

平成 30 年度 春期
基本情報技術者試験
午後 問題

試験時間

13:00 ~ 15:30 (2 時間 30 分)

注意事項

- 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。
- 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。
- 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つてから始めてください。
- 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

- 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み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、消しき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。

(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 [] に入る正しい答えを、解答群の中から選べ。

春の情報処理技術者試験は、[a] 月に実施される。

解答群 ア 2 イ 3 ウ 4 エ 5

正しい答えは“ウ 4”ですから、次のようにマークしてください。

例題	a	ア	イ	[]	エ	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 3, 問 4, 問 6, 問 7, 問 9 を選択した場合の例]

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

C

COBOL

Java

アセンブラー

表計算

正誤表

基本情報技術者試験 午後 問題

平成30年4月15日実施

ページ	問題番号	行	誤	正	訂正の内容
51	10	上から 9行目	② 差額欄には、入金金額から請求金額を減算した金額を印字する。結果欄のコードが 10 の場合は、 <u>0を印字する。</u>	② 差額欄には、入金金額から請求金額を減算した金額を印字する。結果欄のコードが 10 の場合は、 <u>請求金額欄と差額欄には、0を印字する。</u> 支払方法が <u>カード払い、着払いのいずれかであった場合は、入金金額欄と差額欄には、0を印字する。</u>	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	Web サービスを利用するためのパスワードを安全に保存する方法

●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	論理回路
問 3	データベース	小学生を対象とした、ある子供会の名簿を管理する関係データベース
問 4	ネットワーク	クラウドサービス上でのシステム構築
問 5	ソフトウェア設計	健康管理システムの設計
問 6	プロジェクトマネジメント	EVM (Earned Value Management) 手法を用いたプロジェクト管理
問 7	経営戦略・企業と法務	収益の検討

●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	ヒープの性質を利用したデータの整列

●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)	簡易集計プログラム
問 10	ソフトウェア開発 (COBOL)	注文と入金の情報の突合せ
問 11	ソフトウェア開発 (Java)	表現式を構築するためのライブラリ作成
問 12	ソフトウェア開発 (アセンブラー)	数字列の数値への変換
問 13	ソフトウェア開発 (表計算)	会議室の予約システム

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。
・ 変数 \leftarrow 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	 ↑ ↓
乗除演算	×, ÷, %	
加減演算	+, -	
関係演算	>, <, ≥, ≤, =, ≠	
論理積	and	
論理和	or	

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 Webサービスを利用するためのパスワードを安全に保存する方法に関する次の記述を読んで、設問1～3に答えよ。

A社が提供するWebサービスを利用するには、利用者が決めた利用者IDとパスワードを、Webアプリケーションが動作するサーバに登録しておく必要がある。A社のWebアプリケーションでは、利用者がWebアプリケーションにログインするときに、Webブラウザから利用者IDとパスワードがサーバに送信される。サーバは、受信した利用者IDとパスワードを、照合することによって認証する。利用者が決めたパスワードは、パスワードファイルに平文で保存されている。

近年、パスワードファイルが漏えいし、不正ログインが発生したと考えられる事件が多数報道されている。そこで、A社に勤めるCさんは、自社のWebアプリケーションにおけるパスワードファイルが漏えいした際の不正ログインを防止するための対策について、上司から検討を命じられた。

Cさんは対策として、パスワードを平文で保存するのではなく、ハッシュ関数でパスワードのハッシュ値を計算（以下、ハッシュ化という）し、そのハッシュ値を保存する方式を提案することにした。この方式におけるログイン時の認証では、受信したパスワードから求めたハッシュ値を、パスワードファイルに保存されているハッシュ値と照合する。パスワードの保存の流れと、照合の流れを図1に示す。

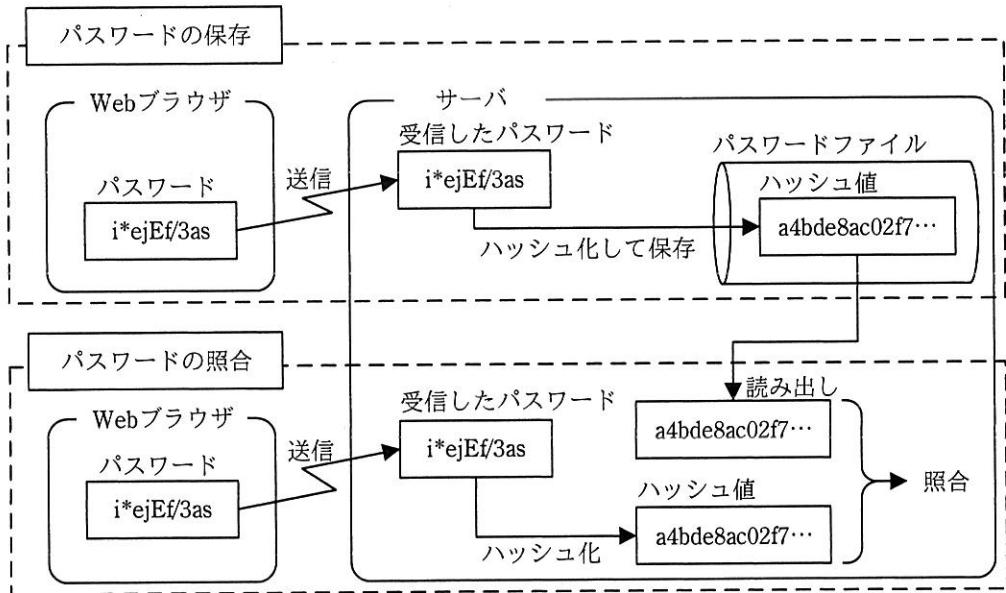


図1 パスワードの保存の流れと、照合の流れ

Cさんは、パスワードのハッシュ化には、ハッシュ関数の一つである a を用いることにした。ハッシュ化に用いるハッシュ関数は、一般的に次のような特徴を備えているので、パスワードが一致していることの確認に用いることができる。また、利用者のパスワードを平文で保存する場合と比べて、パスワードファイルが漏えいしても、より安全だと考えたからである。

[ハッシュ化に用いるハッシュ関数の特徴]

- (1) パスワードの長さに関係なく、ハッシュ値は固定長になる。
- (2) b
- (3) ハッシュ値からパスワードを推測することが非常に困難である。
- (4) パスワードが1文字でも異なれば、ハッシュ値は大きく異なる。

設問1 本文中の に入る適切な答えを、解答群の中から選べ。

aに関する解答群

ア AES

イ Diffie-Hellman

ウ RSA

エ SHA-256

オ TLS

b に関する解答群

- ア 異なるパスワードをハッシュ化したとき、同じハッシュ値になる可能性が高い。
- イ 同一のパスワードをハッシュ化すると、同じハッシュ値になる。
- ウ パスワードをハッシュ化した結果のハッシュ値を再度ハッシュ化すると、元のパスワードになる。
- エ 秘密鍵を使用してハッシュ値から元のパスワードを復元できる。

設問2 次の記述中の に入る適切な答えを、解答群の中から選べ。

Cさんは、自身が提案する方式について、社内の情報セキュリティ責任者にレビューを依頼したところ、この方式は漏えいしたパスワードファイルを攻撃者に入手された場合、事前計算による辞書攻撃に弱いという指摘を受けた。この攻撃では、あらかじめ攻撃者はパスワードとしてよく使われる文字列を、よく使われているハッシュ関数でハッシュ化し、ハッシュ値から元のパスワードが検索可能な一覧表を作成しておく。その後、攻撃者が漏えいしたパスワードファイル入手したとき、この作成した一覧表からハッシュ値を検索する。ハッシュ値が一覧表に載っている場合は、元のパスワードを容易に知ることができる。

Cさんは、事前計算による辞書攻撃を難しくする方式を調査し、ソルトを用いる方式を提案することにした。ソルトとは、十分な長さをもつランダムな文字列である。

この方式におけるパスワードの保存では、まず、サーバは新しいパスワードの保存の都度、新しいソルトを生成し、ソルトとパスワードを連結した文字列をハッシュ化する。このとき得られるハッシュ値は、パスワードだけをハッシュ化した場合のハッシュ値 c。次に、ハッシュ化に使用したソルトと得られたハッシュ値をパスワードファイルに保存する。

この方式におけるパスワードの照合では、まず、サーバはパスワードファイルからソルトとハッシュ値を読み出す。次に、読み出したソルトと受信したパスワードを連結した文字列をハッシュ化し、得られたハッシュ値を、読み出したハッシュ値と照合する。ソルトを用いたパスワードの保存の流れと、照合の流れを図2に示す。

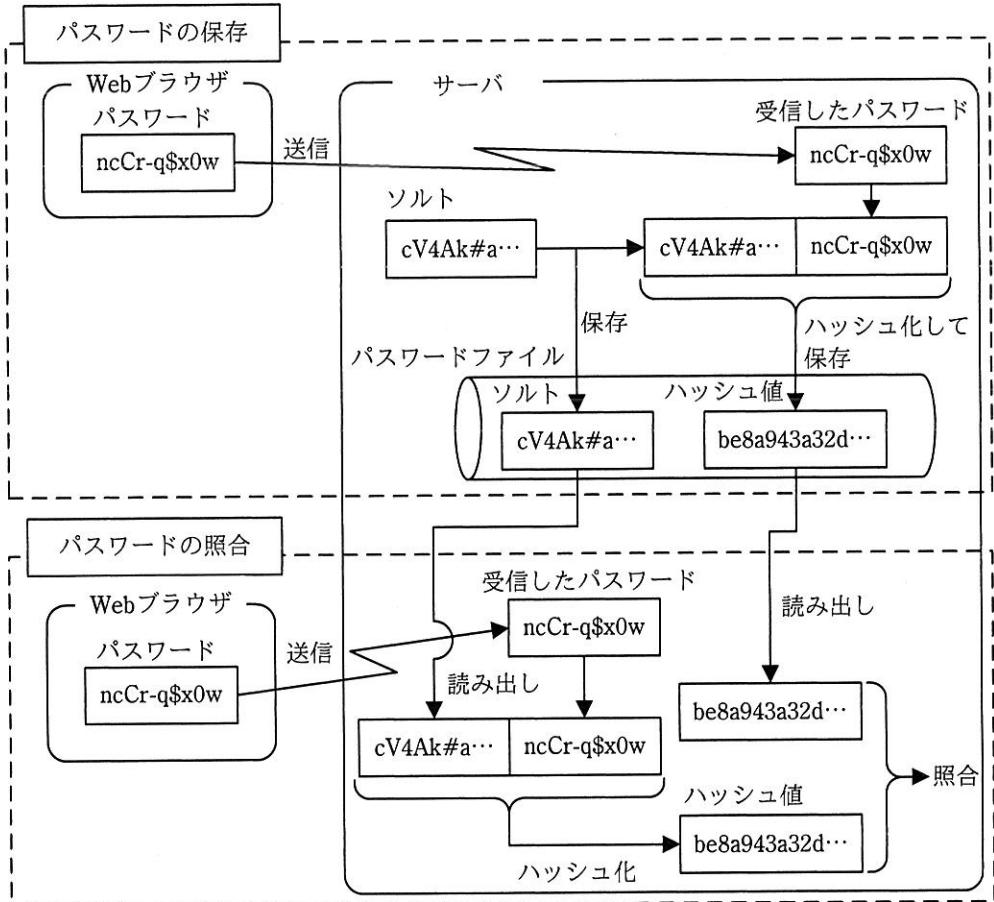


図2 ソルトを用いたパスワードの保存の流れと、照合の流れ

ソルトを用いる方式が、事前計算による辞書攻撃の対策として効果があるのは、dからである。

cに関する解答群

- | | |
|------------|-------------|
| ア 同じ値になる | イ とは異なる値になる |
| ウ よりも長さが長い | エ よりも長さが短い |

dに関する解答群

- | |
|--|
| ア 攻撃者が、ハッシュ値からではなくソルトから元のパスワードを検索するための一覧表を事前に作成しておく必要がある |
| イ 攻撃者がパスワードファイルからソルトを入手できない |
| ウ 攻撃者がパスワードファイル入手するのが困難になる |
| エ 攻撃者が一つのパスワードに対して事前に求めるハッシュ値の数が膨大になる |

設問3 次の記述中の [] に入る適切な答えを、解答群の中から選べ。

Cさんは、オフライン総当たり攻撃についても、対策を検討することにした。

漏えいしたパスワードファイルに対するオフライン総当たり攻撃とは、攻撃者が、パスワードファイルを入手した後、全てのパスワードの候補を逐次生成してはハッシュ化し、得られたハッシュ値がパスワードファイルに保存されているハッシュ値と一致するかどうか、しらみつぶしに確認することによって、ハッシュ値の元のパスワードを見つける攻撃方法である。

Cさんは、オフライン総当たり攻撃を難しくする方式として、ストレッチングという方式があることを知った。

この方式では、まず、ソルトとパスワードを連結した文字列をハッシュ化してハッシュ値を得る。次に、得られたハッシュ値の後にソルトとパスワードを連結し、その連結結果をハッシュ化する。この操作を指定した回数だけ繰り返すことによって、パスワードの照合に用いるハッシュ値を得る。パスワードファイルには、ソルト及びパスワードの照合に用いるハッシュ値に加えて、繰返し回数も保存する。この方式では、ハッシュ化の操作を1回だけ行う方式と比べると、攻撃者が、オフライン総当たり攻撃を行う際、[]。

解答群

- ア 生成すべきパスワードの候補の最大文字列長が長くなる
- イ 一つのパスワードの候補から求めたハッシュ値の長さが長くなる
- ウ 一つのパスワードの候補から求めたハッシュ値を、パスワードファイルのハッシュ値と比較する回数が増える
- エ 一つのパスワードの候補からハッシュ値を求める時間が増加する

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 論理回路に関する次の記述を読んで、設問1～3に答えよ。

主要な論理演算の真理値表を表1に示す。

表1 主要な論理演算の真理値表

入力		出力			
		AND (論理積)	OR (論理和)	NAND (否定論理積)	NOR (否定論理和)
0	0	0	0	1	1
0	1	0	1	1	0
1	0	0	1	1	0
1	1	1	1	0	0

設問1 表1に示した論理演算を行う論理回路を用いて、表2に示すXOR（排他的論理和）の論理演算を行う論理回路を図1のとおり作成した。図1中の
[] に入る正しい答えを、解答群の中から選べ。

表2 XOR（排他的論理和）の真理値表

入力		出力
0	0	0
0	1	1
1	0	1
1	1	0

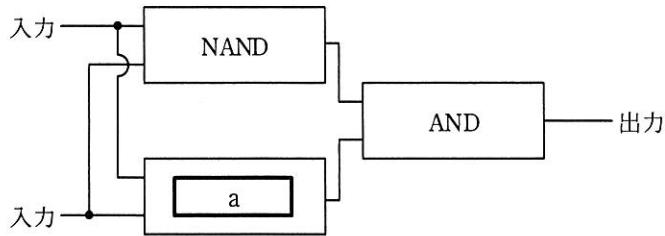


図 1 XOR (排他的論理和) の論理回路

a に関する解答群

ア AND

イ NAND

ウ NOR

エ OR

設問 2 1 桁の 2 進数 X, Y を入力して、その和の下位桁を Z, 桁上がりを C に出力する半加算器の論理回路を図 2 に示す。図 2 中の [] に入れる正しい答えを、解答群の中から選べ。

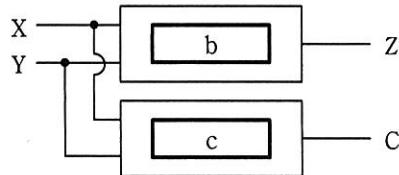


図 2 半加算器の論理回路

b, c に関する解答群

ア AND

イ NAND

ウ NOR

エ OR

オ XOR

設問 3 論理回路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。

この論理回路は、1 ビットの入力 X, Y をそれぞれパラメタ W_x , W_y で重み付けして加算した結果を求め、パラメタ T をしきい値として、次のとおりに動作する。

$W_x \times X + W_y \times Y \geq T$ のとき, 1をZに出力する。

$W_x \times X + W_y \times Y < T$ のとき, 0をZに出力する。

例えば, パラメタ W_x が 0.5, W_y が 0.5, T が 0.3 (以下, パラメタ [0.5, 0.5, 0.3] のように表記する) の場合には, 表 3 に示すとおり, この論理回路における入力と出力の関係 (以下, 入出力関係という) は OR (論理和) になる。

表 3 パラメタ [0.5, 0.5, 0.3] の場合の入出力関係

入力		$W_x \times X + W_y \times Y$	T	出力
X	Y			Z
0	0	$0.5 \times 0 + 0.5 \times 0$	0.3	0
0	1	$0.5 \times 0 + 0.5 \times 1$	0.3	1
1	0	$0.5 \times 1 + 0.5 \times 0$	0.3	1
1	1	$0.5 \times 1 + 0.5 \times 1$	0.3	1

同様に,

- (1) AND (論理積) になる入出力関係は, パラメタ d で実現できる。
(2) NAND (否定論理積) になる入出力関係は, パラメタ e で実現できる。

d, e に関する解答群

ア [-0.5, -0.5, -0.8]

イ [-0.5, -0.5, -0.2]

ウ [0.5, 0.5, -0.5]

エ [0.5, 0.5, 0.2]

オ [0.5, 0.5, 0.8]

カ [0.5, 0.5, 1.5]

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 小学生を対象とした、ある子供会の名簿を管理する関係データベースに関する次の記述を読んで、設問1~4に答えよ。

D子供会は、小学校に入学するときに入会を受け付け、小学校を卒業したら退会する。D子供会では、会員名簿を管理するためのデータベースを構築して、会の運営に活用している。

このたび、児童のイベントへの参加実績を記録するために、活動表とイベント表を追加した。

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、主キーを表す。

保護者表

保護者番号	保護者氏名	電話番号	住所
12021	情報花子	03-1111-2222	東京都○○区□□□
:	:	:	:

児童表

児童番号	児童氏名	学年	保護者番号
12027	情報一郎	6	12021
14021	情報二郎	4	12021
:	:	:	:

活動表

児童番号	イベント番号
12027	18001
14021	18001
12027	18002
:	:

イベント表

イベント番号	イベント名	開催日
18001	歓迎会	20180407
18002	地域清掃	20180414
:	:	:

図1 データベースの表構成とデータ格納例

設問1 6年生を対象に実施するイベントの案内を配布するために、6年生の保護者の氏名と住所を抽出する。ここで、同一の保護者は重複して抽出しない。また、同じ住所に氏名が同じ保護者は、複数人いなものとする。正しいSQL文を、解答群の中から選べ。

解答群

- ア SELECT DISTINCT 保護者表.保護者氏名, 保護者表.住所
FROM 保護者表
WHERE 保護者表.保護者番号 NOT IN
(SELECT 児童表.保護者番号 FROM 児童表 WHERE 児童表.学年 = 6)
- イ SELECT DISTINCT 保護者表.保護者氏名, 保護者表.住所
FROM 保護者表, 児童表
WHERE 児童表.学年 = 6
GROUP BY 保護者表.保護者氏名, 保護者表.住所
- ウ SELECT DISTINCT 保護者表.保護者氏名, 保護者表.住所
FROM 保護者表, 児童表
WHERE 保護者表.保護者番号 = 児童表.保護者番号 AND 児童表.学年 = 6
- エ SELECT 保護者表.保護者氏名, 保護者表.住所
FROM 保護者表, 児童表
WHERE 保護者表.保護者番号 = 児童表.保護者番号
GROUP BY 保護者表.保護者氏名, 保護者表.住所 HAVING 児童表.学年 = 6

設問2 イベント番号が18001のイベントに参加した児童のうち、1年生である児童の保護者の保護者番号と氏名を抽出する。ここで、同一の保護者は重複して抽出しない。次のSQL文の [] に入る正しい答えを、解答群の中から選べ。

SELECT DISTINCT 保護者表.保護者番号, 保護者表.保護者氏名

[]
a

aに関する解答群

- ア FROM 児童表, 保護者表, イベント表
WHERE 児童表.学年 = 1 AND
イベント表.イベント番号 = 18001
- イ FROM 児童表, 保護者表, イベント表
WHERE 児童表.保護者番号 = 保護者表.保護者番号 AND
児童表.学年 = 1 AND
イベント表.イベント番号 = 18001
- ウ FROM 児童表, 活動表, 保護者表
WHERE 児童表.児童番号 = 活動表.児童番号 AND
児童表.保護者番号 = 保護者表.保護者番号 AND
活動表.イベント番号 = 18001
GROUP BY 児童表.児童氏名 HAVING 児童表.学年 = 1
- エ FROM 児童表, 活動表, 保護者表
WHERE 児童表.児童番号 = 活動表.児童番号 AND
児童表.保護者番号 = 保護者表.保護者番号 AND
児童表.学年 = 1 AND
活動表.イベント番号 = 18001

設問3 イベント名と、そのイベントに参加した児童の数を表示する。次のSQL文
の [] に入る正しい答えを、解答群の中から選べ。ここで、イベン
ト名は全て異なるものとする。

```
SELECT イベント表.イベント名, [ ] b  
FROM 活動表, イベント表  
WHERE 活動表.イベント番号 = イベント表.イベント番号  
GROUP BY イベント表.イベント名
```

bに関する解答群

- ア AVG(活動表.イベント番号)
イ COUNT(*)
ウ MAX(活動表.イベント番号)
エ SUM(活動表.イベント番号)

設問4 年度の切替えのために、次に示す手順で表を更新する。(1), (2)は入会前の準備のために3月31日に実行し、(3)～(7)は6年生が退会した4月1日に実行する。次のSQL文の [] に入る正しい答えを、解答群の中から選べ。

[手順]

- (1) 新入会児童の保護者のうち、未登録の保護者を登録する。
- (2) 新入会児童を登録する。このとき、学年の値は0とする。
- (3) 活動表のレコードを全て削除する。
- (4) 児童表の全ての児童に対して、学年の値に1を加える。
- (5) 児童表から、学年の値が7の児童を削除する。
- (6) 次のSQL文を実行して、保護者表から、在籍する児童がいなくなった保護者を削除する。

DELETE FROM 保護者表

WHERE [] c

- (7) イベント表のレコードを全て削除してから、新年度の計画に合わせてイベントを登録する。

cに関する解答群

- ア 保護者表.保護者番号 = NULL
- イ 保護者表.保護者番号 IN
(SELECT 児童表.保護者番号 FROM 児童表 WHERE 児童表.学年 = 7)
- ウ 保護者表.保護者番号 IN
(SELECT 児童表.保護者番号 FROM 児童表)
- エ 保護者表.保護者番号 NOT IN
(SELECT 児童表.保護者番号 FROM 児童表)

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 クラウドサービス上でのシステム構築に関する次の記述を読んで、設問1, 2に答えよ。

G社は、J社が運営するクラウドサービス上で、写真投稿サービス及び写真検索サービスを構築することにした。

- (1) 写真投稿サービスは、利用者から投稿された写真を受け付け、自動で分類し、保管するサービスである。
- (2) 写真検索サービスは、利用者から指定された条件に合致する写真を、保管されている写真の中から検索し、表示させるサービスである。
- (3) 利用者は、PC、スマートフォンなど（以下、クライアントという）を用いてサービスを利用する。

システム構成を図1に示す。図1中の矢印の向きはアクセスの方向を示している。

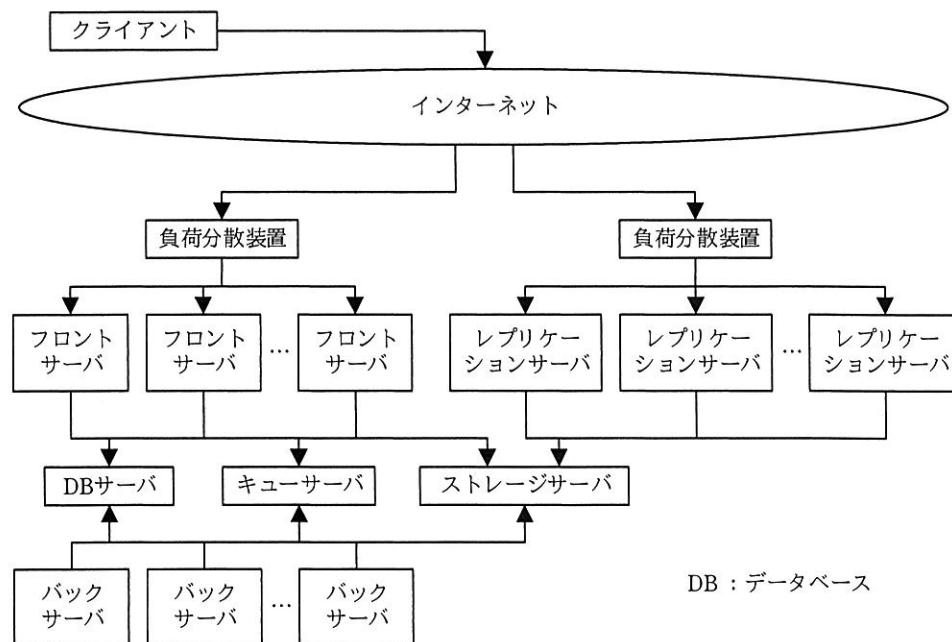


図1 システム構成

写真投稿サービスにおける処理の概要は、次のとおりである。

- (1) フロントサーバは、クライアントから写真を受け取り、一意な ID を写真に割り当て、ストレージサーバに保存する。
- (2) フロントサーバは、写真の ID、ストレージサーバ上での写真の保存場所などを、その写真の属性情報として DB サーバに登録する。
- (3) フロントサーバは、キューサーバに写真の ID を格納する。
- (4) バックサーバは、キューサーバから写真の ID を取得する。
- (5) バックサーバは、(4)で取得した ID に該当する写真の属性情報を DB サーバから検索し、ストレージサーバから写真を取得する。
- (6) バックサーバは、(5)で取得した写真をあるアルゴリズムによって分類し、分類結果を DB サーバのその写真の属性情報に付加する。
- (7) レプリケーションサーバは、ストレージサーバに定期的にアクセスし、新規に保存された写真を取得して自サーバ上に保存する。レプリケーションサーバ上の写真の保存場所は、ストレージサーバ上のそれと一意に対応付けられるように、あらかじめ定めてある規則に従って決定する。

写真検索サービスにおける処理の概要は、次のとおりである。

- (1) フロントサーバは、クライアントから検索要求を受け取り、条件に合致する写真の属性情報を DB サーバから検索する。
- (2) フロントサーバは、検索された写真の属性情報から、レプリケーションサーバに保存された写真にアクセスするための URL を作成する。
- (3) フロントサーバは、作成した URL を含む HTML データを生成してクライアントに返す。
- (4) クライアントは、フロントサーバから返された HTML データに基づきレプリケーションサーバにアクセスし、写真を取得して表示する。

なお、クライアントは、インターネットと負荷分散装置を介して、フロントサーバとレプリケーションサーバにアクセスする。

サーバは仮想マシン上で稼働させる。フロントサーバ及びバックサーバを稼働させ

る仮想マシンの主記憶容量やディスク容量は十分にあり、負荷に応じて台数を増減できる。

計算処理能力やネットワーク処理能力に着目すると、仮想マシンには幾つかのタイプがある。仮想マシンのタイプを表1に示す。

表1 仮想マシンのタイプ

タイプ	計算処理能力	ネットワーク処理能力	コスト(円/時間)
A	1	1	10
B	2	1.5	18
C	4	2	34
D	8	2	60

表1中の計算処理能力は、タイプAの計算処理能力を1としたときの相対的な値である。ネットワーク処理能力は、タイプAのネットワーク処理能力を1としたときの相対的な値である。

“1秒の計算処理量”とは、タイプAの仮想マシン1台を計算処理能力の100%で1秒間使用したときの処理量をいう。また、“1秒のネットワーク処理量”とは、タイプAの仮想マシン1台をネットワーク処理能力の100%で1秒間使用したときの処理量をいう。

フロントサーバにおいては、1要求当たり、計算処理量は0.1秒、ネットワーク処理量は0.07秒である。

クライアントからの要求が非常に多いとき、フロントサーバのコストを最も低く抑えることができる仮想マシンのタイプは a である。ここで、各仮想マシンの計算処理能力とネットワーク処理能力の平均の使用率は、それぞれ50%以下に抑えることとする。

バックサーバにはタイプDの仮想マシンを使用する。

バックサーバの写真1枚当たりの計算処理量は、25秒である。1時間当たり4,000枚の写真的投稿があるとき、計算処理能力の平均の使用率を50%以下とするのに最低限必要な仮想マシンの台数は b 台である。ここで、ネットワーク処理能

力は足りているものとする。

図 1 中の各サーバ及び負荷分散装置（以下、サーバ類という）は表 2 に示すいずれかのグループに属しており、グループごとに他のグループやインターネットからのアクセス許可を設定することができる。サーバ類が受け付けるプロトコルを表 3 に示す。

表 2 グループとグループに属するサーバ類との対応

グループ	グループに属するサーバ類
1	負荷分散装置
2	フロントサーバ
3	DB サーバ, キューサーバ
4	バックサーバ
5	ストレージサーバ
6	レプリケーションサーバ

表 3 サーバ類が受け付けるプロトコル

サーバ類	プロトコル	ポート番号
フロントサーバ	HTTP	80
キューサーバ	独自	15672
バックサーバ	無し	無し
ストレージサーバ	HTTP	80
DB サーバ	独自	15432
レプリケーションサーバ	HTTP	80
負荷分散装置	HTTP over TLS	443

各グループが許可するアクセスを必要最低限とした結果、c が許可するアクセスは一致する。また、グループ 3 が許可するアクセスは表 4 に示すとおりになった。

表 4 は、グループ 3 に属するサーバ類が、アクセス元に指定したグループに属するサーバ類からの、指定したポート番号のポートを介してのアクセスを許可することを示している。

表4 グループ3が許可するアクセス

アクセス元	ポート番号
グループ2	15432と15672
グループ4	15432と15672

設問1 本文中の 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 A

イ B

ウ C

エ D

bに関する解答群

ア 4

イ 7

ウ 28

エ 56

cに関する解答群

ア グループ1, 2, 5及び6

イ グループ2, 5及び6

ウ グループ2及び6

エ グループ5及び6

設問2 クライアントからの写真へのアクセスを、ストレージサーバがインターネットを介して直接受ける方法も考えられるが、この方法ではなく、図1のように負荷分散装置を介し、レプリケーションサーバが受けることの利点として適切な答えを、解答群の中から二つ選べ。

解答群

ア クライアントからの写真へのアクセスが増加しても、ストレージサーバの負荷は高まらない。

イ クライアントと写真へのアクセスに応答するサーバとの間に介在するサーバ類の台数が少ないので、ネットワーク遅延が小さい。

ウ ストレージサーバに障害が発生しても、写真検索サービスの提供を継続できる。

エ ストレージサーバに障害が発生しても、写真投稿サービスの提供を継続できる。

オ 全てのフロントサーバに障害が発生しても、写真検索サービスの提供を継続できる。

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 健康管理システムの設計に関する次の記述を読んで、設問1、2に答えよ。

K社は、従業員の健康診断管理業務に健康管理システムを利用している。全従業員には、年1回、誕生日に定期健康診断を受診させる。再検査が必要となった場合には、再度、健康診断を受診させる。

[健康管理システムを利用した健康診断管理業務の説明]

月初に、受診日（当月中旬の特定日）を指定して受診案内を通知する。受診すれば、受診した月の月末に健康診断結果を通知する。

- (1) 健康管理システムに、当月の受診対象者を登録する。
 - ① 当月が誕生日である従業員を抽出する。
 - ② 前月の定期健康診断において、再検査が必要と判定された従業員を抽出する。
 - ③ ①と②で抽出した従業員を受診対象者とし、健康診断コースを決定する。
 - ④ 受診日を決定し、受診対象者の健康診断レコードを健康管理システムに登録する。
- (2) 受診案内を受診対象者に通知する。
- (3) 定期健康診断及び再検査の受診後に、受診者の判定結果などを健康管理システムに登録する。
- (4) 受診者に、健康診断結果を通知する。

[健康管理システムの機能の説明]

健康管理システムの機能一覧を、表1に示す。表1において、“業務との対応”列の項目番号は、[健康管理システムを利用した健康診断管理業務の説明]の項目番号に対応する。各機能では、健康管理システムの健康診断ファイルのアクセスと、人事システムの各マスタファイル（以下、マスタという）の情報の参照を行う。

表1 健康管理システムの機能一覧

機能名	処理内容	業務との対応
定期健康診断対象者抽出機能	当月が誕生月である従業員を従業員マスタから抽出する。	(1)①
再検査対象者抽出機能	前月に定期健康診断を受診した従業員のうち、再検査が必要と判定された従業員を、健康診断ファイルから抽出する。	(1)②
健康診断内容決定機能	定期健康診断対象者抽出機能、再検査対象者抽出機能で抽出した従業員に対して、健康診断コース及び健康診断実施場所を決定する。 健康診断コースは、定期健康診断の場合は従業員の年齢、性別、管理職かどうかで決定する。再検査の場合は、前月に受診した定期健康診断の健康診断コースと同じとする。 健康診断実施場所は、転勤になることも考慮し、従業員の現在の在勤地に基づいて決定する。	(1)③
対象者登録機能	健康診断レコードを作成して健康診断ファイルに登録する。	(1)④
判定結果登録機能	健康診断レコードの判定結果、再検査要否及び判定年月日を更新する。 再検査要否は、定期健康診断後に再検査が必要な場合は“要”で、必要ない場合は“否”で更新する。再検査後の場合は、再検査要否は常に“否”で更新する。 判定年月日は、処理を実行した年月日で更新する。	(3)

〔健康管理システム及び人事システムの関係の説明〕

健康管理システムは、健康診断結果を健康診断ファイルで管理し、従業員の情報については、必要な項目の最新情報を人事システムの各マスタを参照して利用する。健康管理システム及び人事システムの関係は、図1のとおりである。

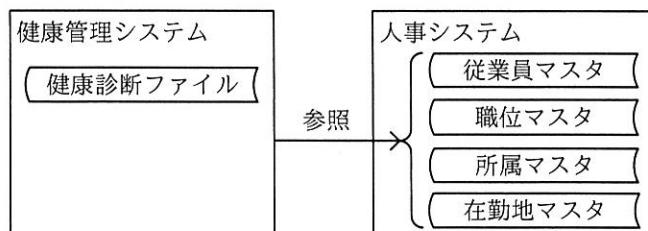


図1 健康管理システム及び人事システムの関係

[健康診断ファイルの説明]

健康診断ファイルのレコード様式を、図2に示す。下線付きの項目は、主キーである。

従業員 コード	<u>対象年</u>	実施 年月	健康診断 コース	区分	健康診断 実施場所	受診 年月日	判定 結果	再検査 要否	判定 年月日
------------	------------	----------	-------------	----	--------------	-----------	----------	-----------	-----------

図2 健康診断ファイルのレコード様式

- (1) 対象年には、定期健康診断の受診対象となった年を設定する。再検査の場合は、前月に受診した定期健康診断の対象年と同じとする。
- (2) 実施年月には、健康診断を受診する年月を設定する。
- (3) 区分には、定期健康診断の場合は“定期”を、再検査の場合は“再検”を設定する。
- (4) 判定結果、再検査要否及び判定年月日は、対象者登録機能で空白を設定し、判定結果登録機能で更新する。

[人事システムの各マスタの説明]

健康管理システムが参照する人事システムの各マスタのレコード様式を、図3に示す。下線付きの項目は、主キーである。

従業員マスタ

従業員コード	氏名	性別	生年月日	職位コード	所属コード	在勤地コード
--------	----	----	------	-------	-------	--------

職位マスタ

職位コード	職位名	管理職フラグ
-------	-----	--------

所属マスタ

所属コード	所属名
-------	-----

在勤地マスタ

在勤地コード	在勤地名	健康診断実施場所
--------	------	----------

図3 健康管理システムが参照する人事システムの各マスタのレコード様式

従業員マスタは、従業員の氏名、性別、生年月日などを保持する。従業員コードは、従業員に一意に付けられた番号であり、在勤地コードには、現在の在勤地の在勤地コードが入っている。職位マスタは、職位名と管理職フラグを保持し、管理職フラグで管理職かどうかが判別できる。在勤地マスタは、在勤地名とその在勤地の健康診断実施場所を保持する。

設問1 次の記述中の [] に入る適切な答えを、解答群の中から選べ。

健康診断内容決定機能において、定期健康診断対象者抽出機能で従業員マスタから抽出された情報に加えて、定期健康診断の健康診断コース及び健康診断実施場所を決定するために最低限必要となるのは、[a] の情報である。再検査対象者抽出機能で健康診断ファイルから抽出された情報に加えて、再検査の健康診断実施場所を決定するために最低限必要となるのは、[b] の情報である。

a, b に関する解答群

- ア 従業員マスタ
- イ 従業員マスタ及び職位マスタ
- ウ 従業員マスタ及び在勤地マスタ
- エ 従業員マスタ、職位マスタ及び在勤地マスタ
- オ 職位マスタ
- カ 職位マスタ及び所属マスタ
- キ 職位マスタ及び在勤地マスタ
- ク 所属マスタ
- ケ 所属マスタ及び在勤地マスタ
- コ 在勤地マスタ

設問2 次の記述中の [] に入る適切な答えを、解答群の中から選べ。

今回、健康管理業務として、12月の受診者の健康診断の判定結果を登録し

た後に、当年の健康診断の未受診者を抽出することになった。そのために、未受診者抽出機能を追加することにした。ここで、当年の健康診断の未受診者は、従業員マスタから情報が抽出できる従業員のうち、次のいずれかに該当する者とする。

- (1) 当年の定期健康診断の健康診断レコードが存在しない。
- (2) 当年の定期健康診断の判定結果が登録されていない。
- (3) 当年の定期健康診断の結果、再検査が必要と判定されたが、再検査の健康診断レコードが存在しない、又はその判定結果が登録されていない。

未受診者抽出機能の処理の流れを図4に示す。ここで、図4の従業員マスタは従業員コードの昇順で整列されているものとする。

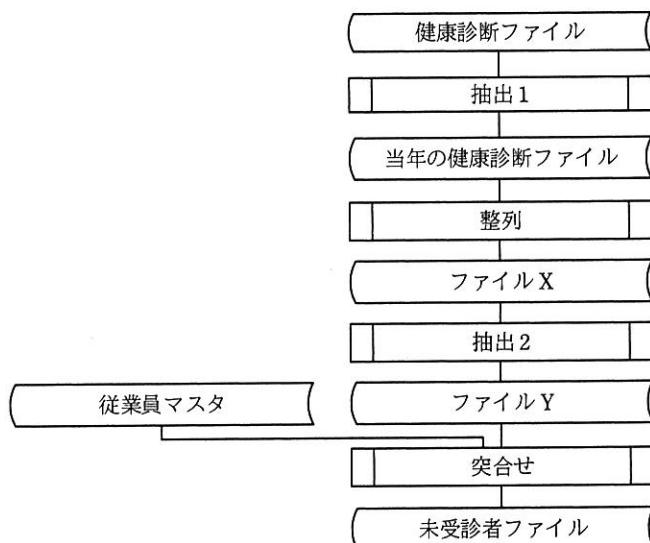


図4 未受診者抽出機能の処理の流れ

[未受診者抽出機能の処理の説明]

- (1) 図4の突合せ処理において、各ファイルを先頭から順次読み込んで処理できるように、前処理として次の①～③を実行する。

① 抽出1処理で、対象年が当年のレコードを健康診断ファイルから抽出し、当年の健康診断ファイルを作成する。

② 整列処理で、当年の健康診断ファイルを [] で整列し、ファイル

ル X を作成する。

③ 抽出 2 処理で、ファイル X のレコードを先頭から順にファイル Y に出力する。ここで、d が連続した場合には、最初に出現したレコードだけを出力する。

(2) 突合せ処理で、従業員マスタに存在する従業員コードに対して、次に示す①又は②のいずれかに該当する場合に、従業員コードを未受診者ファイルに出力する。

- ① 従業員コードが一致するレコードがファイル Y に存在しない。
- ② e のレコードがファイル Y に存在する。

c に関する解答群

- ア 従業員コードの昇順
- イ 従業員コードの昇順、実施年月の昇順
- ウ 従業員コードの昇順、実施年月の降順
- エ 従業員コードの降順
- オ 従業員コードの降順、実施年月の昇順
- カ 従業員コードの降順、実施年月の降順

d に関する解答群

- | | |
|-------------|-----------------|
| ア 同じ従業員コード | イ 同じ従業員コード及び区分 |
| ウ 同じ健康診断コース | エ 同じ健康診断コース及び区分 |

e に関する解答群

- ア 判定結果が空白
- イ 再検査要否が“要”
- ウ 再検査要否が“否”
- エ 判定結果が空白、又は再検査要否が“要”
- オ 判定結果が空白、又は再検査要否が“否”

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 EVM (Earned Value Management) 手法を用いたプロジェクト管理に関する次の記述を読んで、設問1~3に答えよ。

S社では、クライアントサーバシステムとして構築されている既存の営業システムを、Webシステムに刷新するプロジェクト（以下、刷新プロジェクトという）を立ち上げた。Webシステムとして構築する営業システムを、新営業システムと呼ぶ。N社は、この刷新プロジェクトにおける外部設計から結合テストまでを受注した。

〔刷新プロジェクトへのN社の対応〕

(1) S社から提示された刷新プロジェクトの結合テストまでのスケジュールは、図1のとおりである。

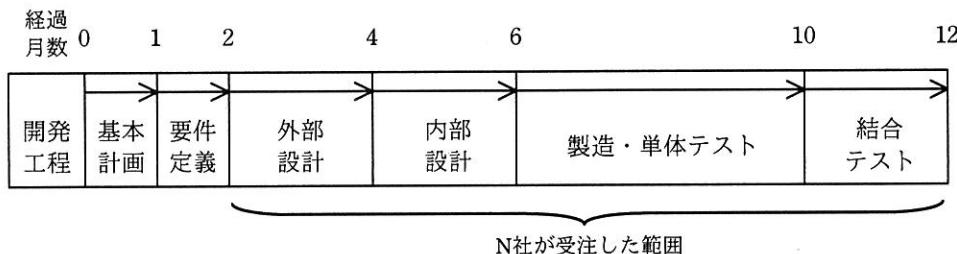


図1 刷新プロジェクトの結合テストまでのスケジュール

(2) 新営業システムは、サブシステムS1（以下、S1という）、サブシステムS2（以下、S2という）及び共通機能の三つのサブシステムで構成される。N社はこの構成に合わせて外部設計から結合テストまでを実施するプロジェクト（以下、プロジェクトという）体制を構築し、プロジェクトマネージャは、システム企画部に所属するY君が担当する。

(3) N社では、新営業システムと類似するWebシステムの開発をこれまで複数案件行っており、組織のプロセス資産として各開発工程の生産性のデータを保有している。N社が保有する各開発工程の生産性を、表1に示す。Y君は、要件定義の成果物である要件定義書とN社の過去の開発実績を参考にして、表2に示すとおりに

新営業システムの開発規模を見積もった。ここで、KLOC (Kilo Lines Of Code) は、ソースコード 1,000 行を単位とする指標である。

表 1 各開発工程の生産性

単位 KLOC／人月	
開発工程	生産性
外部設計	8.00
内部設計	6.40
製造・単体テスト	3.20
結合テスト	8.00

表 2 新営業システムの開発規模

単位 KLOC	
サブシステム	開発規模
S1	200.00
S2	240.00
共通機能	80.00

Y 君は、(1)～(3) を前提に、各開発工程を完了させるために必要となる計画時点の工数（以下、計画工数という）を算出してプロジェクト計画を作成した。ここで、1か月の作業日数は 20 日とする。各開発工程における計画工数は、作業する各月に対して均等に割り当てる。例えば、外部設計を 10.00 人月と見積もった場合、外部設計の期間は 2 か月なので、各月に 5.00 人月を割り当てる。

設問 1 外部設計と内部設計のサブシステムごとの計画工数を、表 3 に示す。表 3 中の に入れる正しい答えを、解答群の中から選べ。

表 3 外部設計と内部設計のサブシステムごとの計画工数

単位 人月		
サブシステム	外部設計	内部設計
S1	25.00	<input type="text" value="a"/>
S2	30.00	
共通機能	10.00	
合計	65.00	

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 25.75

イ 30.00

ウ 31.25

エ 37.50

設問2 次の記述中の [] に入る正しい答えを、解答群の中から選べ。

N社では、進捗遅延やコスト超過を早期に検出することを目的として、EVM手法を用いたプロジェクト管理を行っている。EVM手法では、プロジェクトの計画と実績について、定量的な情報を用いて進捗状況やコスト状況を分析し、評価する。EVM手法で使う各指標（以下、EVM指標という）のN社での意味を、表4に示す。

表4 EVM指標のN社での意味

指標	名称	意味
BAC(人月)	完了までの総予算	プロジェクト計画時点に見積もったプロジェクト完了までの予定総工数のこと。
PV(人月)	出来高計画値	プロジェクト計画時点において、プロジェクト状況の評価時点までの予定作業に割り当てていた予定工数のこと。プロジェクト完了時点におけるPVはBACと同じ値になる。
EV(人月)	出来高実績値	プロジェクト状況の評価時点までに完了した作業に対して、プロジェクト計画時点において割り当てていた予定工数のこと。
AC(人月)	コスト実績値	プロジェクト状況の評価時点までに完了した作業に対して、実際に必要となった工数のこと。
SV(人月)	スケジュール差異	プロジェクト状況の評価時点におけるEVとPVの差のこと。スケジュール面から見た差異を示す。
CV(人月)	コスト差異	プロジェクト状況の評価時点におけるEVとACの差のこと。コスト面から見た差異を示す。
SPI	スケジュール効率指数	$SPI=EV/PV$ スケジュール面から見た効率のこと。1未満の場合、進捗遅延していることを示す。
CPI	コスト効率指数	$CPI=EV/AC$ コスト面から見た効率のこと。1未満の場合、コスト超過していることを示す。

外部設計を開始してから35作業日(1.75か月)が経過した時点での、外部設計工程のサブシステムごとのEVM指標値を、表5に示す。ここで、表5のEVM指標値は、小数点第3位を四捨五入した値である。

表 5 外部設計工程のサブシステムごとの EVM 指標値

サブシステム	外部設計終了時点	外部設計を開始してから 1.75 か月が経過した時点			単位 人月
	PV (出来高計画値)	PV (出来高計画値)	EV (出来高実績値)	AC (コスト実績値)	
S1	25.00	21.88	21.15	21.60	
S2	30.00	26.25	26.25	26.55	
共通機能	10.00	8.75	8.75	8.10	

次の(1)～(6)の記述のうち、表 5 の EVM 指標値から予測した、外部設計が終了する時点での見通しとして適切な組合せは、 である。ここで、予測に当たって SPI 及び CPI は、表 5 の実績と同等であるものとする。

- (1) S1 については、AC が外部設計終了時点の PV を超過しない。また、外部設計はスケジュール遅延せずに完了する。
- (2) S1 については、AC が外部設計終了時点の PV を超過する。また、外部設計はスケジュール遅延せずに完了する。
- (3) S2 については、外部設計はスケジュール遅延する。
- (4) S2 については、AC が外部設計終了時点の PV を超過する。また、外部設計はスケジュール遅延せずに完了する。
- (5) 共通機能については、外部設計はスケジュール遅延する。
- (6) 共通機能については、AC が外部設計終了時点の PV を超過しない。また、外部設計はスケジュール遅延せずに完了する。

解答群

- | | | | |
|-----------|-----------|-----------|-----------|
| ア (1)と(4) | イ (1)と(5) | ウ (2)と(3) | エ (2)と(4) |
| オ (3)と(5) | カ (3)と(6) | キ (4)と(5) | ク (4)と(6) |

設問 3 次の記述中の に入る正しい答えを、解答群の中から選べ。ここで、e1～e3 に入る答えは、e 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2は、外部設計を開始した時点から、結合テストを開始して1か月が経過した時点までの、各経過月末時点のEVM指標値をグラフにしたものである。Y君は、このグラフを用いてプロジェクトの現状分析を行うことにした。

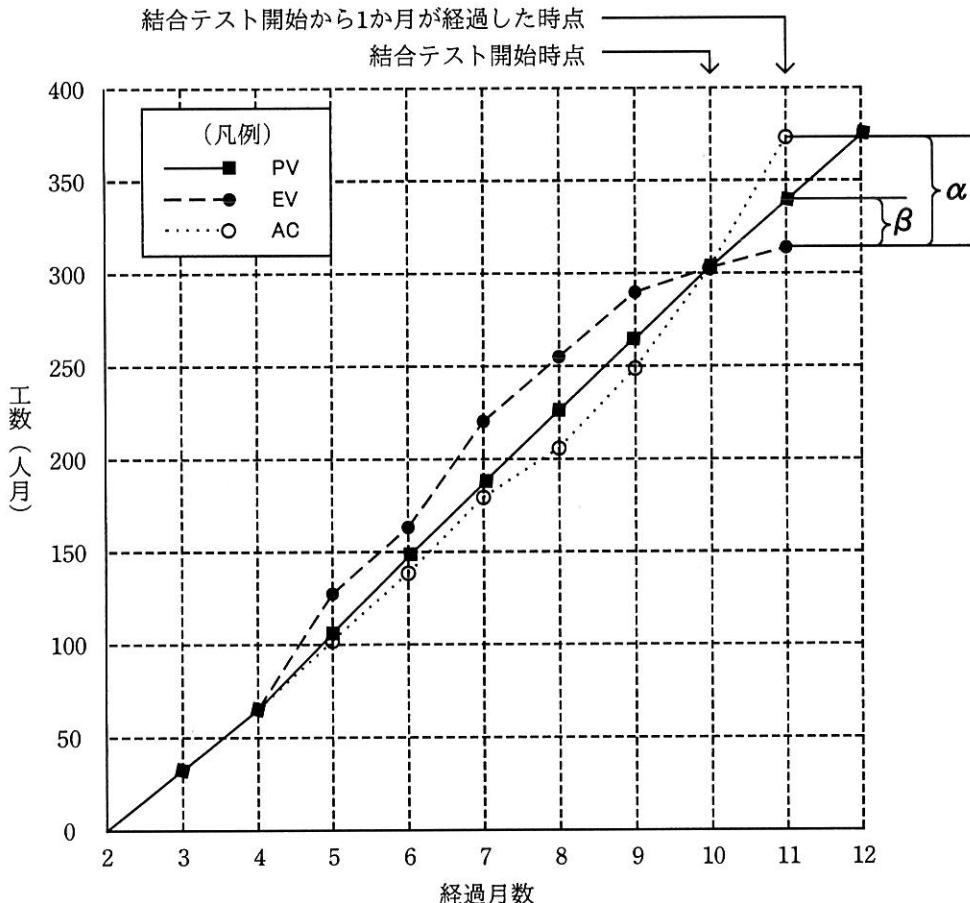


図2 各経過月末時点のEVM指標値

結合テスト開始後、bであることがグラフから読み取れる。また、
αはcを表し、βはdを表している。

次に、未完了である残作業の作業工数（以下、残作業工数という）を式
“e1 - e2”で求めた。この残作業工数は、プロジェクト計
画時点の見積りに基づいているが、今後の作業は計画どおり進捗するものとし
て、この残作業工数にe3を加算し、プロジェクトの全ての作業が完
了したときの総工数の予測値を見直した。

b に関する解答群

- ア 進捗状況とコスト状況のどちらも悪化傾向
- イ 進捗状況とコスト状況のどちらも改善傾向
- ウ 進捗状況は改善傾向、コスト状況は悪化傾向
- エ 進捗状況は悪化傾向、コスト状況は改善傾向

c, d に関する解答群

- | | |
|------------------|---------------|
| ア BAC (完了までの総予算) | イ PV (出来高計画値) |
| ウ EV (出来高実績値) | エ AC (コスト実績値) |
| オ SV (スケジュール差異) | カ CV (コスト差異) |

e に関する解答群

	e1	e2	e3
ア	BAC	AC	EV
イ	BAC	EV	AC
ウ	BAC	PV	EV
エ	PV	AC	EV
オ	PV	BAC	EV
カ	PV	EV	AC

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 収益の検討に関する次の記述を読んで、設問1～3に答えよ。

小規模な部品メーカーであるR社は、部品Tだけを生産して大手機械メーカーに販売している。

設問1 次の記述中の [] に入る適切な答えを、解答群の中から選べ。

利益計画を策定するために、部品Tの販売数について2通りの検討を行つた。表1は、部品Tの販売数を1,000千個見込むケースXと、1,200千個見込むケースYについての収益検討表である。両ケースの、売上高に対する変動費の比率（以下、変動費率という）は等しく、固定費は同額である。

表1 収益検討表

ケース	X	Y
販売数（千個）	1,000	1,200
売上高（千円）	200,000	240,000
変動費（千円）		
固定費（千円）		
利益（千円）	16,000	26,000

注記 網掛けの部分は表示していない。

表1から、変動費率は [a] %、固定費は [b] 千円である。よって、利益が0になる売上高（以下、損益分岐点売上高という）は、[c] 千円となる。

R社では、販売先から値下げ要求があることを想定して、販売数及び変動費を変えずに販売単価を下げた場合の値下げ率（値下げ額÷値下げ前の販売単価）と利益の計算を行った。利益がマイナスにならない最大の値下げ率は、ケースXでは %であり、ケースYではケースX %。

aに関する解答群

ア 25 イ 40 ウ 60 エ 75 オ 80

bに関する解答群

ア 16,000 イ 34,000 ウ 64,000 エ 104,000 オ 134,000

cに関する解答群

ア 45,333 イ 64,000 ウ 85,333 エ 136,000 オ 256,000

dに関する解答群

ア 4 イ 8 ウ 10 エ 12 オ 16

eに関する解答群

ア と変わらない イ よりも大きい ウ よりも小さい

設問2 R社は、変動費と固定費の合計（以下、費用という）と売上高の関係を他の3社と比較して、分析した。その結果、変動費率はR社が他社と比べて最も高いことが分かった。売上高と費用の関係を示したグラフを、図1に示す。図1のグラフ①～④のうち、R社に該当するものを、解答群の中から選べ。

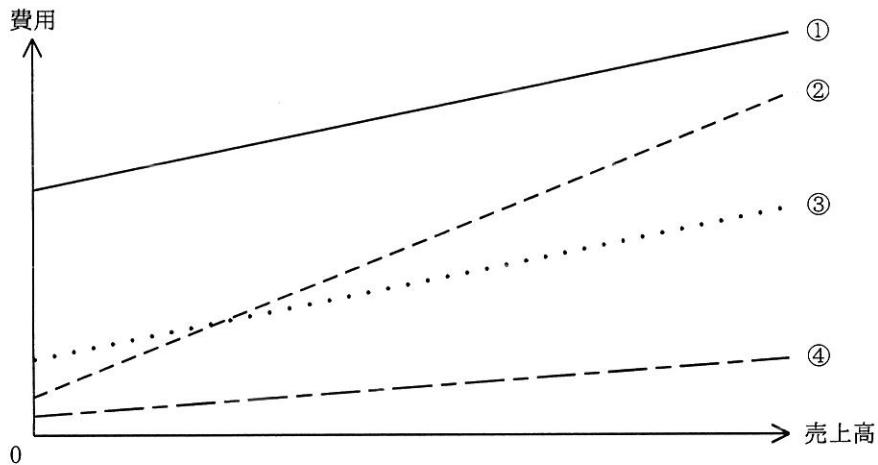


図1 4社の売上高と費用の関係を示したグラフ

解答群

ア ①

イ ②

ウ ③

エ ④

設問3 R社は、変動費率を下げる取組みを開始した。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。

R社は、固定費を変えずに変動費率だけを下げることによって、損益分岐点
売上高を f こととした。そのために、g 取組みを開始した。

fに関する解答群

ア 上げる

イ 下げる

ウ 0にする

gに関する解答群

ア 原材料の単価を下げる

イ 社員の給与を上げる

ウ 販売数を増やす

エ 販売単価を下げる

次の問8は必須問題です。必ず解答してください。

問8 次のプログラムの説明及びプログラムを読んで、設問1, 2に答えよ。

ヒープの性質を利用して、データを昇順に整列するアルゴリズムを考える。ヒープは二分木であり、本問では、親は一つ又は二つの子をもち、親の値は子の値よりも常に大きいか等しいという性質をもつものとする。ヒープの例を図1に示す。図1において、丸は節を、丸の中の数値は各節が保持する値を表す。子をもつ節を、その子に対する親と呼ぶ。親をもたない節を根と呼び、根は最大の値をもつ。

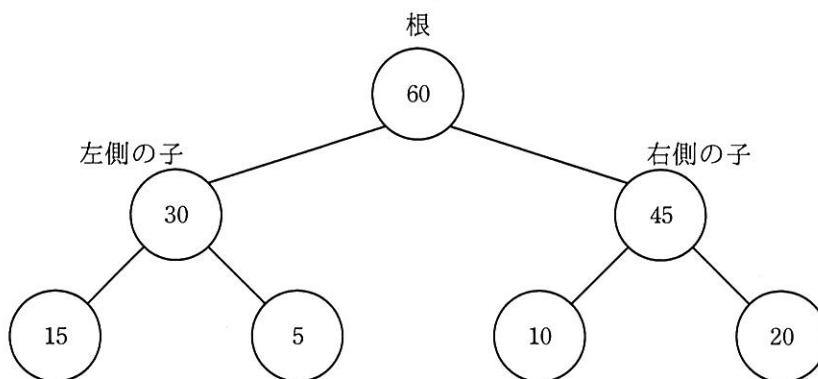
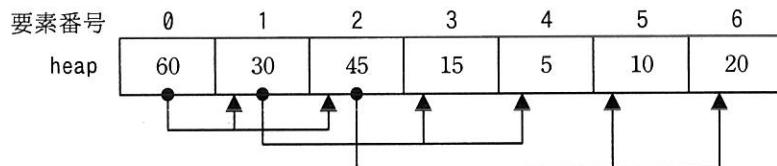


図1 ヒープの例

[プログラム1の説明]

- (1) 配列の要素番号は、0から始まる。
- (2) 副プログラム makeHeap は、整数型の1次元配列 data に格納されている hnum 個 ($hnum > 0$) のデータを、次の①～③の規則で整数型の1次元配列 heap に格納して、ヒープを配列で実現する。この状態を、“配列 heap は、ヒープの性質を満たしている”という。
 - ① 配列要素 $heap[i]$ ($i = 0, 1, 2, \dots$) は、節に対応する。配列要素 $heap[i]$ には、節が保持する値を格納する。
 - ② 配列要素 $heap[0]$ は、根に対応する。
 - ③ 配列要素 $heap[i]$ ($i = 0, 1, 2, \dots$) に対応する節の左側の子は配列要素 $heap[2 \times i + 1]$ に対応し、右側の子は配列要素 $heap[2 \times i + 2]$ に対応する。子が一つの場合、左側の子として扱う。

(3) 図 1 のヒープの例に対応した配列 `heap` の内容を、図 2 に示す。



注記 矢印 $\bullet \rightarrow$ は、始点、終点の二つの配列要素に対応する節が、親子関係にあることを表す。

図 2 図 1 のヒープの例に対応した配列 `heap` の内容

(4) 親の要素番号と子の要素番号を関係付ける三つの関数がある。

① 整数型 : `lchild` (整数型 : `i`)

要素番号 `i` の配列要素に対応する節の左側の子の配列要素の要素番号 $2 \times i + 1$ を計算して返却する。

② 整数型 : `rchild` (整数型 : `i`)

要素番号 `i` の配列要素に対応する節の右側の子の配列要素の要素番号 $2 \times i + 2$ を計算して返却する。

③ 整数型 : `parent` (整数型 : `i`)

要素番号 `i` の配列要素に対応する節の親の配列要素の要素番号 $(i - 1) \div 2$ (小数点以下切捨て) を計算して返却する。

(5) 副プログラム `swap` は、二つの配列要素に格納されている値を交換する。

(6) 副プログラム `makeHeap` の引数の仕様を表 1 に、副プログラム `swap` の引数の仕様を表 2 に示す。

表 1 副プログラム `makeHeap` の引数の仕様

引数	データ型	入出力	説明
<code>data[]</code>	整数型	入力	データが格納されている 1 次元配列
<code>heap[]</code>	整数型	出力	ヒープの性質を満たすようにデータを格納する 1 次元配列
<code>hnum</code>	整数型	入力	データの個数

表2 副プログラム swap の引数の仕様

引数	データ型	入出力	説明
heap[]	整数型	入力／出力	交換対象のデータが格納されている1次元配列
i	整数型	入力	交換対象の要素番号
j	整数型	入力	交換対象の要素番号

[プログラム1]

○副プログラム: makeHeap(整数型: data[], 整数型: heap[], 整数型: hnum)

○整数型: i, k

■ i: 0, i < hnum, 1
 · heap[i] ← data[i] /* heap にデータを追加 */
 · k ← i

■ k > 0
 a
 · swap(heap, k, b)
 · k ← parent(k)
 · break /* 内側の繰返し処理から抜ける */

○副プログラム: swap(整数型: heap[], 整数型: i, 整数型: j)

○整数型: tmp

· tmp ← heap[i]
 · heap[i] ← heap[j]
 · heap[j] ← tmp

設問1 プログラム1中の [] に入る正しい答えを、解答群の中から選べ。

aに関する解答群

ア heap[k] > heap[lchild(k)]

イ heap[k] > heap[parent(k)]

ウ heap[k] > heap[rchild(k)]

エ heap[k] < heap[lchild(k)]

オ heap[k] < heap[parent(k)]

カ heap[k] < heap[rchild(k)]

bに関する解答群

ア heap[hnum-1]

イ heap[k]

ウ parent(hnum-1)

エ parent(k)

設問2 [プログラム2の動作] の記述中の に入る正しい答えを、解答群の中から選べ。

[プログラム2の説明]

- (1) 副プログラム `heapSort` は、最初に副プログラム `makeHeap` を使って、配列 `heap` にデータを格納する。配列 `heap` は、整列対象領域と整列済みデータ領域に分かれている（図3参照）。`last` は、整列対象領域の最後の配列要素の要素番号を示している。最初は、配列 `heap` 全体が整列対象領域であり、このとき `last` の値は `hnum-1` である。

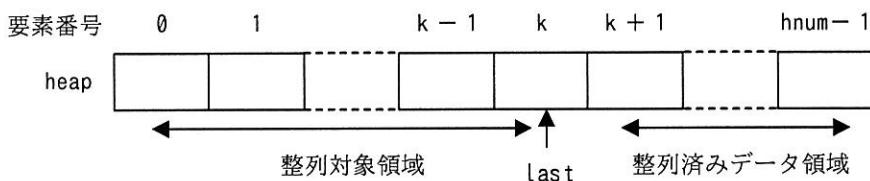


図3 配列 `heap` における整列対象領域と整列済みデータ領域

- (2) 整列対象領域がヒープの性質を満たすとき、配列要素 `heap[0]` の値は、この領域での最大の値となっている。そこで、配列要素 `heap[0]` の値と配列要素 `heap[last]` の値を交換し、`last` の値を 1 減らして、整列対象領域の範囲を狭め、整列済みデータ領域を広げる。値の交換によって、整列対象領域はヒープの性質を満たさなくなるので、副プログラム `downHeap` を使って、整列対象領域のデータがヒープの性質を満たすように再構成する。これを繰り返すことによって、整列済みデータ領域には昇順に整列されたデータが格納されることになる。
- (3) 副プログラム `heapSort` の引数の仕様を表3に、副プログラム `heapSort` で使用する副プログラム `downHeap` の引数の仕様を表4に示す。

表3 副プログラム heapSort の引数の仕様

引数	データ型	入出力	説明
data[]	整数型	入力	整列対象のデータが格納されている1次元配列
heap[]	整数型	出力	整列済みのデータを格納する1次元配列
hnum	整数型	入力	データの個数

表4 副プログラム downHeap の引数の仕様

引数	データ型	入出力	説明
heap[]	整数型	入力／出力	整列対象のデータを格納する1次元配列
hlast	整数型	入力	整列対象領域の最後の要素番号

[プログラム2]

(行番号)

```

1  ○副プログラム: heapSort( 整数型: data[], 整数型: heap[], 整数型: hnum )
2  ○整数型: last
3  · makeHeap(data, heap, hnum)
4  ■ last: hnum-1, last > 0, -1           ← α
5  · swap(heap, 0, last)                  /* heap[0]と heap[last]の値を交換 */
6  · downHeap(heap, last-1)               /* heap を再構成 */
7

```

(行番号)

```

1  ○副プログラム: downHeap( 整数型: heap[], 整数型: hlast )
2  ○整数型: n, tmp
3  · n ← 0
4  ■ lchild(n) ≤ hlast
5  · tmp ← lchild(n)
6  ▲ rchild(n) ≤ hlast
7  ▲ heap[tmp] ≤ heap[rchild(n)]
8  · tmp ← rchild(n)
9
10
11 ▲ heap[tmp] > heap[n]
12 · swap(heap, n, tmp)
13 · return                                /* downHeap から抜ける */
14
15 · n ← tmp
16

```

[プログラム 2 の動作]

副プログラム `heapSort` の行番号 3 の実行が終了した直後の α における配列 `heap` の内容は、図 2 のとおりであった。このとき、副プログラム `heapSort` の行番号 4 から行番号 7 までの 1 回目の繰返し処理について考える。

副プログラム `heapSort` の行番号 5 の副プログラム `swap` の実行が終了した直後の配列要素 `heap[0]` の値は、c となる。このため、配列 `heap` の要素番号 0 から `hnum-2` までのデータは、根に対応する配列要素 `heap[0]` が最大の値をもつというヒープの性質を満たさなくなる。

副プログラム `heapSort` の行番号 6 で呼び出している副プログラム `downHeap` は、配列 `heap` の整列対象領域の要素番号 0 から `hlast` までのデータがヒープの性質を満たすように、その領域のデータを次の手順で再構成する。

- (1) 配列要素の値の大きさを比較する際に使用する要素番号を `n` とし、`n` の初期値を 0 とする。
- (2) 要素番号 `n` の配列要素に対応する節の左側の子の要素番号を `tmp` に代入する。要素番号 `n` の子が二つあり (`rchild(n) ≤ hlast`)、右側の子の値が左側の子の値d、右側の子の要素番号を `tmp` に代入する。
- (3) 子に対応する配列要素 `heap[tmp]` の値と、その親に対応する配列要素 `heap[n]` の値とを比較し、配列要素 `heap[tmp]` の値が大きければ、配列要素 `heap[n]` の値と配列要素 `heap[tmp]` の値を交換し、`tmp` を次の `n` として(2)に戻る。ここで、副プログラム `downHeap` の行番号 15 において最初に `n` に代入する `tmp` の値は、e である。

c に関する解答群

ア 5 イ 10 ウ 15 エ 20

d に関する解答群

ア 以下のときには	イ 以上のときには
ウ よりも大きいときには	エ よりも小さいときには

e に関する解答群

ア 1	イ 2	ウ 3
エ 4	オ 5	カ 6

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。

簡易集計プログラムである。このプログラムを用いると、例えば、図1の入力ファイルから、品番ごとのレコード件数と金額の合計を求めて、図2の集計ファイルを得ることができる。

プログラムは、キー項目及び数値項目の、開始桁位置及び桁数を引数で受け取り、キー項目で整列済みのレコードを入力ファイルから読み込み、キー項目の値ごとに、その件数と数値項目の値の合計を求め、集計ファイルにレコードとして書き出す。ここで、数値項目には整数の値が入る。

年月日	時分秒	品番	数量	金額
20180410	112610	FE-111	0001	000100
20180410	154358	FE-111	0001	000100
20180410	123820	FE-222	0002	000400
20180410	153249	FE-333	0001	000300
20180410	135044	FE-333	0001	000300
20180410	152859	FE-333	0001	000300
20180410	131923	FE-444	0002	000800
20180410	123907	FE-444	0001	000400

注記 見出しの行はレコードに含まれない。

改行文字は表示していない。

破線は項目の区切りを表す。

図1 入力ファイルのレコード例

品番	件数	合計金額
FE-111	2	200
FE-222	1	400
FE-333	3	900
FE-444	2	1200

注記 見出しの行はレコードに含まれない。

改行文字は表示していない。

破線は項目の区切りを表す。

図2 集計ファイルのレコード例

[プログラム1の説明]

- (1) 入力ファイルは、固定長レコードの並びから成る。レコードは、1,000文字以下の1バイト文字の並びであり、最後の文字の後には改行文字が付いている。ファイル名は、引数 `dataFile` で指定する。
- (2) レコード中のキー項目の開始桁位置及び桁数は、それぞれ引数 `keyPos` 及び `keyLen` で指定する。また、数値項目の開始桁位置及び桁数は、それぞれ引数

`valuePos` 及び `valueLen` で指定する。開始桁位置は、レコードの先頭文字の桁位置を 0 として数える。キー項目及び数値項目の桁数は、いずれも 9 桁以下とする。

- (3) 入力レコードは、キー項目の昇順に整列されている。
- (4) 集計ファイルにレコードとして、キー項目の値、キー項目ごとの件数及び数値項目の値の合計を各 9 桁分の領域に右詰めで出力し、各項目の直前に 1 個の空白文字を出力する。レコードの最後の文字の後には改行文字を付ける。ファイル名は、引数 `listFile` で指定する。
- (5) 引数及び入力レコードの内容に誤りはないものとする。また、数値項目の値の合計は 9 桁以下であり、算術演算であふれば起きないものとする。
- (6) プログラム中で使用しているライブラリ関数（一部）の概要は、次のとおりである。

- ・ `atol(s)`: 文字列 `s` が表す数値を `long` 型の表現に変換した値を返す。
- ・ `fgets(s, m, f)`: ストリーム `f` から文字の列（改行文字まで、最大 $m - 1$ 文字）を読み取り、配列 `s` に格納し、`s` を返す。ストリームの終わりに達した場合は `NULL` を返す。
- ・ `strcmp(s1, s2)`: 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。
- ・ `strcpy(s1, s2)`: 文字列 `s2` を文字列 `s1` に複写する。
- ・ `strncpy(s1, s2, n)`: 文字列 `s2` の先頭から `n` 個の文字を文字列 `s1` に複写し、文字列 `s1` の値を返す。

[プログラム 1]

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

#define recSize 1002 /* 入力レコードの最大文字数 + 2 ('\'n', '\0') */

void outList(char *dataFile, char *listFile,
            int keyPos, int keyLen, int valuePos, int valueLen) {
    a *inFile, *outFile;
    char inBuf[recSize], inKey[10], key[10], temp[10];
    long count, inValue, value;
    char format[] = "%9s %9ld %9ld\n";
```

```

key[keyLen] = '\0';
inKey[keyLen] = '\0';
temp[valueLen] = '\0';
outFile = fopen(listFile, "w");
inFile = fopen(dataFile, "r");

if (fgets(inBuf, recSize, inFile) != NULL) {
    strncpy(key, inBuf + keyPos, keyLen);
    count = 1;
    value = atol(strncpy(temp, inBuf + valuePos, valueLen));
    while (fgets(inBuf, recSize, inFile) != NULL) {
        strncpy(inKey, inBuf + keyPos, keyLen);
        inValue = atol(strncpy(temp, inBuf + valuePos, valueLen));
        if (strcmp(key, inKey) != 0) {
            fprintf(outFile, format, key, count, value);
            count = 1;
            strcpy(key, inKey);
            value = inValue;
        }
        else {
            b
        }
    }
    c
}
fclose(inFile);
fclose(outFile);
}

```

設問 1 プログラム 1 中の に入る適切な答えを、 解答群の中から選べ。

a に関する解答群

ア char

イ FILE

ウ file

エ int

b, c に関する解答群

- ア `fprintf(outFile, format, inKey, count, inValue);`
- イ `fprintf(outFile, format, key, count, value);`
- ウ `count++;`
`value += inValue;`
- エ `count++;`
`value += inValue;`
`fprintf(outFile, format, key, count, value);`
- オ `count--;`
`fprintf(outFile, format, inKey, count, inValue);`

設問 2 次の記述中の [] に入る適切な答えを、解答群の中から選べ。ここで、プログラム 2 中の [a] には、設問 1 の正しい答えが入っているものとする。

図 3 に示す、時分秒のうちの時をキー項目として昇順に整列済みのレコードを入力ファイルから読み込み、時間帯（0 時台, 1 時台, …, 23 時台）ごとの件数と金額の合計（合計金額）を求め、時、件数、合計金額と合計金額を表す棒グラフを印字する。この処理を、次の手順で行う。

- (1) プログラム 1 を利用して、図 3 の入力ファイルから、図 4 の集計ファイルを得る。
- (2) プログラム 2 を利用して、図 4 の集計ファイルから、図 5 の印字結果を得る。

年月日	時	分	秒	品番	数量	金額
20180410	11	26	10	FE-111	0001	000100
20180410	12	38	20	FE-222	0002	000400
20180410	12	39	07	FE-444	0001	000400
20180410	13	19	23	FE-444	0002	000800
20180410	13	50	44	FE-333	0001	000300
20180410	15	28	59	FE-333	0001	000300
20180410	15	32	49	FE-333	0001	000300
20180410	15	43	58	FE-111	0001	000100

注記 見出しの行はレコードに含まれない。
改行文字は表示していない。
破線は項目の区切りを表す。

図 3 入力ファイルのレコード例

時	件数	合計金額
11	1	100
12	2	800
13	2	1100
15	3	700

注記 見出しの行はレコードに含まれない。
改行文字は表示していない。

図 4 集計ファイルのレコード例

時	件数	合計金額	
11	1	100	**
12	2	800	*****
13	2	1100	*****
15	3	700	*****

注記 見出しの行は印字結果に含まれない。

図 5 印字結果の例

[プログラム 2 の説明]

- (1) プログラム 1 で書き出した集計ファイルからレコードを読み込む。ファイル名は、引数 `listFile` で指定する。
- (2) 読み込んだ各レコードに、そのレコードの合計金額の値 `value` に応じた長さのグラフを追加して印字する。集計ファイル中の合計金額の値の最大値 `valueMax` を 25 個の “*” で表し、他の値は、 $25 \times \text{value} \div \text{valueMax}$ 個（小数点以下切捨て）の “*” で表す。ここで、集計ファイル中の合計金額の値の最大値は正であり、最小値は 0 以上であるものとする。

[プログラム 2]

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

#define listLen 32      /* 入力レコードの最大文字数 + 2 ('\'n', '\0') */

void outListG(char *listFile) {
    a *inFile;
    char inBuf[listLen];
    long value, valueMax;
    char graph[] = "*****";
    inFile = fopen(listFile, "r");
    valueMax = 0;
    while (fgets(inBuf, listLen, inFile) != NULL) {
        value = atol(inBuf + 21);
        if (value > valueMax) {
            valueMax = value;
        }
    }
    fclose(inFile);
```

```

inFile = fopen(listFile, "r");
while (fgets(inBuf, listLen, inFile) != NULL) {
    value = atol(inBuf + 21);
    printf("%.30s |%s\n",
           inBuf, &graph[25 - 25 * value / valueMax]); /* α */
    /* %.30s はレコードの先頭からの 30 衡 (\n の直前まで) を出力する */
}
fclose(inFile);
}

```

プログラム 2 は、作成途中である。図 4 の集計ファイルを用いると図 5 の印字結果が得られるが、集計ファイル中の合計金額の値によっては、コメント /* α */ を付した印字処理の実行時に問題が発生する場合がある。

[プログラム 2 の説明] の(2)にある前提“集計ファイル中の合計金額の値の最大値は正であり、最小値は 0 以上である”が満たされない場合も考慮に含め、コメント /* α */ を付した印字処理の実行時に発生し得る事象として、① 算術演算であふれが発生、② 算術演算でゼロ除算が発生、③ 配列の定義外の要素位置を参照、がある。これらの事象が発生し得る value と valueMax の値の例を、表 1 にまとめた。ここで、long 型の数値の範囲は、 $-2^{31} \sim 2^{31} - 1$ ($2^{31} = 2,147,483,648$) とする。

表 1 事象 ①～③ が発生し得る value と valueMax の値の例

発生し得る事象	value と valueMax の値の例
① 算術演算であふれ	d
② 算術演算でゼロ除算	e
③ 配列の定義外の要素位置を参照	f

d～f に関する解答群

	value の値	valueMax の値
ア	-1,000,000	0
イ	-10,000,000	10,000,000
ウ	0	10,000,000
エ	100	10,000,000
オ	10,000,000	10,000,000
カ	100,000,000	100,000,000

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 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]

Web 上でショッピングサイトを運営している P 社では、注文を受けて入金確認の処理を待っている商品の注文情報を注文ファイルに格納し、前日に入金された情報を入金ファイルに格納する。P 社では毎朝 10 時に入金確認の処理を実行し、注文ファイルの情報と入金ファイルの情報を突き合わせて結果リストを印字する。このとき、入金が確認できなかった商品の注文情報は、出荷待ちファイルに格納する。前処理として、注文ファイルには、前日にプログラムを実行した際に出力された出荷待ちファイルの情報と、前日に受けた商品の注文情報を格納しておく。

このショッピングサイトでは、代金の支払方法として、銀行振込、コンビニ払い、クレジットカード払い（以下、カード払いという）、商品着払い（以下、着払いという）のいずれかを、注文時に選択できる。注文時に選択した支払方法と入金方法が異なることはない。銀行振込及びコンビニ払いによる入金の情報が入金ファイルに格納され、入金が確認できた注文を出荷対象とする。支払方法にカード払い又は着払いが選択された場合は、注文を受けた翌日に出荷する。

(1) 注文ファイルは、図 1 に示すレコード様式の順ファイルである。

注文番号 8 桁	注文日時 14 桁	請求金額 7 桁	支払種別 1 桁
-------------	--------------	-------------	-------------

図 1 注文ファイルのレコード様式

- ① 注文番号には、注文ごとに一意に割り振られた番号が設定されている。注文番号として、99999999 が割り振られることはない。
- ② 注文日時には、西暦の年、月、日と、24 時間表記の時、分、秒が、それぞれ 4 桁、2 桁、2 桁、2 桁、2 桁で設定されている。
- ③ 請求金額には、正の整数が設定されている。

- ④ 支払種別には、注文時に選択された支払方法が、銀行振込、コンビニ払い、カード払い、着払いのそれぞれに対応する値、1, 2, 3, 4で設定されている。
- ⑤ レコードは、注文番号の昇順に整列されている。

(2) 入金ファイルは、図 2 に示すレコード様式の順ファイルである。

注文番号 8 桁	入金日時 14 桁	入金金額 7 桁	入金種別 1 桁
-------------	--------------	-------------	-------------

図 2 入金ファイルのレコード様式

- ① 注文番号には、入金が確認できた注文の注文番号が設定されている。
- ② 入金日時には、西暦の年、月、日と、24 時間表記の時、分、秒が、それぞれ 4 桁、2 桁、2 桁、2 桁、2 桁で設定されている。
- ③ 入金金額には、正の整数が設定されている。
- ④ 入金種別には、入金方法が、銀行振込、コンビニ払いのそれぞれに対応する値、1, 2 で設定されている。
- ⑤ レコードは、注文番号の昇順に整列されている。同じ注文番号のレコードが複数格納されることはない。

(3) 出荷待ちファイルは、図 3 に示すレコード様式の順ファイルである。支払方法に銀行振込又はコンビニ払いが選択された注文のうち、入金を確認できなかった商品の注文情報が格納される。

注文番号 8 桁	注文日時 14 桁	請求金額 7 桁	支払種別 1 桁
-------------	--------------	-------------	-------------

図 3 出荷待ちファイルのレコード様式

- ① 各項目の内容は、注文ファイルのレコード様式と同じである。
- (4) 結果リストの印字様式を図 4 に示す。見出しは印刷済みとする。

注文番号	結果	請求金額	入金金額	差額
99999999	XX	Z, ZZZ, ZZ9	Z, ZZZ, ZZ9	--, ---, --9
99999999	XX	Z, ZZZ, ZZ9	Z, ZZZ, ZZ9	--, ---, --9

図 4 結果リストの印字様式

① 結果リストには、出荷する注文と、エラーと判定した注文の情報を印字する。

結果欄に印字するコードと意味を表1に示す。

表1 結果欄に印字するコードと意味

コード	判定	意味
00	出荷対象	請求金額と入金金額が一致した。又は、支払方法がカード払い、着払いのいずれかであった。
10	エラー	入金ファイルの注文番号に対応する注文がなかった。
20	エラー	請求金額と入金金額が一致しなかった。

② 差額欄には、入金金額から請求金額を減算した金額を印字する。結果欄のコードが10の場合は、0を印字する。

③ 判定がエラーの場合は、出荷待ちファイルに情報は出力せず、個別に対処する。

[プログラム]

(行番号)

```

1  DATA DIVISION.
2  FILE SECTION.
3  FD ORD-FILE.
4  01 ORD-REC      PIC X(30).
5  FD RCP-FILE.
6  01 RCP-REC      PIC X(30).
7  FD WAT-FILE.
8  01 WAT-REC      PIC X(30).
9  FD PRT-FILE.
10 01 PRT-REC.
11 02 PRT-NO      PIC 9(8).
12 02              PIC X(4).
13 02 PRT-RSLT    PIC X(2).
14 02              PIC X(6).
15 02 PRT-ORD     PIC Z,ZZZ,ZZ9.
16 02              PIC X(4).
17 02 PRT-RCP     PIC Z,ZZZ,ZZ9.
18 02              PIC X(4).
19 02 PRT-DIFF    PIC --,---,--9.
20 WORKING-STORAGE SECTION.
21 01 ORD-DATA.
22 02 ORD-NO      PIC 9(8).

```

```
23      88 ORD-EOF      VALUE 99999999.  
24      02 ORD-DT.  
25          03 ORD-DATE    PIC 9(8).  
26          03 ORD-TIME    PIC 9(6).  
27      02 ORD-AMNT    PIC 9(7).  
28      02 ORD-CODE    PIC 9(1).  
29 01 RCP-DATA.  
30      02 RCP-NO      PIC 9(8).  
31      88 RCP-EOF      VALUE 99999999.  
32      02 RCP-DT.  
33          03 RCP-DATE    PIC 9(8).  
34          03 RCP-TIME    PIC 9(6).  
35      02 RCP-AMNT    PIC 9(7).  
36      02 RCP-CODE    PIC 9(1).  
37 PROCEDURE DIVISION.  
38 MAIN-PROC.  
39     OPEN INPUT  ORD-FILE RCP-FILE  
40             OUTPUT WAT-FILE PRT-FILE.  
41     INITIALIZE ORD-DATA RCP-DATA.  
42     PERFORM ORD-PROC.  
43     PERFORM RCP-PROC.  
44     MOVE SPACE TO PRT-REC.  
45     PERFORM UNTIL [ ] a [ ]  
46         INITIALIZE PRT-REC  
47         MOVE ORD-NO TO PRT-NO  
48         EVALUATE TRUE  
49             WHEN ORD-NO < RCP-NO  
50                 IF [ ] b [ ] THEN  
51                     MOVE "00" TO PRT-RSLT  
52                 ELSE  
53                     WRITE WAT-REC FROM ORD-DATA  
54                 END-IF  
55                 MOVE ORD-AMNT TO PRT-ORD  
56                 PERFORM ORD-PROC  
57             WHEN ORD-NO = RCP-NO  
58                 IF [ ] c [ ] THEN  
59                     MOVE "00" TO PRT-RSLT  
60                 ELSE  
61                     MOVE "20" TO PRT-RSLT  
62                     COMPUTE PRT-DIFF = RCP-AMNT - ORD-AMNT  
63                 END-IF  
64                 MOVE ORD-AMNT TO PRT-ORD  
65                 MOVE RCP-AMNT TO PRT-RCP
```

```

66          PERFORM ORD-PROC
67          PERFORM RCP-PROC
68      WHEN ORD-NO > RCP-NO
69          MOVE "10" TO PRT-RSLT
70          MOVE RCP-AMNT TO PRT-RCP
71          MOVE RCP-NO TO PRT-NO
72          PERFORM RCP-PROC
73      END-EVALUATE
74      IF PRT-RSLT NOT = SPACE THEN
75          d
76      END-IF
77  END-PERFORM.
78  CLOSE ORD-FILE RCP-FILE WAT-FILE PRT-FILE.
79  STOP RUN.
80  ORD-PROC.
81  IF NOT ORD-EOF THEN
82      READ ORD-FILE INTO ORD-DATA AT END SET ORD-EOF TO TRUE
83      END-READ
84  END-IF.
85  RCP-PROC.
86  IF NOT RCP-EOF THEN
87      READ RCP-FILE INTO RCP-DATA AT END SET RCP-EOF TO TRUE
88      END-READ
89  END-IF.

```

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 ORD-EOF | イ ORD-EOF AND RCP-EOF |
| ウ ORD-EOF OR RCP-EOF | エ RCP-EOF |

b, c に関する解答群

- | | |
|---------------------------|-----------------------|
| ア ORD-AMNT = RCP-AMNT | イ ORD-AMNT = ZERO |
| ウ ORD-AMNT NOT = RCP-AMNT | エ ORD-CODE = 1 OR 2 |
| オ ORD-CODE = 3 OR 4 | カ ORD-CODE = RCP-CODE |

d に関する解答群

- | | |
|-------------------------------|-------------------------|
| ア ADD 1 TO PRT-NO | イ MOVE SPACE TO PRT-REC |
| ウ SET ORD-EOF RCP-EOF TO TRUE | エ WRITE PRT-REC |

設問 2 支払方法に銀行振込又はコンビニ払いが選択された注文のうち、注文日の翌日を 1 日目として、7 日を過ぎて入金が確認できなかった注文の注文番号を表示するようにプログラムを変更する。表 2 中の [] に入る正しい答えを、解答群の中から選べ。ここで、組込み関数 INTEGER-OF-DATE は、引数に指定された YYYYMMDD 形式の日付を西暦 1601 年 1 月 1 日からの通算の日数に変換して返す。

表 2 プログラムの変更内容

処置	変更内容
行番号 36 と 37 の間に追加	01 CR-DATE-X. 02 CR-DATE PIC 9(8). 77 W-CNT PIC 9(3).
行番号 38 と 39 の間に追加	MOVE FUNCTION CURRENT-DATE TO CR-DATE-X. ¹⁾
[e] に追加	COMPUTE W-CNT = FUNCTION INTEGER-OF-DATE(CR-DATE) - FUNCTION INTEGER-OF-DATE(ORD-DATE) IF [f] THEN DISPLAY ORD-NO END-IF

注¹⁾ プログラムを実行した日付が、YYYYMMDD 形式で CR-DATE-X に格納される。

e に関する解答群

- | | |
|------------------|------------------|
| ア 行番号 51 と 52 の間 | イ 行番号 53 と 54 の間 |
| ウ 行番号 65 と 66 の間 | エ 行番号 71 と 72 の間 |

f に関する解答群

- | | |
|-------------|-----------------|
| ア W-CNT < 7 | イ W-CNT = 7 |
| ウ W-CNT > 7 | エ W-CNT NOT = 7 |

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 ~ 4 に答えよ。

[プログラムの説明]

A 君は、先輩エンジニアである B さんの指導の下で、表現式（以下、式という）を構築するためのライブラリの作成を進めている。ライブラリは API を提供する。

まず、次のインターフェース及びクラスをパッケージ `com.example.expr` に作成した。

(1) インタフェース `Expression` は、式を表す。すなわち、`Expression` を実装するクラスは、型としての式を表す。

① メソッド `evaluate` は、式を評価した結果を型 `int` で返す。

(2) クラス `Constant` は、定数を式（型 `Expression`）として表す。

① コンストラクタは、引数で与えられた型 `int` の値を表す定数を生成する。

② メソッド `evaluate` は、この定数の値を返す。

③ メソッド `toString` は、この定数の値を文字列として返す。

(3) クラス `Addition` は、加算を式（型 `Expression`）として表す。

① コンストラクタは、加算式を生成する。加算式 `a + b` において、演算子+の左側 `a` の部分を左側の式、右側 `b` の部分を右側の式とし、それぞれ引数 `left` 及び引数 `right` で指定する。いずれかの引数が `null` のときは、`NullPointerException` を投げる。

② メソッド `evaluate` は、左側の式を評価した値に右側の式を評価した値を加え、その値を返す。

③ メソッド `toString` は、この加算式を表す文字列を返す。

[プログラム 1]

```
package com.example.expr;

public interface Expression {
    public int evaluate();
}
```

[プログラム 2]

```
package com.example.expr;

public class Constant implements Expression {
    private final int value;

    public Constant(int value) {
        this.value = value;
    }

    public int evaluate() { return value; }

    public String toString() { return String.valueOf(value); }
}
```

[プログラム 3]

```
package com.example.expr;

public class Addition implements Expression {
    private final Expression left, right;

    public Addition(Expression left, Expression right) {
        if (left == null || right == null) {
            throw new NullPointerException();
        }
        this.left = left;
        this.right = right;
    }

    public int evaluate() {
        return left.evaluate() + right.evaluate();
    }

    public String toString() {
        return String.format("(%s + %s)", left, right);
    }
}
```

設問 1 A君は、クラス Constant 及び Addition を API として呼び出すテストをするために、クラス Test をパッケージ com.example.test に作成して実行したところ、図 1 の出力結果を得た。プログラム 4 中の [] に入る正しい答えを、解答群の中から選べ。

(2 + 5) = 7

図 1 メソッド main の出力結果

[プログラム 4]

```
package com.example.test;

import [a] .Addition;
import [a] .Constant;
import [a] .Expression;

public class Test {
    public static void main(String[] args) {
        Expression two = new Constant(2);
        Expression five = new Constant(5);
        Expression add = new Addition([b]);
        System.out.println([c] + " = " + [d]);
        /* α */
    }
}
```

a に関する解答群

- | | |
|---------------------------|---------------------------|
| ア com.example.expr | イ com.example.test |
| ウ static com.example.expr | エ static com.example.test |

b に関する解答群

- | | | |
|-------------|--------|-------------|
| ア 2, 5 | イ 5, 2 | ウ five |
| エ five, two | オ two | カ two, five |

c, d に関する解答群

- | | | |
|-------------------|------------------|--------------|
| ア add | イ add.evaluate() | ウ five |
| エ five.evaluate() | オ two | カ two + five |
| キ two.evaluate() | | |

設問2 次の記述中の [] に入る正しい答えを、解答群の中から選べ。

A 君は、これまでに作成したプログラムを B さんに見てもらったところ、次のアドバイスを受けた。

“乗除算などの式を実装するクラスを追加する前に、上位クラスを導入して二項演算を表すクラスを簡単に実装できるようにすべきである。また、アプリケーションが必要とする全ての二項演算をライブラリで用意するのは一般に難しいので、パッケージ外でも二項演算がそのクラスの下位クラスとして簡単に実装できるようにすべきである。”

そこで、A 君は、次の仕様の抽象クラスをライブラリに追加した。

(1) 抽象クラス `BinaryOperatorExpression` は、二項演算子を使用する式（以下、二項演算式という）を表す。

- ① コンストラクタは、二項演算式を生成する。二項演算式 `a OP b` において、二項演算子 `OP` の左側 `a` の部分を左側の式、右側 `b` の部分を右側の式とし、それぞれ引数 `left` 及び引数 `right` で指定する。いずれかの引数が `null` のときは、`NullPointerException` を投げる。
- ② メソッド `getLeft` 及び `getRight` は、それぞれ左側の式及び右側の式を返す。
- ③ 抽象メソッド `getOperator` は、この二項演算式の演算子を文字列で返す。
- ④ メソッド `toString` は、この二項演算式を表す文字列を返す。

このクラスを B さんに見てもらったところ、“ライブラリで用意された API を呼び出すアプリケーションで二項演算式を表すクラスの実装に使えるようにすることが目的なので、[] ようにするために、コンストラクタとメソッド `getLeft`, `getRight` 及び `getOperator` は `public` ではなく `protected` にすべきである” という指摘を受け、その修正をしてプログラム 5 を作成した。

〔プログラム 5〕

```
package com.example.expr;

public abstract class BinaryOperatorExpression implements Expression {
    private final Expression left, right;

    protected BinaryOperatorExpression(Expression left, Expression right) {
        if (left == null || right == null) {
            throw new NullPointerException();
        }
        this.left = left;
        this.right = right;
    }

    protected Expression getLeft() {
        return left;
    }

    protected Expression getRight() {
        return right;
    }

    protected abstract String getOperator();

    public String toString() {
        return String.format("(%s %s %s)",
            getLeft(), getOperator(), getRight());
    }
}
```

解答群

- ア 下位クラスを実装するクラスだけで使える
- イ 上位クラスの実装が下位クラスで改変されない
- ウ 通常のアプリケーションがどのパッケージからでも呼び出せる
- エ パッケージ com.example.expr 内の下位クラスだけで使える

設問3 次の記述中の [] に入る正しい答えを、解答群の中から選べ。ここで、プログラム 6 中の [a] には、設問 1 の正しい答えが入っているものとする。

プログラム 6 は、ライブラリのパッケージ外で二項演算式を実装するテストとして、減算式を表すクラス Subtraction を実装したものである。

[プログラム 6]

```
package com.example.test;

import [ a ].BinaryOperatorExpression;
import [ a ].Expression;

public class Subtraction extends BinaryOperatorExpression {
    public Subtraction(Expression left, Expression right) {
        super(left, right);
    }

    public int evaluate() {
        return getLeft().evaluate() - getRight().evaluate();
    }

    public String getOperator() {
        return "-";
    }
}
```

プログラム 6 をテストするために、プログラム 4 のメソッド main の /* α */ の行を図 2 に示す行に置き換えて実行したところ、図 1 に示した出力結果に加え、出力結果 “ [] ” を得た。

```
System.out.println(new Subtraction(add, new Subtraction(two, five)));
```

図 2 メソッド main の /* α */ の行を置き換える行

解答群

- | | | |
|-------------------------|------------------|---------------------|
| ア $((2 + 5) - (2 - 5))$ | イ $((7) - (-3))$ | ウ $(2 + 5 - 2 - 5)$ |
| エ $(2 + 5) - (2 - 5)$ | オ 0 | カ 10 |

設問4 次の記述中の に入る正しい答えを、解答群の中から選べ。

B さんから、更に次の指摘を受けた。

“オブジェクトの等価についての考慮も必要である。例えば、 を評価すると、二つのインスタンスが表す値が等しいので `true` になるべきだが、今のクラス `Constant` の実装では `false` になる。”

解答群

- ア `new Constant(9) == new Constant(9)`
- イ `new Constant(9).equals(new Constant(9))`
- ウ `new Constant(9).evaluate() == new Constant(9).evaluate()`
- エ `new Constant(9).evaluate().equals(new Constant(9).evaluate())`

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 1~3 に答えよ。

[プログラム 1 の説明]

0 ~ 65534 の整数を表す数字から成る文字列（以下、数字列という）を数値に変換する副プログラム DT0B である。

- (1) 文字列は、DC 命令の文字定数と同じ形式で主記憶に格納される。数字列 567 を格納した例を、図 1 に示す。

#0035	#0036	#0037
(1 語目)	(2 語目)	(3 語目)

図 1 数字列の格納例

- (2) 主プログラムは、数字列が格納されている領域の先頭アドレスを GR1 に、数字列の長さを GR2 に設定して、DT0B を呼ぶ。DT0B は、数値に変換して得た値を GR0 に格納して呼出し元に戻る。
- (3) 副プログラム DT0B 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]

```
DT0B    START
        RPUSH
        ADDL  GR2, GR1
        LAD   GR0, 0          ; 戻り値の初期化
LP      CPL   GR1, GR2          ; 変換終了?
        JZE   FIN
        LD    GR4, 0, GR1      ; 数字 1 文字の取出し
        SUBL  GR4, =' 0'        ; 1 桁を数値に変換
        SLL   GR0, 1          ; GR0 を 10 倍して GR4 を加算
        LD    GR5, GR0
        a
ADDL  GR0, GR5
```

```
ADDL GR0, GR4
LAD GR1, 1, GR1
JUMP LP
FIN RPOP
RET
END
```

設問 1 プログラム 1 中の に入る正しい答えを、解答群の中から選べ。

a に関する解答群

- | | | |
|-----------------|--------------|--------------|
| ア ADDL GR5, GR4 | イ SLL GR4, 1 | ウ SLL GR4, 2 |
| エ SLL GR5, 1 | オ SLL GR5, 2 | |

設問 2 文字列の先頭から数字列を探索し、順に、対応する数値を管理テーブルに格納する副プログラム GETWD を、 DTOB を使用して作成した。プログラム 2 中の に入る正しい答えを、解答群の中から選べ。

[プログラム 2 の説明]

- (1) 文字列は、一つ以上の空白文字で区切られた任意の個数の数字列を含み、最後はピリオドで終わる。最後の数字列とピリオドの間、又は文字列の先頭に一つ以上の空白文字があってもよい。文字列の例を、図 2 に示す。

△△1234△56789△△△9876△.

注記 空白文字は “△” と表記する。

図 2 文字列の例

- (2) 管理テーブルには、文字列中に数字列が現れるごとに、順に 1 語から成る要素を追加し、数字列を数値に変換して得た値を格納する。数字列の探索が終了したとき、管理テーブルの最後に 1 語から成る要素を追加し、数値の終わりを示す印として -1 を格納する。図 2 の文字列を GETWD で処理して得た管理テーブルを、図 3 に示す。

1234	56789	9876	-1
------	-------	------	----

注記 数値は、10進数で表記している。

図3 GETWD で処理して得た管理テーブル

- (3) 主プログラムは、文字列が格納されている領域の先頭アドレスを GR1 に、管理テーブルの先頭アドレスを GR2 に設定して、GETWD を呼ぶ。
- (4) 副プログラム GETWD 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 2]

```

GETWD  START
        RPUSH
        LD    GR6, GR1
        LD    GR7, GR2
        LD    GR3, =-1      ; 数字列の処理状態フラグの初期化
        LAD   GR6, -1, GR6
LP      LAD   GR6, 1, GR6
        LD    GR4, 0, GR6      ; 1 文字の取り出し
        CPL   GR4, ='.'
        JZE   FIN
        CPL   GR4, =' '
        b
CALL   SETWD
JUMP   LP
NUM    LD    GR3, GR3      ; 数字列の処理中 ?
        JZE   LP
        LD    GR3, =0      ; 次の数字列の処理開始
        LD    GR1, GR6      ; 数字列の先頭アドレスを退避
        JUMP  LP
FIN     CALL  SETWD
        LD    GR2, =-1
        ST    GR2, 0, GR7      ; 数値の終わりを示す印を格納
        RPOP
        RET
SETWD  LD    GR3, GR3
        c
        LD    GR2, GR6
        SUBL  GR2, GR1
        CALL  DTOB      ; 数字列を数値に変換

```

```

ST    GR0, 0, GR7
LD    GR3, =-1      ; 数字列の処理中状態を解除
    d
FIN2   RET
END

```

b に関する解答群

ア JNZ FIN	イ JNZ LP	ウ JNZ NUM
エ JZE FIN	オ JZE LP	カ JZE NUM

c に関する解答群

ア JNZ FIN	イ JNZ FIN2	ウ JNZ NUM
エ JZE FIN	オ JZE FIN2	カ JZE NUM

d に関する解答群

ア LAD GR6, 1, GR1	イ LAD GR6, 1, GR6	ウ LAD GR7, 1, GR2
エ LAD GR7, 1, GR7	オ LD GR6, GR1	カ LD GR7, GR2

設問3 GETWD を使用して、二つの整数の積を求める副プログラム MULT を作成した。

プログラム 3 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 3 の説明]

- (1) MULT は、数字列を二つだけ含む、[プログラム 2 の説明] の (1) で示した形式の文字列について、最初の数字列に対応する数値を被乗数とし、2 番目の数字列に対応する数値を乗数として乗算を行う。乗算においてあふれは発生しないものとする。
- (2) 主プログラムは、文字列が格納されている領域の先頭アドレスを GR1 に設定して、MULT を呼ぶ。MULT は、演算結果を GR0 に格納して呼出し元に戻る。
- (3) 副プログラム MULT 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 3]

```

MULT    START
RPUSH
LAD    GR2, CTBL
CALL   GETWD
LD     GR4, 0, GR2      ; GR4 ← 被乗数
LD     GR5, 1, GR2      ; GR5 ← 乗数
LD     GR0, =0
LD     GR5, GR5
LP      e
LD     GR3, GR5
AND   GR3, =#0001      ; 乗数の最下位ビットのチェック
JZE   NEXT
ADDL  GR0, GR4
NEXT   SLL  GR4, 1      ; 被乗数を 1 ビット左論理シフト
f
JUMP  LP
FIN    RPOP
RET
CTBL   DS   3          ; GETWD 用管理テーブル
END

```

e に関する解答群

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

アセンブリ

f に関する解答群

ア ADDL GR0, GR5	イ ADDL GR5, GR4	ウ LD GR5, GR4
エ SLL GR5, 1	オ SRL GR5, 1	

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, 2 に答えよ。

(表計算の説明)

Z 社は現在、表計算ソフトを用いた会議室の予約システムを作成中である。会議室は 15 室あり、それぞれ一意の会議室番号が振られている。また、収容可能人数（以下、定員という）は同一ではなく、スクリーンがある会議室とない会議室がある。会議室の利用は、9 時から 21 時まで可能である。予約システムは、ワークシート“会議室選定”とワークシート“予約リスト”で構成される。

(ワークシート：会議室選定)

はじめに、ワークシート“会議室選定”を作成した。ワークシート“会議室選定”的例を、図 1 に示す。

	A	B	C	D	E
1	会議室番号	定員	スクリーン	使用可否	収容率
2	101	15	有	可	0.40
3	102	5	無	可	0.00
4	103	8	有	否	0.00
5	201	10	有	可	0.60
:	:	:	:	:	:
14	501	20	無	可	0.00
15	502	5	有	可	0.00
16	503	50	有	可	0.12
17					
18	利用日	開始時刻	終了時刻	利用人数	スクリーン
19	20180415	1300	1530	6	要
20					
21	推奨会議室				
22	201				

図 1 ワークシート“会議室選定”の例

- (1) セル A2～A16 には、会議室番号が入力されている。セル B2～B16 には、会議室の定員が入力されている。セル C2～C16 には、スクリーンがある会議室であれば“有”が、そうでなければ“無”が入力されている。
- (2) 予約したい会議室の条件を、セル A19～E19 に入力する。セル A19 に会議室の利用日を、セル B19 に開始時刻を、セル C19 に終了時刻を入力する。セル D19 には、利用人数を入力する。セル E19 には、スクリーンの利用を希望する場合は“要”を、そうでない場合は“不要”を入力する。終了時刻は開始時刻よりも後でなければならない。日付は YYYYMMDD、時刻は hhmm の形式で表現される整数値である。
- (3) (2)で条件を入力した後、ワークシート“会議室選定”に格納されているマクロ SelectRoom を実行すると、ワークシート“予約リスト”に格納されている予約の一覧を参照し、会議室ごとに、条件に指定した利用日の開始時刻から終了時刻までの間（開始時刻及び終了時刻を含まない）に他の予約が入っていない場合は“可”を、そうでなければ“否”を、セル D2～D16 に格納する。
- (4) セル E2～E16 には、対応する会議室の収容率を求める式が入力されている。ここで収容率は、次の式で求める。

$$\text{収容率} = \frac{\text{利用人数}}{\text{定員}}$$

ただし、利用人数が定員を超える場合、スクリーンの利用を希望しているのに会議室にスクリーンがない場合、又は当該行の列 D の値が“否”的な場合は、収容率は 0 にする。

- (5) セル A22 には、推奨会議室の会議室番号を表示する式が入力されている。ここで、推奨会議室とは、収容率が最も大きな正の値となる会議室である。ただし、この収容率が等しい会議室が複数あるとき、それらの中で表の最も上に位置する会議室番号を表示する。また、条件を満たす会議室がないときには、“なし”を表示する。

[ワークシート：予約リスト]

予約の一覧は、ワークシート“予約リスト”に格納されている。ワークシート“予約リスト”的例を、図 2 に示す。

	A	B	C	D	E	F	G
1	予約 ID	利用日	開始時刻	終了時刻	利用人数	スクリーン	会議室番号
2	613	20180415	1300	1530	6	要	401
3	574	20180415	1300	1530	6	要	103
4	588	20180415	1300	1500	6	要	301
5	565	20180415	900	1100	10	要	201
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
520	893	20180501	900	1000	5	不要	102
521	924	20180501	1000	1100	30	不要	503
522							
⋮							
10000							

図2 ワークシート“予約リスト”の例

- (1) 1件の予約のデータが1行に入力されている。列Aには、予約ごとに一意となる予約IDが格納されている。
- (2) 列Bには利用日、列Cには開始時刻、列Dには終了時刻、列Eには利用人数が格納されている。日付はYYYYMMDD、時刻はhhmmの形式で表現される整数値である。
- (3) 列Fには、スクリーンの利用を希望する場合は“要”が、そうでない場合は“不要”が格納されている。
- (4) 列Gには、予約された会議室番号が格納されている。
- (5) 予約のデータは、利用日の昇順に常に整列されている。
- (6) 新しい予約のデータを登録するには、ワークシート“予約リスト”に格納されているマクロRegisterを実行する。また、予約総数は9,998件までとし、データの最終行よりも下の行の列A～Gの各セルには空値が格納されている。
- (7) 会議室の予約は9時から21時まで行うことができる。21時を過ぎたら直ちに、ワークシート“予約リスト”に格納されているマクロUpdateを実行する。マクロUpdateは、利用が終了した予約のデータを予約の一覧から消去するために、その予約のデータの入った行より下の行の予約のデータを上方向に詰める。
- (8) マクロRegister、Update、SelectRoomは、いずれもその実行中に他のマクロが実行されることはない。

設問1 ワークシート“会議室選定”に関する次の記述中の [] に入る正しい答えを、解答群の中から選べ。

(1) 次の式をセルE2に入力し、セルE3～E16に複写する。

IF(論理和([] a), 0, D\$19 / B2)

(2) 次の式を、セルA22に入力する。

IF([] b, 'なし', [] c)

aに関する解答群

- ア 論理積(E\$19='要', C2='有'), D2='否', D\$19 > B2
- イ 論理積(E\$19='要', C2='無'), D2='否', D\$19 > B2
- ウ 論理積(E\$19='要', C2='有', D2='否'), D\$19 > B2
- エ 論理積(E\$19='要', C2='無', D2='否'), D\$19 > B2
- オ 論理積(E\$19='不要', C2='有'), D2='否', D\$19 > B2
- カ 論理積(E\$19='不要', C2='無'), D2='否', D\$19 > B2
- キ 論理積(E\$19='不要', C2='有', D2='否'), D\$19 > B2
- ク 論理積(E\$19='不要', C2='無', D2='否'), D\$19 > B2

bに関する解答群

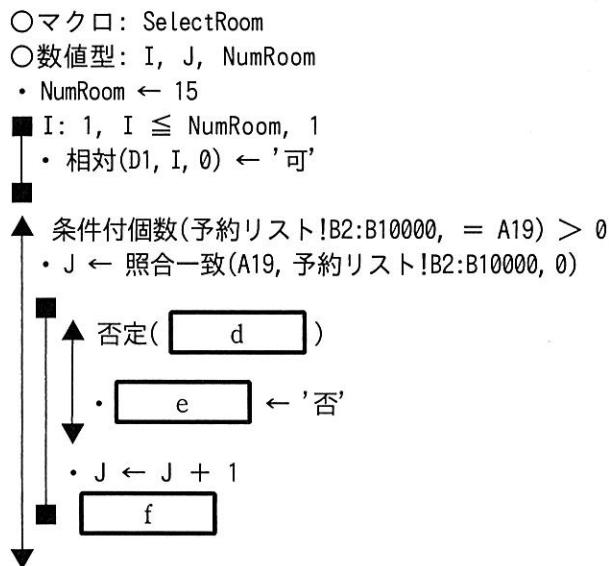
- | | |
|------------------|------------------|
| ア 最小(E2:E16) = 0 | イ 最小(E2:E16) = 1 |
| ウ 最小(E2:E16) > 0 | エ 最小(E2:E16) < 1 |
| オ 最大(E2:E16) = 0 | カ 最大(E2:E16) = 1 |
| キ 最大(E2:E16) > 0 | ク 最大(E2:E16) < 1 |

cに関する解答群

- ア 照合一致(最小(E2:E16), E2:E16, 0)
- イ 照合一致(最大(E2:E16), E2:E16, 0)
- ウ 照合検索(最小(E2:E16), E2:E16, A2:A16)
- エ 照合検索(最大(E2:E16), E2:E16, A2:A16)
- オ 表引き(A2:A16, 順位(最小(E2:E16), E2:E16, 0), 1)
- カ 表引き(A2:A16, 順位(最大(E2:E16), E2:E16, 0), 1)

設問2 マクロ SelectRoom 中の に入る正しい答えを、解答群の中から選べ。

[マクロ : SelectRoom]



dに関する解答群

- ア 論理積(表引き(予約リスト!C2:C10000, J, 1) ≥ C19,
表引き(予約リスト!D2:D10000, J, 1) ≤ B19)
- イ 論理積(表引き(予約リスト!C2:C10000, J, 1) ≤ C19,
表引き(予約リスト!D2:D10000, J, 1) ≥ B19)
- ウ 論理積(表引き(予約リスト!D2:D10000, J, 1) ≥ C19,
表引き(予約リスト!C2:C10000, J, 1) ≤ B19)
- エ 論理積(表引き(予約リスト!D2:D10000, J, 1) ≤ C19,
表引き(予約リスト!C2:C10000, J, 1) ≥ B19)
- オ 論理和(表引き(予約リスト!C2:C10000, J, 1) ≥ C19,
表引き(予約リスト!D2:D10000, J, 1) ≤ B19)
- カ 論理和(表引き(予約リスト!C2:C10000, J, 1) ≤ C19,
表引き(予約リスト!D2:D10000, J, 1) ≥ B19)
- キ 論理和(表引き(予約リスト!D2:D10000, J, 1) ≥ C19,
表引き(予約リスト!C2:C10000, J, 1) ≤ B19)
- ク 論理和(表引き(予約リスト!D2:D10000, J, 1) ≤ C19,
表引き(予約リスト!C2:C10000, J, 1) ≥ B19)

e に関する解答群

- ア 相対(D2, J - 1, 0)
- イ 相対(D2, J, 0)
- ウ 相対(D2, 照合一致(表引き(予約リスト!G2:G10000, J, 1),
A2:A16, 0) - 1, 0)
- エ 相対(D2, 照合一致(表引き(予約リスト!G2:G10000, J, 1),
予約リスト!G2:G10000, 0) - 1, 0)
- オ 相対(D2, 照合一致(予約リスト!G2, A2:A16, 0) - 1, 0)
- カ 相対(D2, 照合一致(予約リスト!G2, A2:A16, 1) - 1, 0)

f に関する解答群

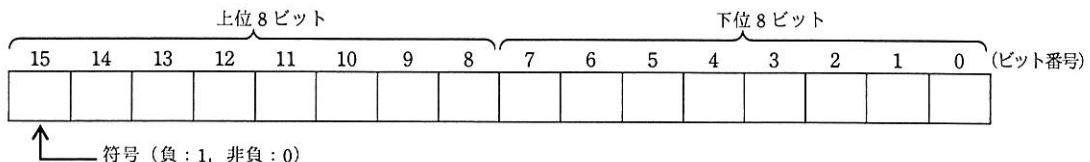
- ア 表引き(予約リスト!B2:B10000, J, 1) = A19
- イ 表引き(予約リスト!B2:B10000, J, 1) = null
- ウ 表引き(予約リスト!B2:B10000, J, 1) ≠ A19
- エ 表引き(予約リスト!B2:B10000, J, 1) ≠ null
- オ 表引き(予約リスト!C2:C10000, J, 1) < B19
- カ 表引き(予約リスト!D2:D10000, J, 1) > C19

■アセンブラー言語の仕様

1. システム COMET II 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 <hr/> r1, r2 r, adr [, x]	r1 ← (r2) r ← (実効アドレス)	○*1
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	r_1, r_2	$r_1 \leftarrow (r_1) + (r_2)$	○
		$r, \text{adr } [, x]$	$r \leftarrow (r) + (\text{実効アドレス})$	
論理加算 ADD Logical	ADDL	r_1, r_2	$r_1 \leftarrow (r_1) +_L (r_2)$	○
		$r, \text{adr } [, x]$	$r \leftarrow (r) +_L (\text{実効アドレス})$	
算術減算 SUBtract Arithmetic	SUBA	r_1, r_2	$r_1 \leftarrow (r_1) - (r_2)$	○
		$r, \text{adr } [, x]$	$r \leftarrow (r) - (\text{実効アドレス})$	
論理減算 SUBtract Logical	SUBL	r_1, r_2	$r_1 \leftarrow (r_1) -_L (r_2)$	○
		$r, \text{adr } [, x]$	$r \leftarrow (r) -_L (\text{実効アドレス})$	
論理積 AND	AND	r_1, r_2	$r_1 \leftarrow (r_1) \text{ AND } (r_2)$	○*1
		$r, \text{adr } [, x]$	$r \leftarrow (r) \text{ AND } (\text{実効アドレス})$	
論理和 OR	OR	r_1, r_2	$r_1 \leftarrow (r_1) \text{ OR } (r_2)$	○*1
		$r, \text{adr } [, x]$	$r \leftarrow (r) \text{ OR } (\text{実効アドレス})$	
排他的論理和 eXclusive OR	XOR	r_1, r_2	$r_1 \leftarrow (r_1) \text{ XOR } (r_2)$	○
		$r, \text{adr } [, x]$	$r \leftarrow (r) \text{ XOR } (\text{実効アドレス})$	

(3) 比較演算命令

算術比較 ComPare Arithmetic	CPA	r_1, r_2	(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																								
		$r, \text{adr } [, x]$																										
論理比較 ComPare Logical	CPL	r_1, r_2	<table border="1"> <thead> <tr> <th>比較結果</th> <th colspan="2">FR の値</th> </tr> <tr> <th></th> <th>SF</th> <th>ZF</th> </tr> </thead> <tbody> <tr> <td>(r1) > (r2)</td> <td>0</td> <td>0</td> </tr> <tr> <td>(r) > (実効アドレス)</td> <td></td> <td></td> </tr> <tr> <td>(r1) = (r2)</td> <td>0</td> <td>1</td> </tr> <tr> <td>(r) = (実効アドレス)</td> <td></td> <td></td> </tr> <tr> <td>(r1) < (r2)</td> <td>1</td> <td>0</td> </tr> <tr> <td>(r) < (実効アドレス)</td> <td></td> <td></td> </tr> </tbody> </table>	比較結果	FR の値			SF	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)			○*1
比較結果	FR の値																											
	SF	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)																												
$r, \text{adr } [, x]$																												

(4) シフト演算命令

算術左シフト Shift Left Arithmetic	SLA	$r, \text{adr } [, x]$	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	$r, \text{adr } [, x]$		
論理左シフト Shift Left Logical	SLL	$r, \text{adr } [, x]$	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。	○*2
論理右シフト Shift Right Logical	SRL	$r, \text{adr } [, x]$		

(5) 分岐命令

正分岐 Jump on Plus	JPL	$\text{adr } [, x]$	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	$\text{adr } [, x]$		
非零分岐 Jump on Non Zero	JNZ	$\text{adr } [, x]$		
零分岐 Jump on ZEro	JZE	$\text{adr } [, x]$		
オーバフロー分岐 Jump on Overflow	JOV	$\text{adr } [, x]$		
無条件分岐 unconditional JUMP	JUMP	$\text{adr } [, x]$	無条件に実効アドレスに分岐する。	

(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	/	?	0	_	o		

2. アセンブラー言語 CASL II の仕様

2.1 言語の仕様

- (1) CASL II は、COMET II のためのアセンブラー言語である。
- (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）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 種類のアセンブラー命令 (START, END, DS, DC), 4 種類のマクロ命令 (IN, OUT, RPUSH, RPOP) 及び機械語命令 (COMET II 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
アセンブラー命令	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 アセンブラー命令

アセンブラー命令は、アセンブラーの制御などを行う。

- (1) START [実行開始番地]

START 命令は、プログラムの先頭を定義する。

実行開始番地は、そのプログラム内で定義されたラベルで指定する。指定がある場合はその番地から、省略した場合は START 命令の次の命令から、実行を開始する。

また、この命令につけられたラベルは、他のプログラムから入口名として参照できる。

(2)	END	
-----	-----	--

END 命令は、プログラムの終わりを定義する。

(3)	DS	語数
-----	----	----

DS 命令は、指定した語数の領域を確保する。

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

(4)	DC	定数 [, 定数] ...
-----	----	---------------

DC 命令は、定数で指定したデータを（連続する）語に格納する。

定数には、10進定数、16進定数、文字定数、アドレス定数の4種類がある。

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

2.4 マクロ命令

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

(1)	IN	入力領域, 入力文字長領域
-----	----	---------------

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

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

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

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

(2)	OUT	出力領域, 出力文字長領域
-----	-----	---------------

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

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

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

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

(3)

RPUSH	
-------	--

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

(4)

RPOP	
------	--

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

2.5 機械語命令

機械語命令のオペランドは、次の形式で記述する。

r, r1, r2 GR は、記号 GR0 ~ GR7 で指定する。

x 指標レジスタとして用いる GR は、記号 GR1 ~ GR7 で指定する。

adr アドレスは、10 進定数、16 進定数、アドレス定数又はリテラルで指定する。

リテラルは、一つの 10 進定数、16 進定数又は文字定数の前に等号 (=) を付けて記述する。CASL II は、等号の後の定数をオペランドとする DC 命令を生成し、そのアドレスを adr の値とする。

2.6 その他

(1) アセンブラーによって生成される命令語や領域の相対位置は、アセンブラー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) アセンブラーは、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、アセンブラー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、読み出し、印刷、^{まい}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”的セル 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”的セル 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ます。

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ぬ。
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）	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の場合には降順で数えて、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)	<p>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。</p> <ul style="list-style-type: none"> ・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。 <p>[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。</p>
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⁾)	<p>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。</p> <p>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。</p> <p>[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。</p> <p>[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。</p>

注¹⁾ 引数として渡したセル範囲の中で、数値以外の値は処理の対象としない。

2) 引数として渡すことができる式の個数は、1以上である。

7. マクロ

(1) ワークシートとマクロ

ワークシートには複数のマクロを格納することができる。

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

[例] ○マクロ: Pro

例は、マクロProの宣言である。

(2) 変数とセル変数

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

[例] ○数値型: row, col

例は、数値型の変数row, colの宣言である。

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

セル変数の表現方法には、絶対表現と相対表現とがある。

セル変数の絶対表現は、セル番地で表す。

セル変数の相対表現は、次の書式で表す。

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

[例1] 相対(B5, 2, 3) は、セル E7 を表す変数である。

[例2] 相対(B5, -2, -1) は、セル A3 を表す変数である。

(3) 配列

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

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

[例] ○文字列型: table[100, 200]

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

(4) 宣言、注釈及び処理

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

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

[例] ○数値型: row

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

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

[メモ用紙]

[メモ用紙]

[メモ用紙]

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

退室可能時間	13:40 ~ 15:20
--------	---------------

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

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