取ったパスワードが当該利用者のパスワードと一致していることを検証する。

なお，③“パスワード認証”は，“公開鍵認証”に比べて，安全性が低いと考えられている。

設問1　本文中の［　　］に入れる適切な答えを，解答群の中から選べ。

a，b に関する解答群

ア　安全な通信経路の確立時に合意したセッション鍵

イ　クライアントの公開鍵

ウ　クライアントの秘密鍵

エ　サーバの公開鍵

オ　サーバの秘密鍵

設問２　本文中の下線①によって防ぐことができる攻撃として適切な答えを，解答群の中から選べ。

解答群

ア　DoS攻撃　　イ　SQLインジェクション

ウ　クロスサイトスクリプティング　　エ　総当たり攻撃

オ　中間者攻撃

設問３　本文中の下線②について，通信データの暗号化に公開鍵暗号方式ではなく共通鍵暗号方式を用いる理由として適切な答えを，解答群の中から選べ。

解答群

ア　共通鍵暗号方式は，公開鍵暗号方式よりも暗号処理が高速である。

イ　共通鍵暗号方式は，公開鍵暗号方式よりも解読に時間が掛かる。

ウ　共通鍵暗号方式は，公開鍵暗号方式よりも鍵の再利用が容易である。

エ　共通鍵暗号方式は，公開鍵暗号方式よりも鍵の配布が容易である。

設問４　本文中の下線③のように考えられている理由として適切な答えを，解答群の中から選べ。

解答群

ア　“パスワード認証”では，サーバが攻撃者に乗っ取られていた場合，送信したパスワードを攻撃者に取得されてしまう。

イ　“パスワード認証”では，正当なサーバとは異なるサーバに接続させられてしまっても利用者が気づけない。

ウ　“パスワード認証”では，パスワードだけを用いるが，“公開鍵認証”では，パスワードの他にディジタル署名も用いる。

エ　“パスワード認証”では，利用者のパスワードが平文でネットワーク上を流れるので，盗聴されるとパスワードを取得されてしまう。

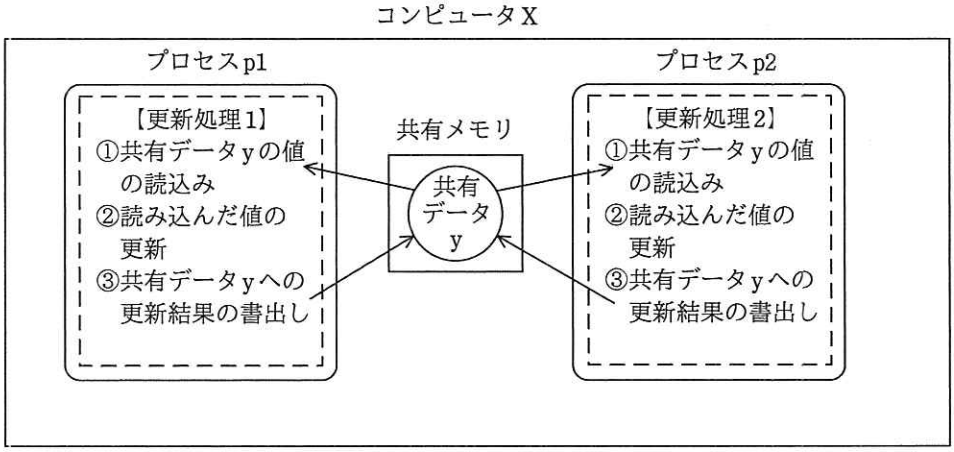
次の問2から問7までの6問については，この中から4問を選択し，選択した問題については，答案用紙の選択欄の(選)をマークして解答してください。

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　プロセスの排他制御に関する次の記述を読んで，設問1～3に答えよ。

単一の CPU と，プロセス間で共有するデータ（以下，共有データという）を格納するためのメモリ（以下，共有メモリという）をもつコンピュータ X において，並行に実行される複数のプロセスが，共有データを更新する場合を考える。

二つのプロセス p1，p2 が共有データ y を更新する際には，図 1 に示す更新処理 1，更新処理 2 を，それぞれ①～③ の順番に行う。



注記　“⟶”は共有データyへのアクセス（読込み又は書出し）を表す。

図1　二つのプロセスが共有データに対して更新処理を行う例

設問1　図1に示すプロセス p1 の更新処理 1 とプロセス p2 の更新処理 2 が，次のとおりに共有データ y を更新する場合を考える。

【更新処理 1】：共有データ y の値を 30 増加させる。

【更新処理 2】：共有データ y の値を 50 減少させる。

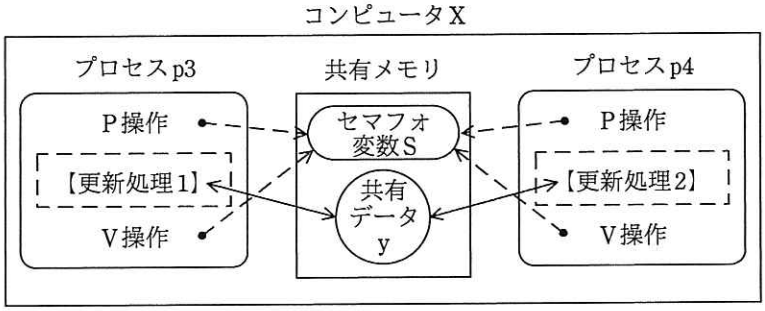
二つのプロセス p1，p2 の実行前の共有データ y の値が 100 であり，共有データ y に対して排他制御を行わずに，プロセス p1 が更新処理 1 を，プロセス p2 が更新処理 2 を並行して各 1 回だけ行った。結果として，共有データ y が**取り得ない値**を，解答群の中から選べ。

解答群

ア 50　　イ 80　　ウ 100　　エ 130

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。

共有データに対して排他制御を行う仕組みとして，セマフォ変数を利用する方法を考える。セマフォ変数は，対応する共有データが，解放されている状態を示す 1，又はいずれかのプロセスに確保されている状態を示す 0 の値をもつ。セマフォ変数の値は，P 操作と V 操作で変更する。共有データ y に対応するセマフォ変数 S を利用したプロセス p3，p4 が共有データ y を更新する際の動作を，図 2 に示す。



注記　“⟷”は共有データ y へのアクセスを表す。
“•－－→”はセマフォ変数 S への操作を表す。
更新処理 1 と更新処理 2 は，図 1 と同じ処理である。

図 2　セマフォ変数 S を利用した更新処理

図2において，共有データyに対して排他制御をした上で，更新処理を行うために，プロセスp3は，P操作，更新処理1，V操作の順に処理を行い，プロセスp4は，P操作，更新処理2，V操作の順に処理を行う。P操作は，［　a　］。V操作は，［　b　］。ここで，P操作とV操作は，それぞれ実行中の中断はないものとする。ただし，プロセスが待ち状態になれば，CPUは別のプロセスを実行させるものとする。

aに関する解答群

ア　セマフォ変数Sの値が0であれば1に変更し，終了する。1であれば0になるまで待った後，1に変更して終了する

イ　セマフォ変数Sの値が0であれば1に変更し，終了する。1であれば何もせずに終了する

ウ　セマフォ変数Sの値が1であれば0に変更し，終了する。0であれば1になるまで待った後，0に変更して終了する

エ　セマフォ変数Sの値が1であれば0に変更し，終了する。0であれば何もせずに終了する

bに関する解答群

ア　セマフォ変数Sの値が0であれば1になるまで待った後，終了する

イ　セマフォ変数Sの値が1であれば0になるまで待った後，終了する

ウ　セマフォ変数Sの値を0にして終了する

エ　セマフォ変数Sの値を1にして終了する

設問３　プロセス p5，p6 が更新する二つの共有データ y1，y2 があり，それぞれに異なるセマフォ変数を対応させて，個別に確保と解放ができるようにした場合を考える。共有データ y1 に対してセマフォ変数 S1 を，共有データ y2 に対してセマフォ変数 S2 を用いて排他制御を行う。プロセス p5 が，次の順序で共有データ y1，y2 に対する確保と解放を行うとき，プロセス p6 が，共有データ y1，y2 に対する確保と解放を行う順序によってはデッドロックが発生する可能性がある。デッドロックが発生する可能性がある，プロセス p6 の共有データ y1，y2 に対する確保と解放の順序を，解答群の中から選べ。

［プロセス p5 の共有データに対する確保と解放の順序］

y1 の確保，y2 の確保，y2 の解放，y1 の解放

解答群

ア　y1 の確保，y1 の解放，y2 の確保，y2 の解放

イ　y1 の確保，y2 の確保，y1 の解放，y2 の解放

ウ　y1 の確保，y2 の確保，y2 の解放，y1 の解放

エ　y2 の確保，y1 の確保，y1 の解放，y2 の解放

オ　y2 の確保，y2 の解放，y1 の確保，y1 の解放

選択した問題は，選択欄の(選)をマークしてください。マークがない場合は，採点されません。

問3　会員制通信販売事業者における会員販売データ管理に関する次の記述を読んで，設問1～4に答えよ。

清涼飲料水の会員制通信販売事業を運営するD社では，販売促進と商品管理の効率化を目的に会員情報や販売情報を管理するシステム（以下，販売管理システムという）を，事業開始当初から導入している。注文の受付は電話応対で行い，電話の受付時間は8時から20時までである。

販売管理システムで利用するデータベースの表構成とデータ格納例を，図1に示す。下線付きの項目は主キーである。

会員表

会員番号	氏名	生年	電話番号	郵便番号	住所
K20001	山田太郎	1960	03-3811-XXXX	112-0003	東京都文京区…
K20002	中村二郎	1953	03-3235-YYYY	160-0022	東京都新宿区…
⋮	⋮	⋮	⋮	⋮	⋮

販売表

伝票番号	販売日	会員番号	販売額
D0000001	2016-10-02	K20002	4200
D0000002	2016-10-02	K40027	2000
D0000003	2016-10-02	K20004	12700
⋮	⋮	⋮	⋮

販売明細表

伝票番号	商品番号	個数
D0000001	S1000001	30
D0000001	S1000002	10
D0000002	S3000001	1
⋮	⋮	⋮

商品表

商品番号	分類	商品名	単価
S1000001	コーヒー	ブラックコーヒー200mL	100
S1000002	コーヒー	エスプレッソ200mL	120
S3000001	ジュース	リンゴジュース1L 12本	2000
⋮	⋮	⋮	⋮

図1　販売管理システムで利用するデータベースの表構成とデータ格納例

設問1　販売促進のために，コーヒーの新商品案内のはがきを送ることになった。その際，購入しそうな会員に効率よく案内するために，2016年の1月1日から12月31日までの1年間において，分類がコーヒーである商品を5回以上購入し，かつ，その購入額の合計が10,000円以上である会員の氏名，郵便番号，住所を抽出することにした。ここで，1回の購入は販売明細表の1行に該当するものとする。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 会員表.氏名, 会員表.郵便番号, 会員表.住所
   FROM 会員表 WHERE 会員表.会員番号 IN
      (SELECT 販売表.会員番号
          FROM 販売表, 販売明細表, 商品表
          WHERE 販売表.伝票番号 = 販売明細表.伝票番号 AND
                商品表.商品番号 = 販売明細表.商品番号 [ a ] )
```

aに関する解答群

ア
```
AND 販売表.販売日 >= '2016-01-01' AND 販売表.販売日 <= '2016-12-31'
AND 商品表.分類 = 'コーヒー'
AND 商品表.単価 * 販売明細表.個数 >= 10000
GROUP BY 販売表.会員番号
HAVING COUNT(*) >= 5
```

イ
```
AND 販売表.販売日 >= '2016-01-01' AND 販売表.販売日 <= '2016-12-31'
AND 商品表.分類 = 'コーヒー'
GROUP BY 販売表.会員番号
HAVING SUM(商品表.単価 * 販売明細表.個数) >= 10000
  AND COUNT(*) >= 5
```

ウ
```
AND 販売表.販売日 >= '2016-01-01' AND 販売表.販売日 <= '2016-12-31'
GROUP BY 販売表.会員番号
HAVING 商品表.分類 = 'コーヒー'
  AND SUM(商品表.単価 * 販売明細表.個数) >= 10000
  AND COUNT(*) >= 5
```

エ
```
GROUP BY 販売表.会員番号
HAVING 販売表.販売日 >= '2016-01-01' AND 販売表.販売日 <= '2016-12-31'
  AND 商品表.分類 = 'コーヒー'
  AND SUM(販売表.販売額) >= 10000
  AND COUNT(*) >= 5
```

設問２　商品表の単価を何回でも変更できるようにする。併せて，販売時点の単価が分かるように，販売明細表の項目として販売時点の単価を追加することにした。変更した販売明細表の表構成を，図２に示す。商品表の単価の変更は，当日の受付時間前に行う。販売時点の単価の追加によって得ることができる情報として最も適切な答えを，解答群の中から選べ。

販売明細表

伝票番号	商品番号	個数	販売時単価

図２　変更した販売明細表の表構成

解答群

ア　ある時，ある商品をある会員が購入した単価と，その直後に変更された単価との価格差

イ　実際に購入された商品の，販売時点の単価の変遷

ウ　全ての商品の，単価の変遷

エ　全ての商品の，直近の単価変更日の前日における単価

設問３　商品表の単価を変更できるようにした後の販売状況を把握するために，2017年の１月１日から６月30日までの半年間を対象に，商品表の分類別の販売額の合計（合計販売額）を会員の年齢ごとに求めて，出力したい。年齢は2017から生年を引いた値とする。次のSQL文の［　　］に入れる正しい答えを，解答群の中から選べ。ここで，b1～b3に入れる答えは，b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。

```
SELECT 年齢, 分類, [ b1 ] AS 合計販売額
  FROM
    (SELECT 2017 - 会員表.生年 AS 年齢, 商品表.分類, [ b2 ]
       FROM 会員表, 販売表, 販売明細表, 商品表
       WHERE 会員表.会員番号 = 販売表.会員番号 AND
             販売表.伝票番号 = 販売明細表.伝票番号 AND
             販売明細表.商品番号 = 商品表.商品番号 AND
             販売表.販売日 >= '2017-01-01' AND
             販売表.販売日 <= '2017-06-30'
    ) FACTTB
  GROUP BY [ b3 ]
ORDER BY 年齢 ASC, 合計販売額 DESC
```

bに関する解答群

	b1	b2	b3
ア	SUM(単価 * 個数)	商品表.単価，販売明細表.個数	年齢，分類
イ	SUM(単価 * 個数)	商品表.単価，販売明細表.個数	年齢，分類，合計販売額
ウ	SUM(販売額)	販売表.販売額	年齢，分類
エ	SUM(販売額)	販売表.販売額	年齢，分類，合計販売額
オ	SUM(販売時単価 * 個数)	販売明細表.販売時単価，販売明細表.個数	年齢，分類
カ	SUM(販売時単価 * 個数)	販売明細表.販売時単価，販売明細表.個数	年齢，分類，合計販売額

設問4　入荷情報を管理するシステム（以下，入荷管理システムという）を販売管理システムと同時に運用開始している。入荷管理システムで利用するデータベースの表構成を図3に示す。

ビュー入荷集計表は運用開始から現在までの入荷数の総数を表示する。さらに販売総数を把握するためにビュー販売集計表を，最新の在庫数を把握するためにビュー在庫表を作成する。ビュー在庫表は一度でも入荷した商品は在庫数ゼロでも表示する仕様である。データベースに追加する表の構成を，図 4 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。ここで，c1 と c2 に入れる答えは，c 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。

図 1～図 3 の表を用いて，図 4 のビュー販売集計表を作成するための必要最小限の表の数は［ c1 ］である。図 4 のビュー在庫表は，ビュー販売集計表を用いて作成する。このとき，ビュー在庫表を作成するための必要最小限の表の数は，ビュー販売集計表も含めて［ c2 ］である。

入荷表

入荷番号	商品番号	入荷日	入荷数

ビュー入荷集計表

商品番号	入荷総数

図3　入荷管理システムで利用するデータベースの表構成

ビュー販売集計表

商品番号	販売総数

ビュー在庫表

商品番号	在庫数

図4　入荷管理システムのデータベースに追加する表の構成

c に関する解答群

	c1	c2
ア	1	2
イ	1	3
ウ	2	2
エ	2	3

選択した問題は，選択欄の(選)をマークしてください。マークがない場合は，採点されません。

問4　コールセンタ設備の構成案及び必要となるオペレータ数の検討に関する次の記述を読んで，設問1，2に答えよ。

X社のコールセンタでは，顧客からの問合せの電話に対して30人のオペレータで処理を行っているが，“電話で待たされる”という顧客からの意見が多く寄せられている。この度，顧客が電話で待たされる時間の短縮を目的に，既存のコールセンタの設備とオペレータ数を見直すことにした。検討の要件とコールセンタ設備の構成案は，次のとおりである。

〔検討の要件〕

(1) コールセンタでは，IPをベースにして拡張が可能なVoIP（Voice over Internet Protocol）を採用する。

(2) コールセンタに着信があってからオペレータと通話を開始できるまでの平均待ち時間を，問合せが最も多い時間帯（以下，ピーク時という）においても，20秒以下になるようにオペレータを増員する。

〔コールセンタ設備の構成案〕

VoIPサーバ群は，顧客情報サーバ，コールサーバ及び着信分配情報サーバで構成する。オペレータ席には，それぞれ電話端末とPCを設置する。VoIPサーバ群と，音声ゲートウェイ及びオペレータ席との間はレイヤ3スイッチで接続する。

コールセンタ設備の構成案を，図１に示す。

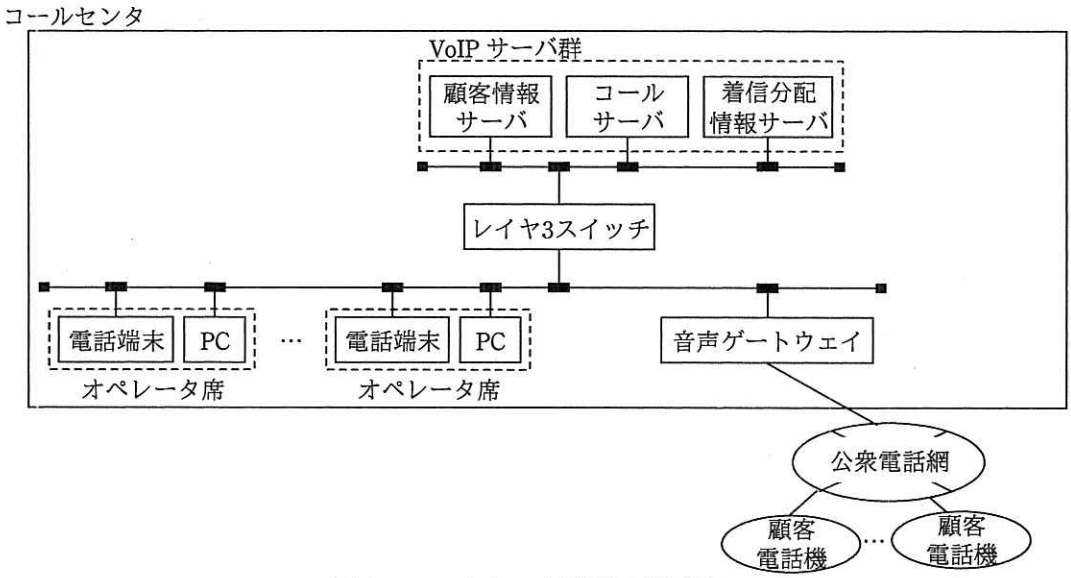


図１　コールセンタ設備の構成案

顧客情報サーバは，過去の問合せ内容の検索や，顧客とのやり取りを記録するために利用される。コールサーバは VoIP の制御を行い，音声ゲートウェイから電話の接続要求があると，オペレータ席にある電話端末と PC の状態（以下，オペレータ席の状態という）を管理している着信分配情報サーバに問い合わせて，接続要求待ちのオペレータ席の電話端末へ接続する。オペレータ席の状態には，接続要求待ち，通話中及び結果記録中の状態があり，着信分配情報サーバは各オペレータ席がどの状態にあるかを認識している。

顧客からの通話要求をオペレータ席の電話端末に接続するシーケンスを，図２に示す。

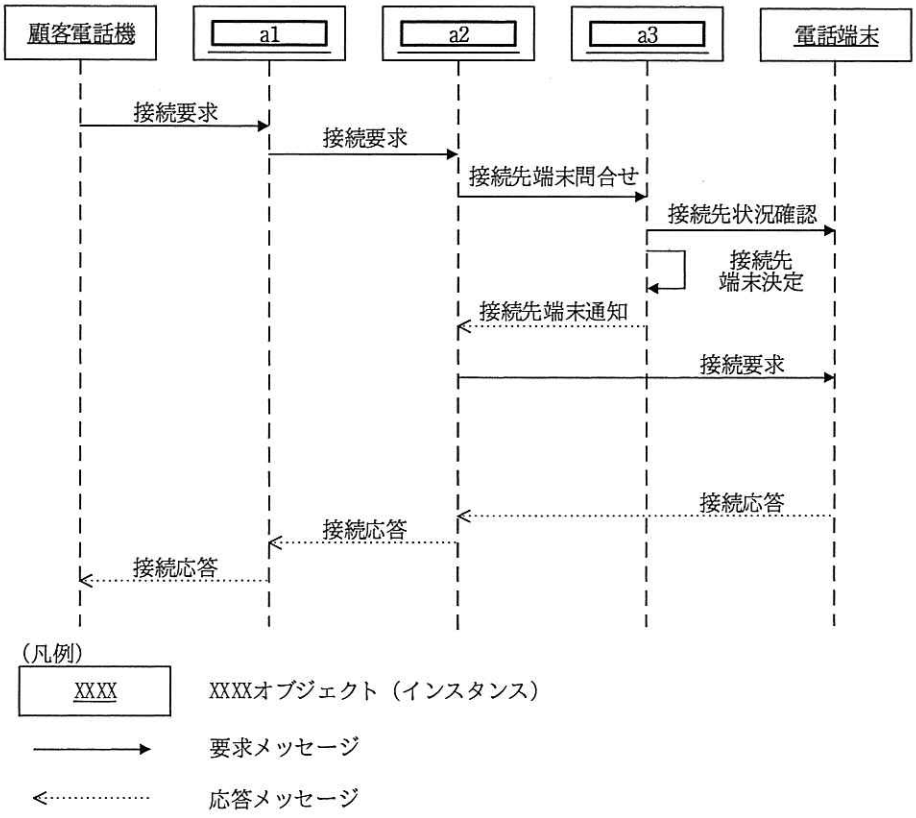


図２　顧客からの通話要求をオペレータ席の電話端末に接続するシーケンス

設問１　図 2 中の［　　］に入れる正しい答えを，解答群の中から選べ。ここで，a1 ～ a3 に入れる答えは，a 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。

a に関する解答群

	a1	a2	a3
ア	音声ゲートウェイ	コールサーバ	顧客情報サーバ
イ	音声ゲートウェイ	コールサーバ	着信分配情報サーバ
ウ	コールサーバ	音声ゲートウェイ	着信分配情報サーバ
エ	着信分配情報サーバ	音声ゲートウェイ	顧客情報サーバ
オ	着信分配情報サーバ	コールサーバ	顧客情報サーバ

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。

〔オペレータ席の状態と問合せの処理件数及び処理時間〕

接続要求待ちの状態にあるオペレータ席の電話端末に接続要求があると，オペレータが顧客と通話を開始することによって通話中の状態になる。通話が終了すると結果記録中の状態となり，オペレータは PC から顧客情報サーバに，その結果の記録を行う。結果の記録が完了すると，接続要求待ちの状態に戻る。

X 社のコールセンタが想定する問合せの処理件数及び平均処理時間を表 1 に示す。1 人のオペレータが 1 回の問合せに応対する通話時間と，結果記録時間の合計を処理時間とする。想定する平均処理時間は［　b　］分であり，コールセンタのピーク時における平均着信間隔は［　c　］秒となる。

表 1　想定する問合せの処理件数及び平均処理時間

ピーク時の着信件数（件／時）	360
平均通話時間（分／件）	4
平均結果記録時間（分／件）	2

〔必要なオペレータ数の検討〕

コールセンタに顧客からの着信があってから，オペレータと通話を開始できるまでの平均待ち時間の算出には，待ち行列理論を使う。窓口数 s の M/M/s モデルに表 1 の条件を適用すると，オペレータ数と平均待ち行列長の関係は，表 2 のようになることが分かっている。

表 2　オペレータ数と平均待ち行列長の関係

オペレータ数（人）	平均待ち行列長
40	3.704
41	2.304
42	1.474
43	0.957
44	0.626

平均到着率（平均着信間隔の逆数）をλとすると，平均待ち行列長 L と平均待ち時間 W との間には，次のリトルの公式が成立する。

$$\lambda \times W = L$$

リトルの公式と表 2 を用いると，ピーク時の平均待ち時間を 20 秒以下にするために必要なオペレータ数は最少 [d] 人となる。ここで，接続要求待ちの状態にあるオペレータ席が一つ以上ある場合に，コールセンタに顧客からの着信があってからオペレータ席の電話端末に接続要求を行うまでの時間と，オペレータが接続応答を行ってから顧客と通話を開始できるまでの時間は，無視できるほど短いものとする。

b，c に関する解答群

ア 3	イ 4	ウ 5	エ 6	オ 7
カ 8	キ 9	ク 10	ケ 11	コ 12

d に関する解答群

ア 40	イ 41	ウ 42	エ 43	オ 44

選択した問題は，選択欄の(選)をマークしてください。マークがない場合は，採点されません。

問5　買上げ・入金管理システムを用いた月次集計処理に関する次の記述を読んで，設問1，2に答えよ。

E社は，10年以上の取引実績がある得意先とは，双方が合意した上で，得意先の買上げに対する支払を，1回の入金で行うことも，複数回の入金で行うことも許容した取引を行っている。入金は，買上げ前に行うことができる。E社の担当者は，得意先の買上げ及び入金が発生するたびに，買上げ・入金管理システムに記録している。毎月の最終営業日の取引終了後に，買上げ・入金管理システムを用いて月次集計処理を行っている。月次集計処理の概要は次のとおりである。

〔月次集計処理の概要〕

月次集計処理は月間買上げ額確定処理と消込み処理から成り，月間買上げ額確定処理の終了後に消込み処理を行う。

月間買上げ額確定処理は，得意先ごとに当月1か月分の買上げ金額を合計して，当月の買上げの合計額（以下，月間買上げ額という）を確定させる処理である。

消込み処理は，充当が完了していない月間買上げ額に対して，得意先からの入金を充当していく処理である。

E社の得意先との取引では，過去の月間買上げ額に対する充当が完了していない場合や，逆に得意先から入金された金額が，全ては月間買上げ額に充当されずに残っている場合がある。したがって，消込み処理では，充当が完了していない月間買上げ額に対して，買上げ年月の古い順に，まだ入金額の全てを充当しきれていない入金を，入金日が古い順に充当していく。

〔買上げ・入金管理システムのデータベース〕

買上げ・入金管理システムでは，関係データベースの表で必要なデータを管理している。関係データベースの E-R 図を図 1 に，各表の説明を表 1 に，主な表の属性の説明を表 2 に示す。

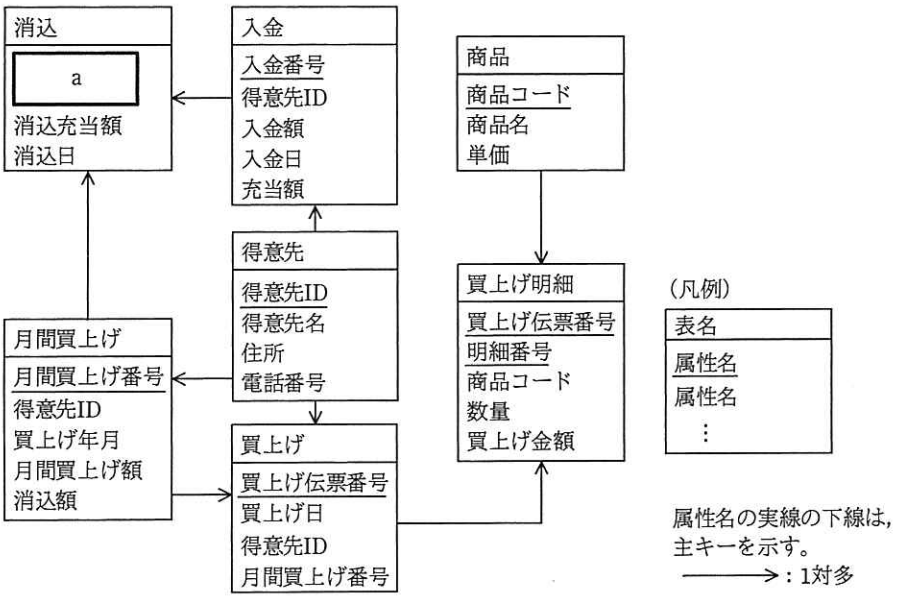


図 1　関係データベースの E-R 図

表 1　各表の説明

表名	管理する情報
得意先	得意先の情報
商品	E 社が取り扱う商品の情報
買上げ	1 伝票単位の買上げ情報
買上げ明細	1 伝票を構成する商品単位の買上げ情報
月間買上げ	得意先の月間買上げ額と，月間買上げ額に対して充当された金額（消込額）に関する情報
入金	得意先からの入金額と，その入金を月間買上げ額に充当した金額（充当額）に関する情報
消込	ある入金からある月間買上げに対して，充当した金額（消込充当額）の情報

表2　主な表の属性の説明

表名	属性名	説明
月間買上げ	月間買上げ額	ある得意先の，ある年月1か月分の買上げ金額
	消込額	月間買上げ額のうち，既に充当された金額
入金	入金額	ある得意先から入金された金額
	充当額	入金額のうち，月間買上げに既に充当した金額
消込	消込充当額	ある入金からある月間買上げに対して，消込み処理で充当した金額

設問1　図1中の［　　］に入れる属性の組みとして適切な答えを，解答群の中から選べ。

aに関する解答群

ア　消込番号
　　月間買上げ番号

イ　消込番号
　　入金番号

ウ　得意先ID
　　消込番号

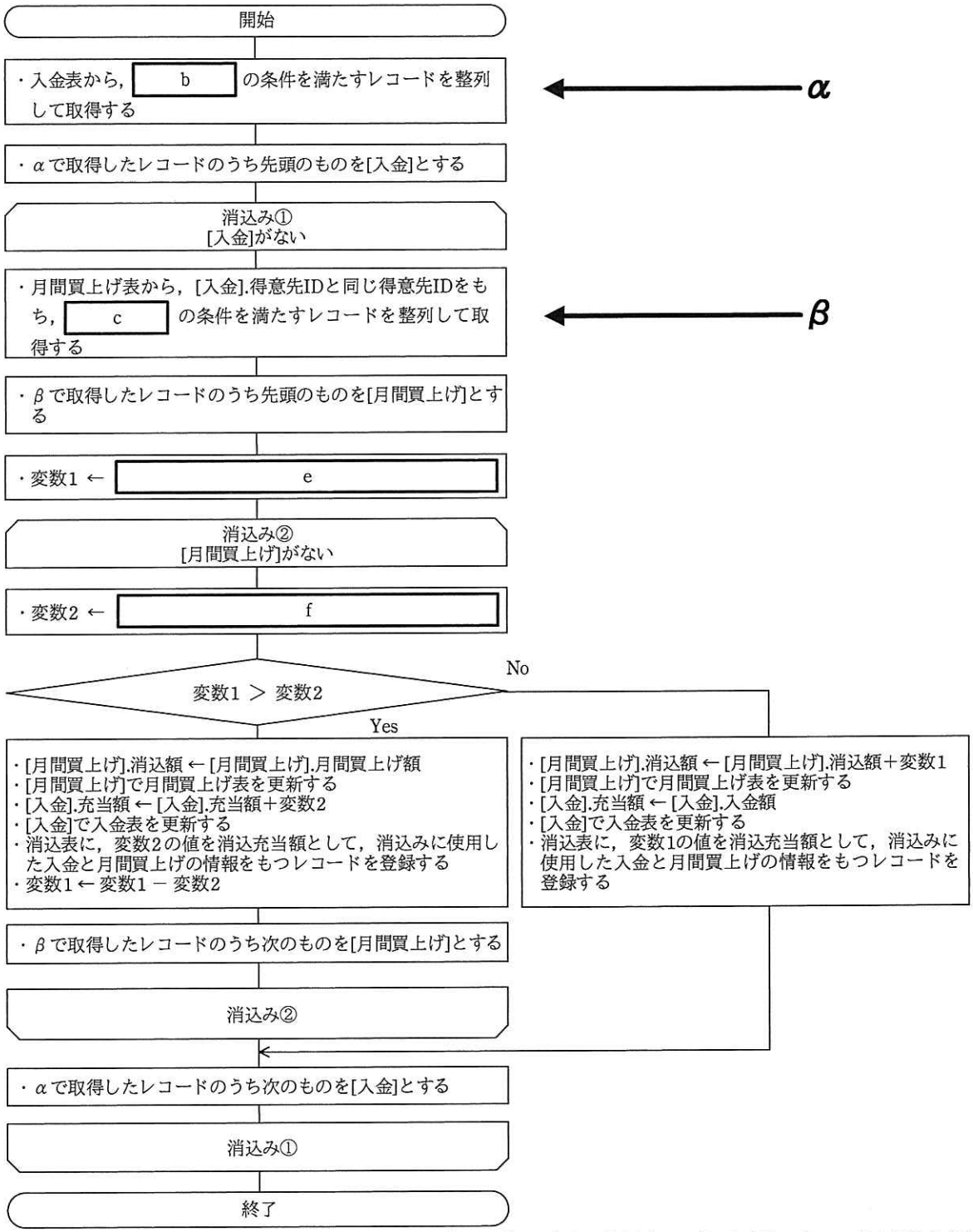
エ　得意先ID
　　月間買上げ番号

オ　得意先ID
　　入金番号

カ　入金番号
　　月間買上げ番号

設問2　図2は，消込み処理の流れ図である。次の記述及び図2中の［　　］に入れる適切な答えを，解答群の中から選べ。

図2中の**α**の部分では，入金表から［　b　］の条件を満たすレコードを，得意先IDの昇順，入金日の昇順に整列して取得する。消込み①の繰返し処理では，取得したレコードを順に処理する。**β**の部分では，消込み①の繰返しで処理中のレコードの得意先IDと同じ得意先IDをもつ月間買上げ表のレコードのうち，［　c　］の条件を満たすレコードを，［　d　］に整列して取得する。消込み②の繰返し処理では，取得したレコードを順に処理する。



注記　“[X]”という記述でX表の処理中のレコードを表し，“[X].Y”という記述でX表の処理中のレコードの属性Yを表す。

図2　消込み処理の流れ図

bに関する解答群

ア　入金額 ＝ 充当額

イ　入金額 ＞ 充当額

ウ　入金額 ＜ 充当額

エ　充当額 ＝ 0

cに関する解答群

ア　月間買上げ額 ＞ 0

イ　月間買上げ額 ＞ 消込額

ウ　消込額 ＝ 0

エ　消込額 ＞ 0

オ　消込額 ＝ 月間買上げ額

カ　消込額 ＞ 月間買上げ額

dに関する解答群

ア　買上げ年月の降順　　イ　買上げ年月の昇順　　ウ　消込額の降順

エ　月間買上げ額の昇順　　オ　得意先 ID の昇順

e，fに関する解答群

ア　[月間買上げ].月間買上げ額 ＋ [月間買上げ].消込額

イ　[月間買上げ].月間買上げ額 － [月間買上げ].消込額

ウ　[月間買上げ].消込額 － [月間買上げ].月間買上げ額

エ　[入金].入金額 ＋ [入金].充当額

オ　[入金].入金額 － [入金].充当額

カ　[入金].充当額 － [入金].入金額

選択した問題は，選択欄の(選)をマークしてください。マークがない場合は，採点されません。

問6　情報システム運用サービスの予算策定と提示価格の計算に関する次の記述を読んで，設問1～3に答えよ。

A社は，顧客であるB社に対して，情報システムの運用サービス（以下，B社サービスという）を提供している。A社の運用サービス部では，B社サービスの提供に必要な次年度の費用を見積もり，見積もった費用（以下，予算という）を営業部に提出している。毎年度，営業部では，運用サービス部で作成した予算を基に，所定の利益率が維持できるようにB社サービスの提示価格を算出してB社と交渉している。近年は，B社から価格を下げるよう要求されることが多くなってきている。

B社サービスに関する2015～2017年度の3年度分の予算と実際に掛かった費用（以下，実績という）は，表1のとおりである。ただし，2017年度の実績は見込みであるが，ここでは実績と呼ぶ。

表1　B社サービスに関する2015～2017年度の3年度分の予算と実績

単位　千円

費目	2015年度		2016年度		2017年度	
	予算	実績	予算	実績	予算	実績
人件費	3,000	3,500	2,800	3,000	4,000	4,400
サーバ費	1,200	1,250	1,250	1,400	1,400	1,250
PC費	500	500	500	500	640	640
ネットワーク費	200	220	250	240	250	250
その他経費	1,000	1,200	1,000	1,100	1,000	1,000
合計	5,900	6,670	5,800	6,240	7,290	7,540

設問1　運用サービス部では，B社サービスに関する2018年度の予算を作成するに当たって，表1を用いて2015～2017年度の3年度分の予算と実績に関する傾向を分析した。正しい答えを，解答群の中から選べ。

解答群

ア　人件費の実績は，3年度とも，各年度の実績の合計の過半を占めている。

イ　サーバ費の実績は，2年度連続で上がっている。

ウ　PC 費の実績は，2016 年度の前年度に対する増分よりも，2017 年度の前年度に対する増分の方が小さい。

エ　ネットワーク費の予算は2年度連続で下がっているが，ネットワーク費の実績は2年度連続で上がっている。

オ　その他経費は，各費目中で，予算も実績も 2015 年度は 2 番目に大きい費目であったが，2017 年度は3番目に大きい費目となっている。

カ　各費目の実績の合計は，3年度とも，各費目の予算の合計を上回っている。

設問2　B 社サービスに関する 2018 年度の予算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。

運用サービス部では，B 社サービスに関する 2018 年度の予算を次のとおり作成した。

(1)　人件費の予算は，2016 年度の人件費の実績と同じとする。

(2)　サーバ費の予算は，2015～2017 年度のサーバ費の実績の平均とする。

(3)　PC 費の予算は，2017 年度の PC 費の実績と同じとする。

(4)　ネットワーク費の予算は，ネットワーク費の実績を用いて，2016 年度に対する 2017 年度の増分を 2017 年度の実績に加えたものとする。

(5)　その他経費の予算は，2015～2017 年度のその他経費の実績の平均とする。

運用サービス部で作成した B 社サービスに関する 2018 年度の予算を，表 2 に示す。

表2　B社サービスに関する2018年度の予算

単位　千円

費目	予算
人件費	
サーバ費	1,300
PC費	
ネットワーク費	a
その他経費	1,100
合計	6,300

注記　網掛けの部分は表示していない。

2018年度の各費目の予算が予算の合計に占める割合を，2017年度の各費目の実績が実績の合計に占める割合と比較すると，［　b　］。

aに関する解答群

ア　220　　イ　230　　ウ　240　　エ　250　　オ　260　　カ　270

bに関する解答群

ア　サーバ費の割合とその他経費の割合が上がって，それら以外の費目の割合が下がっている

イ　サーバ費の割合とその他経費の割合が下がって，それら以外の費目の割合が上がっている

ウ　人件費の割合が上がって，それ以外の費目の割合が下がっている

エ　人件費の割合が下がって，それ以外の費目の割合が上がっている

オ　人件費の割合とサーバ費の割合が上がって，それら以外の費目の割合が下がっている

カ　人件費の割合とサーバ費の割合が下がって，それら以外の費目の割合が上がっている

設問3　2018 年度の提示価格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。

営業部では，運用サービス部が作成した予算を基に，利益率が 10%となるように B 社サービスの提示価格を算出した。2018 年度の提示価格は，［ c ］千円となった。ここで，利益率は，提示価格から予算を引いた額を提示価格で割った値であり，100 を乗じて%表示する。

営業部が B 社に B 社サービスの提示価格を提案したところ，提示価格から 10%低い価格（以下，要求価格という）を要求された。

運用サービス部と営業部で検討した結果，サービスレベルの変更について B 社と合意できれば，その他経費を 10%，人件費を 5%削減できることが分かった。この場合，2018 年度の予算の合計は［ d ］削減となり，要求価格と同額を提示価格とすると，利益率は［ e ］。

また，作業の一部を自動化することによって，人件費を先の 5%と合わせて 15%削減できることが分かった。ただし，この場合には，サーバ費については 5%上がる見込みである。運用サービス部では，先のその他経費の 10%削減と合わせて B 社サービスの 2018 年度の予算を再度作成した。このとき，提示価格を要求価格と同額にすると，利益率は［ f ］%になる。ここで，%表示する値は，利益率に 100 を乗じて小数第 2 位で四捨五入したものである。

c に関する解答群

ア　5,670　　イ　6,300　　ウ　6,930　　エ　7,000

d に関する解答群

ア　10%未満の　　イ　10%の　　ウ　10%よりも大きな

eに関する解答群

ア　上がる

イ　変わらない

ウ　下がるがゼロ以下にはならない

エ　下がってゼロになる

オ　下がってマイナスになる

fに関する解答群

ア　5.5　　イ　6.1　　ウ　7.1　　エ　7.9　　オ　8.0　　カ　8.9

E
後

選択した問題は，選択欄の(選)をマークしてください。マークがない場合は，採点されません。

問7　購買管理システムの導入による業務改善効果に関する次の記述を読んで，設問 1，2に答えよ。

日用品メーカの C 社では，ソフトウェアパッケージを用いた購買管理システムの導入を進めており，システムの要件定義を終えたところである。定義された要件には，ソフトウェアパッケージの機能を活用できる要件と，追加でシステム開発が必要な要件（以下，システム要件という）がある。全てのシステム要件を実現すると，開発費が当初の予算を超えることが分かった。そこで，各システム要件を実現した場合に削減される1年間のコスト（以下，年間効果という）と必要な開発費とを併せて評価し，実現するシステム要件を決定することにした。この決定に当たっては，購買管理システムの保守運用費は考慮しないものとする。

C社は，システム要件8件について評価することにした。

設問1　システム要件を実現（以下，システム化という）するための開発費と年間効果の試算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，c1 と c2 に入れる答えは，c 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。

(1)　システム化するための開発費と年間効果を試算する。また，各システム要件を，“必ず実現したい”という要件（以下，必須要件という）と，“できれば実現したい”という要件（以下，要望要件という）に分類する。

(2)　必須要件の一つである“仕入先からの注文請書(うけしょ)を購買管理システムで処理できるようにする”の年間効果を試算する。購買管理システムは仕入先も利用するものとし，システム化によって，C 社での注文請書の確認と再提出依頼の作業は次のようになる。

① 仕入先からファックスで送付されてきた注文請書の内容と注文書の内容の目視での照合に代えて，購買管理システム上のデータチェックで照合を行う。

② データチェックで誤りが見つかったときに，仕入先への注文請書の再提出の依頼を Web 画面で行う。

③ 再提出された注文請書の内容と注文書の内容の照合（以下，再照合という）も購買管理システムで行う。

上記 ①～③ の年間効果を試算するために，注文請書の確認と再提出依頼の作業状況を調査して条件を設定した。その結果を，表 1 に示す。将来値はシステム化後の作業状況を想定して設定した値である。ここで，再照合時に誤りは発生しないものとする。

表 1　年間効果を試算するための条件

試算条件	現状値	将来値
注文書に対して提出される注文請書枚数（枚／日）	200	200
注文請書の誤りの発生枚数（枚／日）	18	3
注文請書と注文書との目視での照合及び再照合の作業に要する時間（分／枚）	5	0
誤りが見つかった注文請書の再提出の依頼作業に要する時間（分／枚）	10	5
作業に要する人件費（円／時間）	1,400	1,400

(3) 表 1 の将来値を設定するに当たり，注文請書と注文書との目視での照合及び再照合の作業に要する時間は，システム化によって［ a ］と想定した。また，仕入先が Web を介して注文書の確認と注文請書の送信を行うことによって，注文請書の誤りの発生枚数は［ b ］と想定した。

1 年当たりの稼働日数を 240 日とすると，現状で注文請書の確認，再提出依頼及び再照合の作業に要する年間費用は［ c1 ］千円であり，システム化された場合は［ c2 ］千円となって，この差額が年間効果となる。

全てのシステム要件に対する年間効果の試算と分類の結果を，表 2 に示す。

表２　システム要件に対する年間効果の試算と分類の結果

単位　千円

要件 ID	システム要件	開発費	年間効果	分類
SYS901		5,800	4,800	必須要件
SYS902		6,200	7,320	必須要件
SYS903		1,200	670	要望要件
SYS904	仕入先からの注文請書を購買管理システムで処理できるようにする	12,400		必須要件
SYS905		2,750	1,640	要望要件
SYS906		4,300	520	要望要件
SYS907		1,630	1,280	要望要件
SYS908		1,800	2,160	要望要件

注記　網掛けの部分は表示していない。

a，b に関する解答群

ア　増加する　　イ　減少するがゼロにならない

ウ　不要になる　　エ　変わらない

c に関する解答群

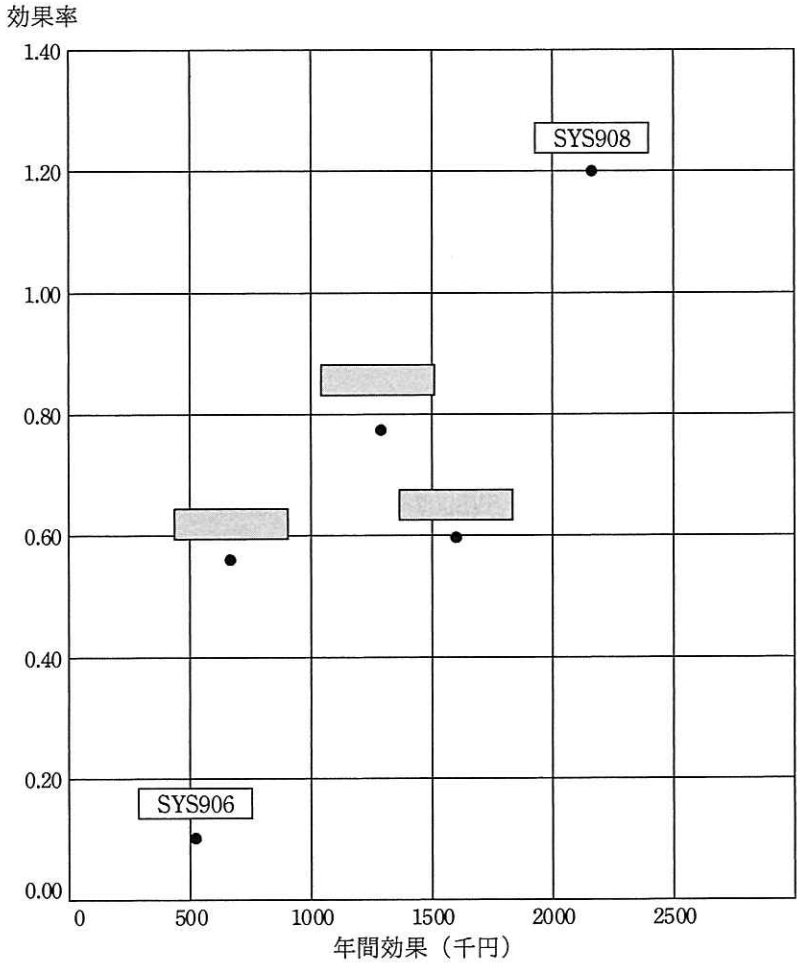
	c1	c2
ア	6,608	84
イ	6,608	6,524
ウ	7,028	6,524
エ	7,112	84
オ	7,112	7,028

設問２　開発費と年間効果の評価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。

表２に示した三つの必須要件については全て採用するものとし，五つの要望要件については開発費と年間効果を評価して，採用，不採用を決定することにした。

必須要件を含めて購買管理システムのシステム化のための開発費が予算の30,000千円に収まるように要望要件を選択することにした。

選択に当たり，開発費に対する年間効果の割合（以下，効果率という）も選択条件とした。要望要件に分類したシステム要件の年間効果と効果率を，図1に示す。



注記　網掛けの部分は表示していない。

図1　要望要件に分類したシステム要件の年間効果と効果率

(1)　SYS908は，効果率も年間効果も最も大きく，開発費が予算に収まるので，採用とした。また，効果率も年間効果も最も小さい SYS906 は不採用とし

た。

(2) 続いて，残りの三つの要望要件から，開発費が予算に収まり，年間効果が最大になるような選択を行うこととした。

① SYS905 を採用したとき，開発費の予算を考えると［ d ］。この場合，開発費の総額は 28,950 千円となり，年間効果は 22,948 千円となる。

② 一方，SYS907 を採用したとき，開発費の予算を考えると［ e ］。この場合，開発費の総額は 29,030 千円となり，年間効果は 23,258 千円となる。

(1)，(2)の評価の結果，開発費が予算に収まり，年間効果を最大にするには，要望要件は SYS908 に加えて［ f ］を採用すればよい。

d に関する解答群

ア SYS903 も SYS907 も採用できる

イ SYS903 も SYS907 も採用できない

ウ SYS903 は採用できるが，SYS907 は採用できない

エ SYS907 は採用できるが，SYS903 は採用できない

e に関する解答群

ア SYS903 も SYS905 も採用できる

イ SYS903 も SYS905 も採用できない

ウ SYS903 は採用できるが，SYS905 は採用できない

エ SYS905 は採用できるが，SYS903 は採用できない

f に関する解答群

ア SYS903 と SYS905　　イ SYS903 と SYS907

ウ SYS905　　エ SYS907

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１～４に答えよ。

文字列の誤りを検出するために，N 種類の文字に 0，1，…，N－1 の整数値を重複なく割り当て，検査文字を生成するプログラムと，元となる文字列の末尾に検査文字を追加した検査文字付文字列を検証するプログラムである。ここで扱う 30 種類の文字，及び文字に割り当てた数値を，表１に示す。空白文字は“␣”と表記する。

表１　文字，及び文字に割り当てた数値

文字	␣	.	,	?	a	b	c	d	e	f	g	h	i	j	k
数値	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14

文字	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
数値	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29

〔プログラムの説明〕

検査文字の生成と検査文字付文字列の検証の手順を示す。

(1)　検査文字の生成

①　文字列の末尾の文字を１番目の文字とし，文字列の先頭に向かって奇数番目の文字に割り当てた数値を 2 倍して N で割り，商と余りの和を求め，全て足し合わせる。

②　偶数番目の文字に割り当てた数値は，そのまま全て足し合わせる。

③　①と②の結果を足し合わせる。

④　N から，③で求めた総和を N で割った余りを引く。さらにその結果を，N で割り，余りを求める。求めた数値に対応する文字を検査文字とする。

(2) 検査文字付文字列の検証

① 検査文字付文字列の末尾の文字を１番目の文字とし，文字列の先頭に向かって偶数番目の文字に割り当てた数値を 2 倍して N で割り，商と余りの和を求め，全て足し合わせる。

② 奇数番目の文字に割り当てた数値は，そのまま全て足し合わせる。

③ ①と②の結果を足し合わせる。

④ ③で求めた総和が N で割り切れる場合は，検査文字付文字列に誤りがないと判定する。N で割り切れない場合は，検査文字付文字列に誤りがあると判定する。

〔検査文字付文字列の生成例〕

表 1 及び検査文字の生成の手順を用いることによって，文字列 ipa␣␣ に対し，生成される検査文字は f である。

検査文字付文字列は，文字列の末尾に検査文字を追加し，ipa␣␣f となる。

〔プログラムの仕様〕

各関数の仕様を(1)～(4)に示す。ここで，配列の添字は１から始まるものとする。

(1) 関数 calcCheckCharacter は，文字列及び文字列長を用いて生成した検査文字を返す。関数 calcCheckCharacter の引数及び返却値の仕様は，表 2 のとおりである。

表 2 関数 calcCheckCharacter の引数及び返却値の仕様

引数／返却値	データ型	入力／出力	説明
input[]	文字型	入力	文字列が格納されている１次元配列
len	整数型	入力	文字列の文字列長（1 以上）
返却値	文字型	出力	生成した検査文字を返す。

関数 calcCheckCharacter は，関数 getValue，関数 getChar を使用する。

(2) 関数 validateCheckCharacter は，検査文字付文字列を検証し，検証結果を返す。

関数 validateCheckCharacter の引数及び返却値の仕様は，表3のとおりである。

表3 関数 validateCheckCharacter の引数及び返却値の仕様

引数／返却値	データ型	入力／出力	説明
input[]	文字型	入力	検査文字付文字列が格納されている 1 次元配列
len	整数型	入力	検査文字付文字列の文字列長（2以上）
返却値	論理型	出力	検査文字付文字列に誤りがないと判定した場合は true，誤りがあると判定した場合は false を返す。

関数 validateCheckCharacter は，関数 getValue を使用する。

(3) 関数 getValue は，表1に従い，引数として与えられた文字に割り当てた数値を返す。

(4) 関数 getChar は，表 1 に従い，引数として与えられた数値に対応する文字を返す。

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。ここで，a1 と a2 に入れる答えは，a 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。

〔プログラム〕

```
○文字型関数： calcCheckCharacter(文字型: input[], 整数型: len)
○整数型： N, sum, i, value, check_value
○論理型： is_even
・N ← 30
・sum ← 0
・is_even ← false
■ i: len, i > 0, −1
 ・value ← getValue(input[i])
 ▲ is_even = [ a1 ]
 |  ・sum ← sum + value
 ——
 ▼  ・sum ← sum + (value × 2) ÷ N + (value × 2) % N
 ・is_even ← not is_even
■
・check_value ← [ b ]
・return getChar(check_value)
```

```
○論理型関数： validateCheckCharacter(文字型: input[], 整数型: len)
○整数型： N, sum, i, value
○論理型： is_odd, ret_value
・N ← 30
・sum ← 0
・is_odd ← true
・ret_value ← true
■ i: len, i > 0, −1
 ・value ← getValue(input[i])
 ▲ is_odd = [ a2 ]
 |  ・sum ← sum + value
 ——
 ▼  ・sum ← sum + (value × 2) ÷ N + (value × 2) % N
 ・is_odd ← not is_odd
■
▲ [ c ]
|  ・ret_value ← false
▼
・return ret_value
```

aに関する解答群

	a1	a2
ア	false	false
イ	false	true
ウ	true	false
エ	true	true

bに関する解答群

ア　N − sum % N　　イ　sum % N

ウ　(N − sum % N) % N　　エ　(sum − N) % N

cに関する解答群

ア　sum ÷ N = 0　　イ　sum ÷ N ≠ 0

ウ　sum % N = 0　　エ　sum % N ≠ 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。

本プログラムでは，検査文字付文字列の誤りが1文字であれば，誤りを検出できる。しかし，複数の文字に誤りがある場合には，誤りがないと判定されることがある。例えば，関数 validateCheckCharacter で表4に示す検査文字付文字列を検証した場合，誤りがないと判定されるケースは［　d　］。ここで，文字列 ipa␣␣ に対し生成される検査文字は f である。

表4　検査文字付文字列

ケース	検査文字付文字列
1	ipb␣␣f
2	api␣␣f
3	pia␣␣f
4	␣␣apif

dに関する解答群

ア　1と2と3と4である　　イ　2である　　ウ　2と3である

エ　2と3と4である　　オ　2と4である　　カ　ない

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。

本プログラムを文字列長が同じである複数の文字列に対して適用することを考える（図1参照）。

列

行	1	2	3	4	5	6
1	i	p	a	␣	␣	
2	t	e	s	t	s	
3	m	a	k	e	␣	
4	i	t	.	␣	␣	
5						

図1　作成中の検査文字付表

〔考え方〕

文字列長がnである m個の文字列について考える。文字列に対して，（m＋1）行（n＋1）列の表を用意する。以後，この表を検査文字付表という。

(1)　検査文字の生成

例えば，文字列長が5である4個の文字列 ipa␣␣，tests，make␣，it.␣␣を，図1の太枠内のように，各文字列の先頭の位置を最左列に揃え，各文字列を上の行から順に格納して，表を作成する。この表の太枠内の各行各列をそれぞれ文字列とみなして検査文字を生成し，最右列と最下行に格納する。

この手順で作成した検査文字付表を図2に示す。作成した検査文字付表の5行5列目（網掛け部分）の検査文字は［　e　］である。

行＼列	1	2	3	4	5	6
1	i	p	a	␣	␣	f
2	t	e	s	t	s	i
3	m	a	k	e	␣	n
4	i	t	.	␣	␣	h
5	r	a	v	b		

図2　完成した検査文字付表

(2)　検査文字付表の検証

(1)で作成した検査文字付表の，最下行を除く各行と最右列を除く各列を文字列とみなし，それぞれ関数 validateCheckCharacter で検証した結果，全て誤りがないと判定された場合には，検査文字付表に誤りがないと判定する。一つでも誤りがあると判定された場合は，検査文字付表に誤りがあると判定する。

eに関する解答群

ア　j　　イ　k　　ウ　l　　エ　m

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。

図2の1行目の検査文字付文字列を取り除いた，図3の検査文字付表について考える。表4のケース1～4の検査文字付文字列を順に，図3の1行目に格納して検証した場合，検査文字付表に誤りがないと判定されるケースは［ f ］。

行＼列	1	2	3	4	5	6
1						
2	t	e	s	t	s	i
3	m	a	k	e	␣	n
4	i	t	.	␣	␣	h
5	r	a	v	b		

注記　5行5列目（網掛け部分）には，設問3の正しい答えが入っているものとする。

図3　1行目を取り除いた検査文字付表

fに関する解答群

ア　1と2と3と4である　　イ　2である　　ウ　2と3である

エ　2と3と4である　　オ　2と4である　　カ　ない

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1～3に答えよ。

〔プログラムの説明〕

文字列の中から，回文（palindrome）を探して表示する関数 find_palindrome である。回文とは，先頭から読んだ文字の並びと末尾から読んだ文字の並びが一致する文字の並びのことである。ただし，ここでは次の条件を満たすものとする。

- 文字列は英字（A～Z，a～z），数字（0～9），記号（!"#%&'()*+,-./:;<=>?[]^_{|}）及び空白文字から成る。
- 文字の並びを読むとき，英字の大文字と小文字は区別しない。
- 文字の並びを読むとき，記号及び空白文字は無視する。
- 回文は英数字で始まり英数字で終わる。ただし，英数字1文字は回文ではない。

本問のプログラムが表示する回文の例を表1に示す。No. 1で示す文字列において，文字の並び bc0cb は回文である。c0c も回文であるが，本問のプログラムでは，文字列の先頭に最も近い文字から始まるものを表示する。また，No. 2で示す文字列において，英字の大文字と小文字は区別しないので，文字の並び Bc0Cb は回文である。さらに，No. 3で示す文字列において，英字の大文字と小文字を区別せず，かつ，記号及び空白文字を無視するので，文字の並び B!c0 Cb は回文である。No. 4で示す文字列において，文字列の先頭に最も近い b を先頭文字とする文字の並び bc0cb と bc0cb1bc0cb は，いずれも回文である。しかし，本問のプログラムでは，先頭文字位置が同じ回文が複数あれば，長さが最も短いものを表示する。

表1　本問のプログラムが表示する回文の例

No.	文字列	表示する回文
1	abc0cbe	bc0cb
2	ABc0CbE	Bc0Cb
3	AB!c0 CbE	B!c0 Cb
4	abc0cb1bc0cbe	bc0cb

(1) 関数 find_palindrome の仕様は，次のとおりである。ここで，引数の値に誤りはないものとする。

機能： 文字列 text の中から，回文を探して表示する。文字列 text の中に複数の回文がある場合，先頭文字位置が文字列の先頭に最も近いものを表示する。さらに，先頭文字位置が同じ回文が複数あれば，長さが最も短いものを表示する。

引数： text　文字列

関数 find_palindrome は，関数 is_palindrome 及び関数 find_char を使用する。

(2) 関数 is_palindrome の仕様は，次のとおりである。ここで，引数の値に誤りはないものとする。

機能： 文字の並び chars が回文かどうかを判定する。

引数： chars　文字の並び

size　文字の並び chars の長さ

返却値：回文の場合は 1

回文でない場合は 0

(3) 関数 find_char の仕様は，次のとおりである。ここで，引数の値に誤りはないものとする。

機能： 文字列 str 中で，文字 ch が最初に現れる位置を求める。ただし，英字の大文字と小文字は区別しない。

引数： str　文字列

ch　文字

返却値：文字 ch が現れる場合は，それが最初に現れる位置へのポインタ

文字 ch が現れない場合は，空ポインタ

(4) プログラム中で使用しているライブラリ関数の概要は，次のとおりである。

isalnum(ch)：文字 ch が英数字のとき，0 以外の値を返し，それ以外のとき，0 を返す。

tolower(ch)：文字 ch が英大文字のとき，その文字に対応する英小文字を返し，それ以外のとき，文字 ch を返す。

putchar(ch)：文字 ch を表示する。

〔プログラム〕

(行番号)

```
 1 #include <stdio.h>
 2 #include <ctype.h>

 3 void find_palindrome(const char*);
 4 int is_palindrome(const char*, int);
 5 const char* find_char(const char*, char);

 6 void find_palindrome(const char* text) {
 7     int i;
 8     int psize;        /* 文字の並びの長さ */
 9     const char* ith;  /* 文字列 text の第 i 文字へのポインタ */
10     const char* hit;  /* 第 i 文字と一致した文字へのポインタ */
11     for (i = 0; text[i] != '\0'; i++) {
12         if (!isalnum(text[i])) {
13             continue;
14         }
15         ith = &text[i];
16         hit = find_char(ith + 1, *ith);
17         while (hit != NULL) {
18             psize = [    a    ];
19             if (is_palindrome(ith, psize)) {
20                 while (ith < hit + 1) {
21                     putchar(*ith);
22                     ith++;
23                 }
24                 putchar('\n');
25                 return;
26             }
27             hit = find_char(hit + 1, *ith);
28         }
29     }
30 }
```

```
31 int is_palindrome(const char* chars, int size) {
32     int l;
33     int r;
34     for (l = 0, r = size - 1; l < r; l++, r--) {
35         while (!isalnum(chars[l])) {
36             l++;
37         }
38         while (!isalnum(chars[r])) {
39             r--;
40         }
41         if ( [ b ] ) {
42             return 0;
43         }
44     }
45     return 1;
46 }

47 const char* find_char(const char* str, char ch) {
48     int i;
49     for (i = 0; str[i] != '\0'; i++) {
50         if ( [ c ] ) {
51             return &str[i];
52         }
53     }
54     return NULL;
55 }
```

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　i　　イ　ith - &text[0]　　ウ　ith - &text[0] + 1

エ　hit - ith　　オ　hit - ith + 1

b に関する解答群

ア　l == r

イ　l != r

ウ　chars[l] == chars[r]

エ　chars[l] != chars[r]

オ　tolower(chars[l]) == tolower(chars[r])

カ　tolower(chars[l]) != tolower(chars[r])

c に関する解答群

ア　ch == str[i]

イ　ch != str[i]

ウ　tolower(ch) == tolower(str[i])

エ　tolower(ch) != tolower(str[i])

オ　(ch == tolower(str[i])) || (tolower(ch) == str[i])

カ　(ch == tolower(str[i])) && (tolower(ch) == str[i])

設問2　関数 find_palindrome が，先頭文字位置が同じ回文が複数ある場合，長さが最も長いものを表示するように，プログラムを変更する。表 2 中の［　　］に入れる正しい答えを，解答群の中から選べ。

(1) 関数 find_char を，関数 find_last_char と置き換える。関数 find_last_char の仕様は，次のとおりである。ここで，引数の値に誤りはないものとする。

機能：　文字列 str の先頭から文字数 count 以内の範囲で文字 ch が現れる最後の位置を求める。ただし，英字の大文字と小文字は区別しない。

引数：　str　　文字列

ch　　文字

count　文字数

返却値：文字 ch が現れる場合は，それが最後に現れる位置へのポインタ

文字 ch が現れない場合は，空ポインタ

(2) 新たに使用するライブラリ関数 strlen(text)は，文字列 text の長さ（ただし，終端ナル文字'\0'は含まない）を返す。

表2 プログラムの変更内容

処置	変更内容
行番号2と3の間に追加	#include <string.h>
行番号5を変更	const char* find_last_char(const char*, char, int);
行番号10と11の間に追加	unsigned int textlen = strlen(text);
行番号16を変更	hit = find_last_char(ith + 1, *ith, textlen - i - 1);
行番号27を変更	hit = find_last_char(ith + 1, *ith, [d]);

行番号47～49を変更	const char* find_last_char(const char* str, char ch, int count) {
	int i;
	for ([e] ; i--) {

dに関する解答群

ア textlen - psize　　イ textlen - i - psize

ウ textlen - i + psize　　エ psize + 2

オ psize - 2

eに関する解答群

ア i = count; i > 0　　イ i = count; i >= 0

ウ i = count - 1; i > 0　　エ i = count - 1; i >= 0

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。

設問2で変更した関数 find_palindrome を，次の文字列を引数として実行した。ここで“␣”は空白文字を表す。

No!␣Madam,␣I'm␣Adam␣Graham.␣This␣is␣my␣gym.

関数 find_palindrome が回文の表示を終了するまでに，関数 find_last_char は［ f ］回呼び出され，その最後の呼出しにおける引数 count の値は［ g ］である。

fに関する解答群

ア　2	イ　3	ウ　4
エ　5	オ　6	カ　7

gに関する解答群

ア　13	イ　20	ウ　31
エ　36	オ　38	

選択した問題は，選択欄の(選)をマークしてください。マークがない場合は，採点されません。

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。

〔プログラムの説明〕

ある駐車場では，自動精算システムを導入している。利用者は，入庫時に利用番号と入庫時刻が記録された駐車券を受け取り，出庫時に駐車券を精算機に投入して駐車料金を支払う。連続した複数日にまたがる駐車も可能である。プログラム1は入庫時に入庫時刻を精算ファイルに記録するサブプログラムであり，プログラム2は出庫時に駐車料金を計算して出庫時刻とともに精算ファイルに記録し，駐車料金の値を呼出し元に返すサブプログラムである。

(1)　精算ファイルは，図1に示すレコード様式の索引ファイルであり，プログラムの実行時に既に存在する。主キーは利用番号である。ファイルの排他制御は適切に行われるものとする。

利用番号 8桁	入庫時刻 12桁	出庫時刻 12桁	駐車料金 6桁

図1　精算ファイルのレコード様式

①　利用番号には，入庫時に発券される駐車券ごとに1ずつ増える8桁の数字が格納される。

②　入庫時刻及び出庫時刻には，西暦の年，月，日と，24時間表記の時（0～23），分（0～59）が，それぞれ4桁，2桁，2桁，2桁，2桁で格納される。

(2)　プログラム1のパラメタの様式は，図2のとおりである。呼出し元プログラムから利用番号と入庫時刻を受け取る。

利用番号 8桁	入庫時刻 12桁

図2　プログラム1のパラメタの様式

(3) プログラム 2 のパラメタの様式は，図 3 のとおりである。呼出し元プログラムから利用番号と出庫時刻を受け取り，駐車料金の値を返す。

図 3　プログラム 2 のパラメタの様式

(4) 駐車料金は，入庫時刻を基準に駐車時間を 1 時間単位に区切って算出する。この単位を課金単位と呼び，1 課金単位当たり 300 円を課金する。各課金単位の開始時刻の“分”は，入庫時刻と同じ“分”とする。各課金単位の開始時刻から 1 分以上が経過すると課金対象になる。例えば，10 時 30 分に入庫して，11 時 30 分に出庫した場合，課金単位は 1 で，駐車料金は 300 円である。10 時 30 分に入庫して，11 時 31 分に出庫した場合，2 時間目の課金単位の開始時刻である 11 時 30 分から 1 分が経過しているので，課金単位は 2 となり，駐車料金は 600 円になる。駐車料金が 999,999 円を超えることはない。

(5) プログラム 2 の中で使用している組込み関数 INTEGER-OF-DATE は，引数に指定された日付を西暦 1601 年 1 月 1 日からの通算の日数に変換して返す。

〔プログラム 1〕

（行番号）

```
 1  DATA DIVISION.
 2  FILE SECTION.
 3  FD ACCT-FILE.
 4  01 ACCT-REC.
 5     02 ACCT-NO        PIC X(8).
 6     02 ACCT-IN        PIC 9(12).
 7     02 ACCT-OUT       PIC 9(12).
 8     02 ACCT-FEE       PIC 9(6).
 9  LINKAGE SECTION.
10  01 PRM.
11     02 PRM-NO         PIC X(8).
12     02 PRM-IN         PIC 9(12).
13  PROCEDURE DIVISION USING PRM.
14  MAIN-PROC.
15      OPEN EXTEND ACCT-FILE.
16      INITIALIZE ACCT-REC.
17      MOVE PRM-NO TO ACCT-NO.
```

```
18      MOVE PRM-IN TO ACCT-IN.
19      WRITE ACCT-REC.
20      CLOSE ACCT-FILE.
21      EXIT PROGRAM.
```

〔プログラム 2〕

（行番号）

```
 1  DATA DIVISION.
 2  FILE SECTION.
 3  FD ACCT-FILE.
 4  01 ACCT-REC.
 5     02 ACCT-NO        PIC X(8).
 6     02 ACCT-IN.
 7        03 ACCT-I-DATE    PIC 9(8).
 8        03 ACCT-I-TIME.
 9           04 ACCT-I-HH      PIC 9(2).
10           04 ACCT-I-MM      PIC 9(2).
11     02 ACCT-OUT.
12        03 ACCT-O-DATE    PIC 9(8).
13        03 ACCT-O-TIME.
14           04 ACCT-O-HH      PIC 9(2).
15           04 ACCT-O-MM      PIC 9(2).
16     02 ACCT-FEE       PIC 9(6).
17  WORKING-STORAGE SECTION.
18  77 W-HRS          PIC 9(4).
19  LINKAGE SECTION.
20  01 PRM1.
21     02 PRM-NO         PIC X(8).
22     02 PRM-OUT        PIC 9(12).
23  01 PRM2.
24     02 PRM-FEE        PIC 9(6).
25  PROCEDURE DIVISION USING PRM1 PRM2.
26  MAIN-PROC.
27      OPEN I-O ACCT-FILE.
28      MOVE PRM-NO TO ACCT-NO.
29      READ ACCT-FILE.
30      MOVE PRM-OUT TO ACCT-OUT.
31      PERFORM COMP-PROC.
32      [          a          ].
33      REWRITE ACCT-REC.
34      CLOSE ACCT-FILE.
35      EXIT PROGRAM.
```

```
36  COMP-PROC.
37      COMPUTE W-HRS =
38          (FUNCTION INTEGER-OF-DATE(ACCT-O-DATE) * 24 + ACCT-O-HH)  [ b ]
39          (FUNCTION INTEGER-OF-DATE(ACCT-I-DATE) * 24 + ACCT-I-HH).
40      IF ACCT-I-MM < ACCT-O-MM THEN
41          [            c            ]
42      END-IF.
43      COMPUTE ACCT-FEE = W-HRS * 300.
```

設問1　プログラム1における精算ファイルの呼出し法として正しい答えを，解答群の中から選べ。ここで，括弧内は，呼出し法に対応する環境部のファイル管理記述項における ACCESS MODE 句の書き方である。

解答群

ア　順呼出し　(ACCESS MODE IS SEQUENTIAL)

イ　動的呼出し　(ACCESS MODE IS DYNAMIC)

ウ　乱呼出し　(ACCESS MODE IS RANDOM)

設問2　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。

aに関する解答群

ア　ADD PRM-FEE TO ACCT-FEE　　イ　MOVE ACCT-FEE TO PRM-FEE

ウ　MOVE PRM-FEE TO ACCT-FEE　　エ　MOVE PRM-NO TO ACCT-NO

bに関する解答群

ア　*　　イ　+　　ウ　-　　エ　/

cに関する解答群

ア　ADD 1 TO W-HRS　　イ　ADD ACCT-O-HH TO W-HRS

ウ　SUBTRACT 1 FROM W-HRS　　エ　SUBTRACT ACCT-I-HH FROM W-HRS

設問３　夜間時間帯の料金割引を実施するために，プログラム２を変更する。夜間時間帯は 20 時 00 分から翌朝 5 時 59 分までとし，各課金単位の開始時刻が夜間時間帯に含まれるときは，1 課金単位当たりの課金を 100 円とする。表 1 中の［　　］に入れる正しい答えを，解答群の中から選べ。

表１　プログラム２の変更内容

処置	変更内容
行番号 18 と 19 の間に追加	77 W-CNT　　PIC 9(4). 77 W-IDX　　PIC 9(2). 01 W-ACCT-VAL　　PIC X(72) VALUE "100100100100100300300300300300300 -　"300300300300300300300100100100100". 01 W-ACCT-TBL　　REDEFINES W-ACCT-VAL. 02 W-ACCT　　PIC 9(3) OCCURS 24.
行番号 43 を変更	[　　d　　]. PERFORM TEST BEFORE VARYING W-CNT FROM 1 BY 1 UNTIL [　　e　　] ADD W-ACCT(W-IDX) TO ACCT-FEE ADD 1 TO W-IDX IF [　　f　　] THEN MOVE 1 TO W-IDX END-IF END-PERFORM.

注記　変更内容の各行とも，空白でない最終文字の直後で改行している。行番号 18 と 19 の間に追加する変更内容の１桁目は標識領域である。

d に関する解答群

ア　COMPUTE W-IDX = ACCT-I-HH + 1　　イ　COMPUTE W-IDX = ACCT-I-HH - 1

ウ　MOVE 1 TO W-IDX　　エ　MOVE ACCT-I-HH TO W-IDX

e，f に関する解答群

ア　W-CNT = W-HRS　　イ　W-CNT > W-HRS

ウ　W-HRS = W-IDX　　エ　W-IDX = 24

オ　W-IDX = W-CNT　　カ　W-IDX > 24

選択した問題は，選択欄の(選)をマークしてください。マークがない場合は，採点されません。

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1～3に答えよ。

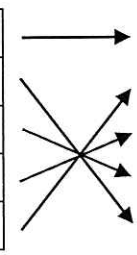
（Javaプログラムで使用するAPIの説明は，この冊子の末尾を参照してください。）

〔プログラムの説明〕

文字列の配列を要素にもつ配列で表現された表中の行を，並べ替えるプログラムである。表は行の配列として，行は文字列の配列として構成される。各行の指定した列が，指定した順序に並ぶように，行を並べ替える。

図1は，表の最左列の文字列を辞書順に，最左列の文字列が等しい行は右隣の列の文字列を数値とみなしたときに降順になるように並べ替えた例である。

apple	3	1,000
cherry	1	1,000
banana	1	300
banana	2	2,000
apple	2	300



apple	3	1,000
apple	2	300
banana	2	2,000
banana	1	300
cherry	1	1,000

図1　並べ替えの例

クラスTableSorterは，表中の行を並べ替える機能を提供する。次のメソッドと入れ子クラスOrderByをもつ。

(1) putSortOrder(String key, Comparator<String> order) ― このクラスのメソッドsortで指定可能な順序を登録する。

key ― 表を並べ替える順序を示すために使用する文字列（以下，順序指定子という）。

order ― インタフェースComparator型のオブジェクトで，keyで指定した順序指定子が示す順序を規定する。

(2) sort(String[][] table, final OrderBy... orderBys) ― 条件として指定した列が指定した順序になるように，表中の行を並べ替える。条件は複数指定できる。

並び順の決定には先に指定した条件が優先される。先に指定した条件で並び順が決まらない行については，後に指定した条件を加えて決定する。指定した全ての条件で順序が決まらない行については，並べ替える前の並び順を維持する。

table — 整列対象の表を格納した String 型の配列を要素にもつ配列を指定する。

orderBys — 条件を必要なだけ指定する。

クラス TableSorter の入れ子クラス OrderBy は並べ替えの条件を表すクラスである。フィールド key は順序指定子，フィールド col は比較対象の列の位置を示す数（行を表す String 型配列の添字）である。フィールド isReversed が false であれば，この条件が表す順序が順序指定子 key で指定されるオブジェクトのメソッド compare が返す大小関係での昇順で並べ替えることを意味し，true であれば，降順で並べ替えることを意味する。

クラス TableSorterTester はテスト用のプログラムである。

実行結果を図 2 に示す。

```
apple 3 1,000
apple 2 300
banana 2 2,000
banana 1 300
cherry 1 1,000
```

図 2　実行結果

〔プログラム1〕

```
import java.util.Arrays;
import java.util.Comparator;
import java.util.HashMap;
import java.util.Map;

public class TableSorter {
   private Map<  a  > orderMap = new HashMap<  a  >();

   public void putSortOrder(String key, Comparator<String> order) {
      orderMap.put(key, order);
   }

   public void sort(String[][] table, final OrderBy... orderBys) {
      Arrays.sort(table, new Comparator<String[]>() {
            public int compare(String[] s1, String[] s2) {
               for (  b  ) {
                  int order = orderMap.get(orderBy.key).
                     compare(s1[  c  ], s2[  c  ]);
                  if (order  d  0) {
                     return  e  ;
                  }
               }
               return 0;
            }
         });
   }

   public static class OrderBy {
      final String key;
      final int col;
      final boolean isReversed;

      public OrderBy(String key, int col) {
         this(key, col, false);
      }

      public OrderBy(String key, int col, boolean isReversed) {
         this.key = key;
         this.col = col;
         this.isReversed = isReversed;
```

```
        }
    }
}
```

〔プログラム 2〕

```
import java.util.Comparator;

public class TableSorterTester {
    public static void main(String... args) {
        String[][] data = new String[][] {
            {"apple", "3", "1,000"},
            {"cherry", "1", "1,000"},
            {"banana", "1", "300"},
            {"banana", "2", "2,000"},
            {"apple", "2", "300"},
        };
        TableSorter sorter = new TableSorter();
        sorter.putSortOrder("lex", new Comparator<String>() {
              public int compare(String o1, String o2) {
                 return o1.compareTo(o2);
              }
           });
        sorter.putSortOrder("num", new Comparator<String>() {
              public int compare(String o1, String o2) {
                 return new Integer(o1).compareTo(new Integer(o2));
              }
           });
        sorter.sort(data, new TableSorter.OrderBy("lex", 0),
                    new TableSorter.OrderBy("num", 1, true));
        for (String[] row : data) {
            for (String col : row) {
                System.out.printf("%s ", col);
            }
            System.out.println();
        }
    }
}
```

α

設問1　プログラム1中の［　　］に入れる正しい答えを，解答群の中から選べ。

aに関する解答群

ア　Character, Comparator<String>　　イ　Character, String

ウ　String, Comparator<String>　　エ　String, String

bに関する解答群

ア　int i = 0; i < orderBys.length; i++

イ　int i = 0; i < table.length; i++

ウ　OrderBy orderBy : orderBys

エ　String[] row : table

cに関する解答群

ア　.get(col)　　イ　.get(orderBy.col)

ウ　[col]　　エ　[orderBy.col]

dに関する解答群

ア　!=　　イ　<　　ウ　==　　エ　>

eに関する解答群

ア　-order

イ　order

ウ　orderBy.isReversed ? -order : order

エ　orderBy.isReversed ? order : -order

設問2　プログラム2のαで示した2行を次の2行と入れ替えて実行したとき，実行結果の1行目に出力される内容として正しい答えを，解答群の中から選べ。

```
sorter.sort(data, new TableSorter.OrderBy("lex", 2),
            new TableSorter.OrderBy("lex", 0));
```

解答群

ア　apple 2 300　　イ　apple 3 1,000　　ウ　banana 1 300

エ　banana 2 2,000　　オ　cherry 1 1,000

設問3　プログラム2のαで示した2行を次の8行と入れ替えて実行したとき，実行結果の3行目に出力される内容として正しい答えを，解答群の中から選べ。

```
sorter.putSortOrder("numC", new Comparator<String>() {
      public int compare(String s1, String s2) {
         return new Integer(s1.replace(",", "")).
            compareTo(new Integer(s2.replace(",", "")));
      }
   });
sorter.sort(data, new TableSorter.OrderBy("numC", 2),
            new TableSorter.OrderBy("lex", 0));
```

解答群

ア　apple 2 300　　イ　apple 3 1,000　　ウ　banana 1 300

エ　banana 2 2,000　　オ　cherry 1 1,000

選択した問題は，選択欄の(選)をマークしてください。マークがない場合は，採点されません。

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1～4に答えよ。

〔プログラム1の説明〕

連続する2語から成るビット列 α の中から，別のビット列 β と一致する部分ビット列を検索する副プログラム BSRH である。部分ビット列の検索の概要を，図1に示す。

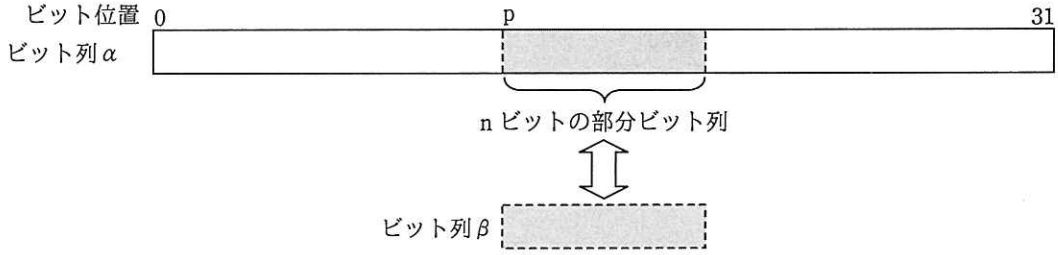


図1　部分ビット列の検索の概要

(1) 主プログラムは，α が格納されている領域の先頭アドレスを GR1 に，β を GR2 に，β の長さ n（1≦n≦16）を GR3 に設定して副プログラム BSRH を呼ぶ。β は GR2 に左詰めで設定し，残りのビットには0を設定する。

(2) 副プログラム BSRH は，α の先頭（ビット位置0）から β と照合し，一致する部分ビット列がある場合は最初に一致する部分ビット列の先頭のビット位置 p を，一致する部分ビット列がない場合は −1 を，GR0 に設定して主プログラムに戻る。

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

〔プログラム1〕

（行番号）

```
1 BSRH    START
2         RPUSH
3         LD     GR0,=-1
4         LD     GR6,=#FFFF      ; マスク作成
```

```
 5          SRL    GR6,0,GR3
 6          XOR    GR6,=#FFFF
 7          LD     GR4,0,GR1       ; αの取出し
 8          LD     GR5,1,GR1
 9          LD     GR1,=32
10          SUBA   GR1,GR3         ; GR1 ← 32 - n
11          LD     GR3,GR1         ; GR3 ← 32 - n
12 LP       LD     GR7,GR4
13          AND    GR7,GR6
14          [      a       ]
15          JZE    FOUND           ; 一致あり
16          SUBA   GR1,=1
17          JMI    FIN             ; 一致なし
18          SLL    GR4,1           ; αを1ビット左シフト
19          SLL    GR5,1
20          [      b       ]
21          JUMP   LP
22 NEXT     OR     GR4,=#0001
23          JUMP   LP
24 FOUND    SUBA   GR3,GR1         ; pの算出
25          LD     GR0,GR3
26 FIN      RPOP
27          RET
28          END
```

設問1　プログラム1中の［　　］に入れる正しい答えを，解答群の中から選べ。

aに関する解答群

ア	AND	GR7,GR2	イ	OR	GR7,GR2	ウ	SLL	GR7,0,GR2
エ	SRL	GR7,0,GR2	オ	XOR	GR7,GR2			

bに関する解答群

ア	JMI	FOUND	イ	JMI	NEXT	ウ	JOV	FOUND
エ	JOV	NEXT	オ	JPL	FOUND	カ	JPL	NEXT

設問2　プログラム1の行番号4～6を，次の2命令で置き換えても，GR6には同じ値が得られる。［　　］に入れる正しい答えを，解答群の中から選べ。

```
LD     GR6,=#8000
[                  ]
```

解答群

ア	SLL	GR6,-1,GR3	イ	SLL	GR6,0,GR3	ウ	SRA	GR6,-1,GR3
エ	SRA	GR6,0,GR3	オ	SRL	GR6,-1,GR3	カ	SRL	GR6,0,GR3

設問３　α の先頭から β と照合し，最初に一致する部分ビット列を，β と同じ長さの別のビット列 γ で置き換える副プログラム BREP を，副プログラム BSRH を使用して作成した。γ は GR4 に左詰めで設定し，残りのビットには 0 を設定する。それ以外の BREP の呼出しに関する仕様は，プログラム 1 の説明中の BSRH を BREP に読み替えて適用する。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。

〔プログラム 2〕

（行番号）

```
 1 BREP    START
 2         RPUSH
 3         CALL  BSRH
 4         LD    GR2,GR0
 5         JMI   FIN
 6         LD    GR6,=#FFFF      ; マスク作成
 7         SRL   GR6,0,GR3
 8         XOR   GR6,=#FFFF
 9         LD    GR7,GR3         ; GR7 ← n
10         LD    GR3,=16
11         SUBA  GR3,GR2         ; GR3 ← 16 - p
12         JMI   ONL2            ; 一致する部分ビット列が 2 語目だけのとき
13         JZE   ONL2
14         CPA   GR3,GR7         ; (16 - p)と n の比較
15         JMI   NEXT            ; 2 語にまたがる処理
16         JUMP  ONL1            ; 一致する部分ビット列が 1 語目だけのとき
17 NEXT    LD    GR5,GR4         ; γとマスクを退避
18         LD    GR7,GR6
19         CALL  S1              ; 1 語目の処理
20         LD    GR4,GR5
21         LD    GR6,GR7
22         SLL   GR4,0,GR3       ; 2 語目用γの調整
23         SLL   GR6,0,GR3       ; 2 語目用マスクの調整
24         LAD   GR1,1,GR1
25         CALL  S2              ; 2 語目の最終処理
26         JUMP  FIN
```

```
27 ONL1    CALL  S1
28         JUMP  FIN
29 ONL2    [      c      ]
30         SUBA  GR2,GR3        ; GR2 ← p - 16
31         LAD   GR1,1,GR1      ; 操作対象を2語目にして,
32         CALL  S1             ; 2語目の処理
33 FIN     RPOP
34         RET
35 S1      SRL   GR4,0,GR2      ; γの調整
36         SRL   GR6,0,GR2      ; マスクの調整
37 S2      LD    GR2,0,GR1      ; 操作対象語の取出し
38         XOR   GR6,=#FFFF
39         AND   GR2,GR6
40         [      d      ]
41         ST    GR2,0,GR1
42         RET
43         END
```

cに関する解答群

ア	ADDL	GR2,GR7	イ	LD	GR2,=0	ウ	LD	GR2,=16
エ	LD	GR2,=32	オ	SUBA	GR2,GR7			

dに関する解答群

ア	AND	GR2,GR4	イ	OR	GR2,GR4	ウ	SLL	GR2,0,GR4
エ	SRL	GR2,0,GR4	オ	XOR	GR2,=#FFFF			

設問 4　次の設定でプログラム 2 を実行したとき，行番号 23 の命令実行直後の GR6 に格納されている値として正しい答えを，解答群の中から選べ。

α：1語目　#FFF3
　　2語目　#7FFF
β：$(11011)_2$
n：5
γ：$(11110)_2$

解答群

ア	#0003	イ	#0007	ウ	#8000
エ	#C000	オ	#E000	カ	#F000

選択した問題は，選択欄の(選)をマークしてください。マークがない場合は，採点されません。

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 1～3 に答えよ。

〔表計算の説明〕

S 社では，社内の業務用 PC（以下，PC という）からインターネットへの接続に，社内に設置した Proxy サーバ（以下，サーバという）を利用している。サーバの運用担当である T さんは，社内から社外の Web ページへのアクセス状況を確認するために，サーバのアクセスログを分析することにした。

〔ワークシート：ログ〕

ワークシート“ログ”には，2017 年 4 月 3 日（月）から 2017 年 4 月 30 日（日）までの 4 週間に社外の Web ページにアクセスした記録だけをアクセスログから取り出し，集計に必要な加工を行った結果が入力されている。ワークシート“ログ”の例を，図 1 に示す。

	A	B	C	D	E	F
1	時刻	IP アドレス	URL	時間帯	曜日コード	集計キー
2	24852960	192.168.20.112	http://○○○.com/index.html	0	4	04
3	24852980	192.168.50.30	http://www.△△△.co.jp/top.html	0	4	04
⋮	⋮	⋮	⋮	⋮	⋮	⋮
9058	24893254	192.168.10.103	http://□□□.co.jp/index.html	23	3	233
9059	24893274	192.168.20.107	http://◇◇◇.co.jp/index.html	23	3	233
9060						

注記　“○○○”，“△△△”，“□□□”，“◇◇◇”は，特定の文字列を表す。

図 1　ワークシート“ログ”の例

(1)　行 1 は見出し行で，データは行 2 以降に入力されている。

(2)　列 A には，PC がサーバにアクセスした時刻が入力されている。ここで，時刻は，基準となる 1970 年 1 月 1 日（木曜日）の 0 時 00 分から当該時刻までの分を単位と

する経過時間で表現する。

(3) 列Bには，社外のWebページにアクセスしたPCのIPアドレスが入力されている。

(4) 列Cには，アクセスしたWebページのURLが入力されている。

(5) 列Dには，列Aの時刻に対応する時間帯（0～23）を表示する式が入力されている。ここで，時間帯は，0時台（0時00分～0時59分）を0，1時台（1時00分～1時59分）を1，…，23時台（23時00分～23時59分）を23とする整数値である。

(6) 列Eには，列Aの時刻に対応する曜日を0～6の整数値で表現する曜日コードを表示する式が入力されている。ここで，曜日コードは，基準となる1970年1月1日の曜日である木曜日を0とし，水曜日の6までを順に割り振る。

(7) 列Fには，時間帯と曜日コードをつないで一つの文字列とした集計キーを表示する式が入力されている。

(8) データは最大9,998件までとし，データの最終行よりも下の行の列A～Cの各セルには空値が入力されている。

〔ワークシート：曜日〕

曜日と曜日コードとの対応を，ワークシート“曜日”に入力する。ワークシート“曜日”を，図2に示す。

	A	B	C	D	E	F	G	H
1	曜日	木	金	土	日	月	火	水
2	曜日コード	0	1	2	3	4	5	6

図2 ワークシート“曜日”

設問1　ワークシート“ログ”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，セル D2～F9999 に入力する各式は，当該行の列 A のセルの値が空値であれば空値を返す。

(1) セル D2 には，セル A2 の時刻に対応する時間帯を表示する次の式を入力する。

［　a　］

(2) セル E2 には，セル A2 の時刻に対応する曜日コードを表示する次の式を入力する。ここで，1440 は，1 日を分に換算した値（24×60）である。

IF(A2＝null, null, 剰余(切捨て(A2／1440, 0), 7))

(3) セル F2 には，セル D2 とセル E2 の値をつないで，一つの文字列として返す次の式を入力する。

IF(A2＝null, null, 結合(D2, E2))

(4) セル D2，E2，F2 に入力した式を，行 3～9999 の対応する列のセルに複写する。

a に関する解答群

ア　IF(A2＝null, null, 切上げ(剰余(A2／24, 60), 0))

イ　IF(A2＝null, null, 切上げ(剰余(A2, 60)／1440, 0))

ウ　IF(A2＝null, null, 切捨て(剰余(A2／24, 60), 0))

エ　IF(A2＝null, null, 切捨て(剰余(A2, 60)／1440, 0))

オ　IF(A2＝null, null, 剰余(切捨て(A2／1440, 0), 60))

カ　IF(A2＝null, null, 剰余(切捨て(A2／60, 0), 24))

設問2　ワークシート“集計”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。

〔ワークシート：集計〕

ワークシート“集計”には，“アクセス集計表”及び“アクセス分析表”を作成する。ワークシート“集計”の例を，図 3 に示す。

	A	B	C	D	E	F	…	N	O	…	Y	Z
1	アクセス集計表											
2	曜日＼時間帯	0	1	2	3	4	…	12	13	…	23	計
3	月	13	9	6	5	6	…	163	183	…	18	1645
4	火	15	11	6	5	1	…	125	157	…	9	1376
5	水	16	6	5	3	3	…	104	143	…	10	1222
6	木	18	10	7	6	8	…	161	203	…	18	1820
7	金	15	7	8	8	1	…	133	167	…	16	1482
8	土	13	6	4	1	4	…	92	96	…	9	945
9	日	5	4	1	3	3	…	50	68	…	6	568
10	計	95	53	37	31	26	…	828	1017	…	86	9058
11												
12	アクセス分析表											
13	(1)時間帯分析											
14	・最多となる時間帯			13	時台							
15	・最少となる時間帯			4	時台							
16	(2)曜日分析											
17	・最多となる曜日			木	曜日							
18	・最少となる曜日			日	曜日							

図3　ワークシート“集計”の例

(1)　“アクセス集計表”は，ワークシート“ログ”を参照して，社外の Web ページにアクセスした回数（以下，アクセス回数という）を，該当する曜日，時間帯ごとに集計して表示する。

(2)　“アクセス分析表”は，アクセス回数が，最多となる時間帯と最少となる時間帯，及び最多となる曜日と最少となる曜日を表示する。

(3)　セル A3～A9 には，曜日として“月”～“日”を入力する。

(4)　セル B2～Y2 には，時間帯として 0～23 を入力する。

(5)　セル B3 には，ワークシート“ログ”のデータの中から，曜日と時間帯がそれぞれ，当該セルと同じ行の列 A で示す曜日と，同じ列の行 2 で示す時間帯に一致するデータの個数を数えて表示する次の式を入力して，セル B3～Y9 に複写する。

条件付個数(ログ!$F$2:$F$9999, [　　b　　])

(6)　セル Z3 には，当該セルと同じ行の列 B～Y の値を合計する式を入力し，セ

ル Z4～Z9 に複写する。

(7) セル B10 には，当該セルと同じ列の行 3～9 の値を合計する式を入力し，セル C10～Z10 に複写する。

(8) セル D14 には，アクセス回数を時間帯ごとに集計したときに，その値が最多となる時間帯を表示する式を入力し，セル D15 にはその値が最少となる時間帯を表示する式を入力する。ここで，最多，最少となる時間帯が複数あったときは，そのうちアクセス集計表の左端に最も近い時間帯を表示する。

(9) セル D17 には，アクセス回数を曜日ごとに集計したときに，その値が最多となる曜日を，“月”～“日”で表示する次の式を入力する。

c

セル D18 には，アクセス回数を曜日ごとに集計したときに，その値が最少となる曜日を，“月”～“日”で表示する式を入力する。

ここで，最多，最少となる曜日が複数あったときは，そのうちアクセス集計表の上端に最も近い曜日を表示する。

b に関する解答群

ア ＝結合(B$2, 照合一致($A3, 曜日!$B$1:$H$1, 0))

イ ＝結合(B$2, 照合検索($A3, 曜日!$B$2:$H$2, 曜日!$B$1:$H$1))

ウ ＝結合(B$2, 水平照合($A3, 曜日!$B$1:$H$2, 2, 0))

エ ＝結合(照合一致($A3, 曜日!$B$1:$H$1, 0), B$2)

オ ＝結合(照合検索($A3, 曜日!$B$2:$H$2, 曜日!$B$1:$H$1), B$2)

カ ＝結合(水平照合($A3, 曜日!$B$1:$H$2, 2, 0), B$2)

c に関する解答群

ア 照合検索(最大(B10:Y10), B10:Y10, B2:Y2)

イ 照合検索(最大(Z3:Z9), Z3:Z9, A3:A9)

ウ 照合検索(照合一致(最大(Z3:Z9), Z3:Z9, 0)－1, 曜日!B2:H2, 曜日!B1:H1)

エ 表引き(A3:Z9, 1, 照合一致(最大(Z3:Z9), Z3:Z9, 0))

オ 表引き(A3:Z9, 照合一致(最大(Z3:Z9), Z3:Z9, 0),
照合一致(最大(B10:Y10), B10:Y10, 0))

カ 表引き(曜日!B1:H1, 1, 照合一致(最大(Z3:Z9), Z3:Z9, 0))

設問3　T さんは，ワークシート“ログ”のデータから，アクセス先の URL ごとのアクセス回数を求めるマクロ Page_count を作成し，ワークシート“アクセス先集計”に格納した。マクロ Page_count 中の［　　］に入れる正しい答えを，解答群の中から選べ。

〔ワークシート：アクセス先集計〕

マクロ実行後のワークシート“アクセス先集計”の例を，図4に示す。

	A	B
1	URL	アクセス回数
2	http://○○○.com/index.html	383
3	http://www.△△△.co.jp/top.html	782
4	http://□□□.co.jp/index.html	675
5	http://◇◇◇.co.jp/index.html	119
⋮	⋮	⋮

注記　“○○○”，“△△△”，“□□□”，“◇◇◇”は，特定の文字列を表す。

図4　ワークシート“アクセス先集計”の例

(1)　行1は見出し行である。

(2)　マクロ実行前のセル A2～B9999 には，あらかじめ空値を入力しておく。

(3)　マクロ Page_count の実行結果は，行2以降に表示される。

〔マクロ：Page_count の説明〕

(1)　ワークシート“ログ”のセル C2 を最初の対象セルとして，以降，セル C3，C4，…と順次，対象セルを下に移しながら，対象セルの値が空値になるまで，(2)の処理を繰り返す。

(2)　ワークシート“ログ”の対象セルに入力されている URL と同じ値が入力されているかどうかを，ワークシート“アクセス先集計”のセル A2 から下に検索し，同じ URL を値としてもつセルが見つかったときは，当該行の列 B のアクセス回数に 1 を加える。列 A のセルの値が空値になるまで検索しても同じ URL を値としてもつセルが見つからなかったときは，最初に現れた空値の行の，列 A に対象セルの値を，列 B に 1 を格納する。

〔マクロ：Page_count〕

```
○マクロ: Page_count
○数値型: log_line, count_line
○文字列型: log_url, count_url
・log_line ← 1
■ 相対(ログ!A1, log_line, 2) ≠ null
  ・count_line ← 1
  ・log_url ← 相対(ログ!A1, log_line, 2)
  ・count_url ← 相対(A1, count_line, 0)
  ■ [ d ]
    ・count_line ← count_line + 1
    ・count_url ← 相対(A1, count_line, 0)
  ▲ [ e ]
    ・相対(A1, count_line, 0) ← log_url
    ・相対(A1, count_line, 1) ← 1
  ▼ ・[ f ]
  ・log_line ← log_line + 1
■
```

dに関する解答群

ア　論理積(count_url = null, count_url = log_url)

イ　論理積(count_url = null, count_url ≠ log_url)

ウ　論理積(count_url ≠ null, count_url ≠ log_url)

エ　論理和(count_url = null, count_url = log_url)

オ　論理和(count_url = null, count_url ≠ log_url)

カ　論理和(count_url ≠ null, count_url ≠ log_url)

eに関する解答群

ア　count_url = log_url

イ　count_url = null

ウ　count_url ≠ null

エ　log_url = null

オ　log_url ≠ null

fに関する解答群

ア　相対(A1, count_line, 1) ← 相対(A1, count_line, 1) ＋ 1

イ　相対(A1, count_line, 1) ← 相対(ログ!A1, log_line, 2)

ウ　相対(A1, count_line, 2) ← 相対(A1, count_line, 2) ＋ 1

エ　相対(A1, count_line, 2) ← 相対(ログ!A1, log_line, 2)

オ　相対(A1, log_line, 1) ← 相対(A1, log_line, 1) ＋ 1

カ　相対(A1, log_line, 2) ← 相対(A1, log_line, 2) ＋ 1

■ Java プログラムで使用する API の説明

java.lang

public final class String

文字列を表す。インタフェース CharSequence を実装する。

メソッド

public int compareTo(String anotherString)

二つの文字列を辞書順に比較する。

引数： anotherString — 比較対象の文字列

戻り値：この文字列が引数文字列に等しいときは 0

この文字列が引数文字列より辞書順で小さいときは負の値

この文字列が引数文字列より辞書順で大きいときは正の値

---

public String replace(CharSequence target, CharSequence replacement)

この文字列中の target で指定された文字の並びに一致する部分文字列を，この文字列の先頭から検索し，順次，replacement で指定された文字の並びに置き換える。

引数： target — 置き換え対象の文字の並び

replacement — 置き換える文字の並び

戻り値：置き換え後の文字列

---

java.lang

public class Integer

プリミティブ型の int の値をオブジェクトにラップする。

コンストラクタ

public Integer(String s)

文字列で表現された符号付き 10 進整数で示される int 値を表す Integer オブジェクトを構築する。

引数： s — 符号付き 10 進整数の文字列表現

メソッド

public int compareTo(Integer anotherInteger)

二つの Integer オブジェクトを数値的に比較する。

引数： anotherInteger — 比較対象の Integer オブジェクト

戻り値：この Integer オブジェクトが anotherInteger に等しいときは 0

この Integer オブジェクトが anotherInteger より小さいときは負の値

この Integer オブジェクトが anotherInteger より大きいときは正の値

java.util

public class Arrays

配列を操作するための様々なメソッドを提供する。

メソッド

`public static <T> void sort(T[] a, Comparator<? super T> c)`

指定された配列を，指定されたコンパレータのメソッド compare が返す大小関係に従って，昇順に並べ替える。大きさが等しい要素の順序は，並べ替える前の順序を維持する。

引数： a — 並べ替え対象の配列
c — 配列の要素の順序を決定するコンパレータ

java.util

public interface Comparator<T>

二つのオブジェクトの大小関係を判定するインタフェースを提供する。

メソッド

`public int compare(T o1, T o2)`

引数で与えられた型 T の二つのオブジェクトを比較し，大小関係を整数値で返す。

引数： o1 — 1 番目のオブジェクト
o2 — 2 番目のオブジェクト

戻り値：o1 と o2 の大きさが等しいときは 0
o1 が o2 より小さいときは負の値
o1 が o2 より大きいときは正の値

java.util

public interface Map<K, V>

型 K のキーに型 V の値を対応付けて保持するインタフェースを提供する。各キーは，一つの値としか対応付けられない。

メソッド

`public V get(Object key)`

指定されたキーに対応付けられた値を返す。

引数： key — キー

戻り値：指定されたキーに対応付けられた型 V の値
このキーと値の対応付けがなければ null

`public V put(K key, V value)`

指定されたキーに指定された値を対応付けて登録する。このキーが既に他の値と対応付けられていれば，その値は指定された値で置き換えられる。

引数： key — キー
value — 値

戻り値：指定されたキーに対応付けられていた型 V の古い値
このキーと値の対応付けがなければ null

java.util

public class HashMap<K, V>

インタフェース Map のハッシュを用いた実装である。

コンストラクタ

public HashMap()

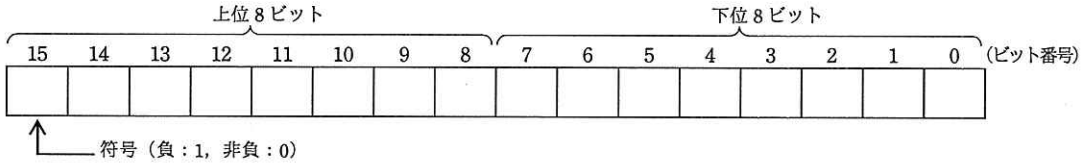
空の HashMap を作る。

■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。	

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

1.3 文字の符号表

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

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

(3) この表にない文字とそのビット構成が必要な場合は，問題中で与える。

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

2. アセンブラ言語 CASLⅡの仕様

2.1 言語の仕様

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

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

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

2.2 命令の種類

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

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

2.3 アセンブラ命令

アセンブラ命令は，アセンブラの制御などを行う。

(1)

START	[実行開始番地]

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

(2) | END | |

END 命令は，プログラムの終わりを定義する。

(3) | DS | 語数 |

DS 命令は，指定した語数の領域を確保する。

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

(4) | DC | 定数 [,定数] … |

DC 命令は，定数で指定したデータを（連続する）語に格納する。

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

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

2.4 マクロ命令

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

(1) | IN | 入力領域,入力文字長領域 |

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

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

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

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

(2) | OUT | 出力領域,出力文字長領域 |

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

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

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

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

(3)

RPUSH	

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

(4)

RPOP	

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

2.5 機械語命令

機械語命令のオペランドは，次の形式で記述する。

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

2.6 その他

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

(2) 生成された命令語，領域は，主記憶上で連続した領域を占める。

3. プログラム実行の手引

3.1 OS

プログラムの実行に関して，次の取決めがある。

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

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

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

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

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

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

3.2 未定義事項

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

表計算ソフトの機能・用語（基本情報技術者試験用）

表計算ソフトの機能，用語などは，原則として次による。

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

1. ワークシート

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

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

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

2. セルとセル範囲

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

［例］列A行1にあるセルのセル番地は，A1と表す。

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

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

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

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

3. 値と式

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

(2) 文字列は一重引用符“’”で囲って表す。

［例］文字列“A”，“BC”は，それぞれ'A'，'BC'と表す。

(3) 論理値の真をtrue，偽をfalseと表す。

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

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

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

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

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

4. 演算子

(1) 単項演算子は，正符号“＋”及び負符号 “－”とする。

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

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

(4) 括弧は丸括弧“(”及び“)”を使う。

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

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

5. セルの複写

(1) セルの値又は式を，他のセルに複写することができる。

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

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

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

［例］セル B1 に式 $A$1 ＋ $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 $A$1 + $A5 + B$5 が入る。

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

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

6. 関数

式には次の表で定義する関数を利用することができる。

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

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

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

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

[2] 引数として渡すことができる式の個数は，1以上である。

7. マクロ

(1) ワークシートとマクロ

ワークシートには複数のマクロを格納することができる。

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

[例] ○マクロ: Pro

例は，マクロProの宣言である。

(2) 変数とセル変数

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

[例] ○数値型: row, col

例は，数値型の変数row，colの宣言である。

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

セル変数の表現方法には，絶対表現と相対表現とがある。

セル変数の絶対表現は，セル番地で表す。

セル変数の相対表現は，次の書式で表す。

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

［例1］相対(B5, 2, 3) は，セル E7 を表す変数である。

［例2］相対(B5, −2, −1) は，セル A3 を表す変数である。

(3) 配列

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

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

［例］○文字列型: table[100, 200]

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

(4) 宣言，注釈及び処理

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

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

［例］○数値型: row

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

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

〔 メ モ 用 紙 〕

〔 メ モ 用 紙 〕

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。ただし，問題冊子を切り離して利用することはできません。
9. Java プログラムで使用する API の説明，アセンブラ言語の仕様及び表計算ソフトの機能・用語は，この冊子の末尾を参照してください。
10. 試験時間中，机上に置けるものは，次のものに限ります。
    なお，会場での貸出しは行っていません。
    受験票，黒鉛筆及びシャープペンシル（B又はHB），鉛筆削り，消しゴム，定規，時計（時計型ウェアラブル端末は除く。アラームなど時計以外の機能は使用不可），ハンカチ，ポケットティッシュ，目薬
    これら以外は机上に置けません。使用もできません。
11. 試験終了後，この問題冊子は持ち帰ることができます。
12. 答案用紙は，いかなる場合でも提出してください。回収時に提出しない場合は，採点されません。
13. 試験時間中にトイレへ行きたくなったり，気分が悪くなったりした場合は，手を挙げて監督員に合図してください。