

平成 30 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

- 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ないことがあります。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。

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)	(ウ)	(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

基本情報技術者試験 午後 問題

平成 30 年 10 月 21 日実施

ページ	行	誤	正	訂正の内容
76	Java プログラム で使用する API の説明	引数 : delimiter — element が返すシークエンスを連結する際に 間に挟む char 値のシークエンス 下から 4 行目	引数 : delimiter — elements が返すシークエンスを連結する際に 間に挟む char 値のシークエンス	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	情報セキュリティ事故と対策

●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	プロセスのスケジューリング
問 3	データベース	コンサートチケット販売サイトの関係データベースの設計及び運用
問 4	ネットワーク	ネットワークの障害分析と対策
問 5	ソフトウェア設計	購買管理システムで行う処理
問 6	プロジェクトマネジメント	プロジェクトのスケジュール作成
問 7	システム戦略	広告制作業務の現状把握と改善

●問 8 (必須問題)

問題番号	出題分野	テーマ
問 8	データ構造及びアルゴリズム	整数式の解析と計算

●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)	鉄道模型における列車の運行シミュレーション
問 10	ソフトウェア開発 (COBOL)	社内資格の保有状況の管理
問 11	ソフトウェア開発 (Java)	書式を表すひな型への置換表の適用による文書の作成
問 12	ソフトウェア開発 (アセンブラー)	日数の計算
問 13	ソフトウェア開発 (表計算)	待ち時間の状況などの分析

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。
・ 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 情報セキュリティ事故と対策に関する次の記述を読んで、設問1～3に答えよ。

自動車の販売代理店であるA社は、Webサイトで自動車のカタログ請求を受け付けている。Webサイトは、Webアプリケーションソフト（以下、Webアプリという）が稼働するWebサーバと、データベースが稼働するデータベースサーバ（以下、DBサーバという）で構成されている。WebサーバはA社のDMZに設置され、DBサーバはA社の社内LANに接続されている。Webサイトの管理はB氏が、A社の社内LANに接続されている保守用PCからアクセスして行っている。カタログ請求者は、Webブラウザからインターネット経由でHTTP over TLSによってWebサイトにアクセスする。

[カタログ請求者の情報の登録]

A社では、次の目的で、カタログ請求者の情報を保持し、利用することの同意を、カタログ請求者から得ている。

- ・情報提供や購入支援を行う。
- ・カタログ請求者が別のカタログを請求したいときなどに、登録した電子メールアドレスとパスワードを使用してログインできるようにする。

同意が得られたときは、氏名、住所、電話番号、電子メールアドレス、パスワード、購入予定期間、購入予算、希望車種などの情報を、Webアプリに入力してもらい、データベースに登録している。パスワードはハッシュ化して、それ以外の情報は平文で、データベースに格納している。A社では、カタログ請求者から要求があったときにだけ、データベースからそのカタログ請求者の情報を消去する運用としている。

[カタログ請求者への対応]

A社では、カタログ請求者へのカタログ送付後の購入支援を、データベースに登録されている情報を基に、電子メールと電話で行っている。

[情報セキュリティ事故の発生]

ある日、A 社の社員から、“A 社のカタログ請求者一覧と称する情報が、インターネットの掲示板に公開されている”と B 氏に連絡があった。公開されている情報を B 氏が確認したところ、データベースに登録されている情報の一部であったので、自社のデータベースから情報が流出したと判断して上司に報告した。B 氏は上司からの指示を受けて、Web サイトのサービスを停止し、情報が流出した原因と流出した情報の範囲を特定することにした。

[情報セキュリティ事故の原因と流出した情報の範囲]

B 氏の調査の結果、Web アプリに SQL インジェクションの脆弱性があることが分かった。そのことから B 氏は、攻撃者が①インターネット経由で SQL インジェクション攻撃を行い、データベースに登録されているカタログ請求者の情報を不正に取得したと推測した。Web サーバとデータベースではアクセスログを取得しない設定にしていたこともあり、流出した情報の範囲は特定できなかった。そこで、データベースに登録されている全ての情報が流出したこと前提に、A 社では、データベースに登録されている全てのカタログ請求者に情報の流出について連絡するとともに、対策を講じることにした。

[情報セキュリティ事故を踏まえたシステム面での対策]

B 氏は、今回の情報セキュリティ事故を踏まえたシステム面での対策案を、表 1 のようにまとめた。

表 1 情報セキュリティ事故を踏まえたシステム面での対策案

目的	対策
SQL インジェクション攻撃からの防御	<ul style="list-style-type: none">SQL 文の組立てはプレースホルダで実装する。a
情報流出リスクの低減	<ul style="list-style-type: none">b
情報流出の原因と流出した情報の範囲の特定	<ul style="list-style-type: none">c

設問1 本文中の下線①について、この攻撃の説明として適切な答えを、解答群の中から選べ。

解答群

- ア 攻撃者が、DNSに登録されているドメインの情報をインターネット経由で外部から改ざんすることによって、カタログ請求者を攻撃者のWebサイトに誘導し、カタログ請求者のWebブラウザで不正スクリプトを実行させる。
- イ 攻撃者が、インターネット経由でDBサーバに不正ログインする。
- ウ 攻撃者が、インターネット経由でWebアプリに、データベース操作の命令文を入力することによって、データベースを不正に操作する。
- エ 攻撃者が、インターネット経由で送信されている情報を盗聴する。

設問2 表1中の [] に入る対策として最も適切な答えを、解答群の中から選べ。

aに関する解答群

- ア Webアプリへの入力パラメタには、Webサーバ内のファイル名を直接指定できないようにする。
- イ Webサーバのメモリを直接操作するような命令を記述できないプログラム言語を用いて、Webアプリを作り直す。
- ウ Webページに出力する要素に対して、エスケープ処理を施す。
- エ データベース操作の命令文の組立てを文字列連結によって行う場合は、連結する文字列にエスケープ処理を施す。

bに関する解答群

- ア カタログ請求者の情報の適切な保管期間を定め、カタログ請求者の同意を得た上で、保管期間を過ぎた時点でデータベースから消去する。
- イ カタログ請求者の情報を、カタログ送付後に直ちに、データベースから消去する。
- ウ カタログ請求者へ送付する電子メールにデジタル署名を付ける。
- エ データベースに登録されている情報を定期的にバックアップする。

c に関する解答群

- ア Web サイトの管理に使用する保守用 PC は、必要なときだけ起動する。
- イ Web サーバと DB サーバにインストールするミドルウェアは、必要最低限にする。
- ウ Web サーバと DB サーバのハードディスクのデフラグメンテーションを、定期的に行う。
- エ データベースへのアクセスログを取得する。

設問 3 B 氏は上司から、表 1 にまとめた対策案だけで十分なのか検討せよとの指示を受けた。そこで、社外のセキュリティコンサルタント会社に相談したところ、“Web アプリに脆弱性がないか調査をした方がよい”と助言され、Web アプリの一部について脆弱性の調査を依頼した。その結果、クロスサイトスクリッピングの脆弱性が存在することが判明した。また、“Web アプリの他の部分にも脆弱性があることが疑われる所以、Web アプリ全体の調査を行うとともに、新たな対策を講じた方がよい”と助言された。新たな対策として適切な答えを、解答群の中から選べ。

解答群

- ア DB サーバを、Web サーバと同じく、DMZ に設置する。
- イ 不正な通信を遮断するために、WAF（Web Application Firewall）を導入する。
- ウ Web サーバを増設して冗長化した構成にする。
- エ 保守用 PC のログインパスワードには英数字及び記号を使用し、推測が難しい複雑なものを設定する。

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 プロセスのスケジューリングに関する次の記述を読んで、設問1、2に答えよ。

OSの機能の一つに、プロセスのCPUへの割当てがある。プロセスをCPUに割り当てる順序（以下、実行順序という）を決定する方式として、本問で示すラウンドロビン方式と優先度順方式を考える。プロセスが実行されるコンピュータのCPUは一つであり、CPUは一度に一つのプロセスしか実行できないものとする。

ラウンドロビン方式では、キューを用いて、複数のプロセスを一定時間（以下、タイムクウォンタムという）を限度にCPUに割り当てて実行する。ラウンドロビン方式でプロセスの実行順序を決定する例を、図1に示す。

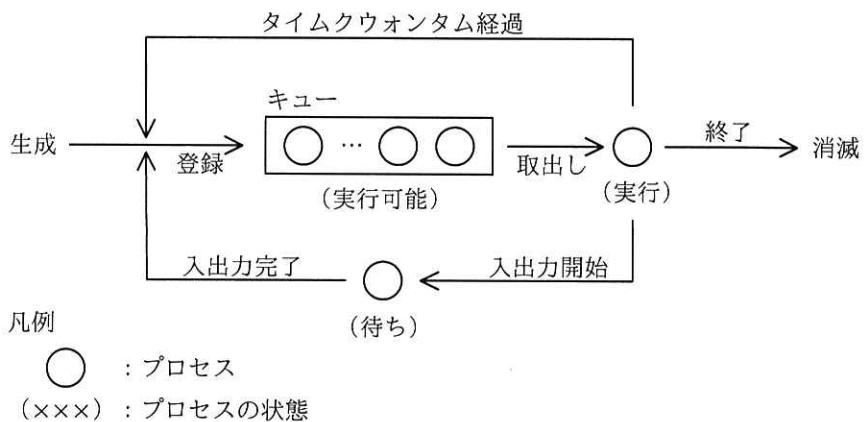


図1 ラウンドロビン方式でプロセスの実行順序を決定する例

- (1) プロセスを生成順にキューの末尾に登録する。
- (2) プロセスの実行の中止などによって、実行中のプロセスがない場合、キューの先頭からプロセスを一つ取り出してCPUに割り当てる、実行を開始する。
- (3) プロセスの実行が終了すると、そのプロセスを消滅させる。

- (4) プロセスの実行中にタイムクウォンタムが経過したら、実行を中断して、キューの末尾に登録する。
- (5) 実行中のプロセスが入出力を開始したら、実行を中断する。
- (6) プロセスの入出力が完了したら、キューの末尾に登録する。

キューに登録されているプロセスの状態を“実行可能”，実行中のプロセスの状態を“実行”，入出力の完了を待っているプロセスの状態を“待ち”と呼ぶ。ここで、実行の中止はタイムクウォンタムの経過と入出力の開始だけで行い，“待ち”への遷移は入出力の開始だけで行うものとする。また、OSによるオーバヘッドはないものとする。

設問1 図2に示す処理順序をもつプロセスXを、図1に示すラウンドロビン方式で実行する場合を考える。プロセスXの処理時間及び待ち時間を表1に示す。

表1において、処理時間とは、処理1、処理2及び処理3のそれぞれが実行を開始してから終了するまでに必要なCPUの使用時間である。待ち時間とは、入出力待ち1及び入出力待ち2のそれぞれが入出力を開始してから完了するまでに要する時間であり、その間CPUを使用しない。

タイムクウォンタムが20ミリ秒のとき、プロセスXが生成されてから消滅するまでに、図3に示す①～④の遷移が起こる回数の組合せとして正しい答えを、解答群の中から選べ。



図2 プロセスXの処理順序

表1 プロセスXの処理時間及び待ち時間

単位 ミリ秒

プロセス名	処理時間及び待ち時間				
	処理1	入出力待ち1	処理2	入出力待ち2	処理3
X	30	30	50	30	10

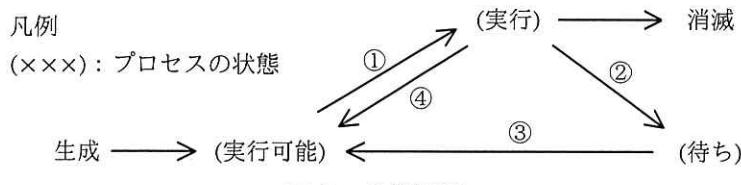


図3 状態遷移

解答群

	①	②	③	④
ア	3	2	2	0
イ	3	2	2	3
ウ	3	2	2	4
エ	6	2	2	0
オ	6	2	2	3
カ	6	2	2	4

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。

優先度順方式では、ラウンドロビン方式と同様にキューを用いて、複数のプロセスをタイムクウォンタムを限度にCPUに割り当てて実行する。この方式では、プロセスには優先度が与えられ、優先度ごとに決められたキューに登録される。優先度順方式でプロセスの実行順序を決定する例を、図4に示す。ここで、優先度は1~5の5段階であり、値の大きい方が優先度は高い。プロセスをキューから取り出すときは、プロセスが登録されているキューの中で、優先度の最も高いキューの先頭からプロセスを取り出す。

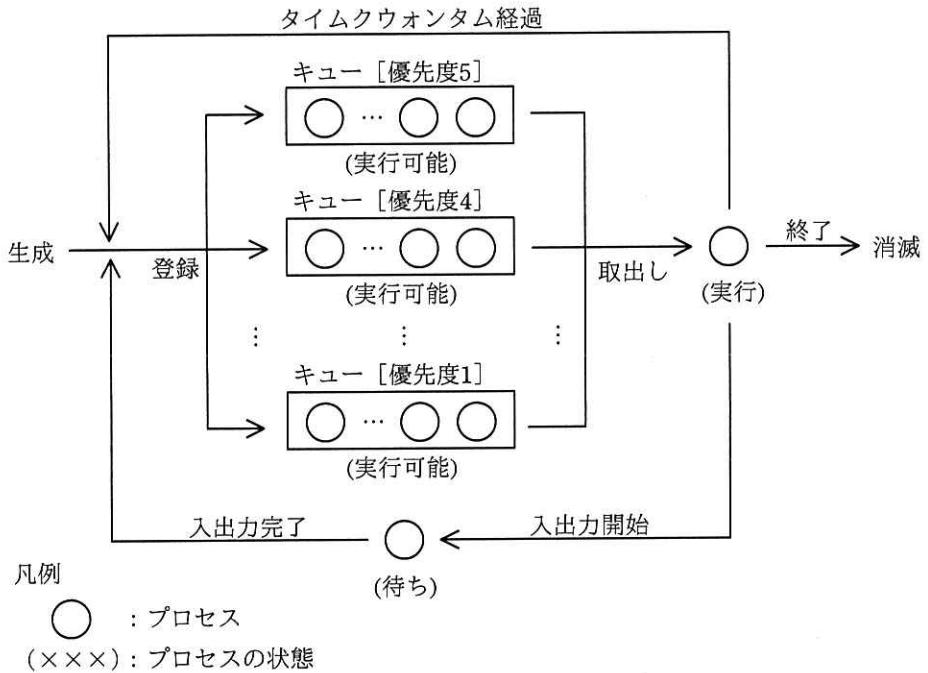


図 4 優先度順方式でプロセスの実行順序を決定する例

この方式では、プロセスの優先度を次のように与える。

- (1) プロセスが生成された場合、そのプロセスに優先度 3 を与える。
- (2) プロセスの実行中にタイムクウォンタムが経過して、実行を中断した場合、そのプロセスの現在の優先度に応じて、次のとおり優先度を与える。
 - ・優先度が 1 のとき、優先度 1 を与える。
 - ・優先度が 2~5 のとき、1 段階下げた優先度を与える。
- (3) プロセスの入出力が完了した場合、そのプロセスに優先度 5 を与える。

表 1 に示すプロセス X を、図 4 に示す優先度順方式で実行する場合を考える。タイムクウォンタムが 20 ミリ秒のとき、プロセス X が生成されてから消滅するまでの優先度の推移は、a となる。

図 4 に示す優先度順方式において，“実行”，“実行可能”及び“待ち”的プロセスが一つもないとき，表 2 に示す三つのプロセス A, B, C が順次生成される場合を考える。

- (1) 生成時刻は，プロセス A が生成された時刻からの経過時間である。
- (2) 三つのプロセス A, B, C が全て消滅するまで，他のプロセスが新たに生成されることなく，OS によるオーバヘッドもない。
- (3) 入出力装置は同時に動作が可能であり，同じ時間帯に複数のプロセスが“待ち”になっても，待ち時間は表 2 に示すとおりで変わらない。複数のプロセスの入出力が同時に完了した場合，入出力を開始してから完了するまでの時間が最も長かったプロセスを先にキューに登録する。

表 2 プロセス A, B, C の生成時刻，処理時間及び待ち時間

単位 ミリ秒

プロセス名	生成時刻	処理時間及び待ち時間				
		処理1	入出力待ち1	処理2	入出力待ち2	処理3
A	0	30	20	50	30	10
B	10	30	30	30	30	30
C	30	20	10	20	10	90

タイムクォンタムが 20 ミリ秒のとき，プロセス B の処理 1 が実行を開始する時刻は，プロセス A の生成時刻から b ミリ秒後であり，プロセス A の処理 1 が終了する時刻は，プロセス A の生成時刻から c ミリ秒後である。

a に関する解答群

- | | |
|-------------------------|-------------------------|
| ア 3 → 2 → 5 → 4 → 3 → 2 | イ 3 → 2 → 5 → 4 → 3 → 5 |
| ウ 3 → 3 → 5 → 3 → 1 | エ 3 → 4 → 5 → 4 → 3 → 5 |

b, c に関する解答群

- | | | |
|------|------|------|
| ア 10 | イ 20 | ウ 30 |
| エ 40 | オ 50 | カ 60 |
| キ 70 | ク 80 | ケ 90 |

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 コンサートチケット販売サイトの関係データベースの設計及び運用に関する次の記述を読んで、設問1～4に答えよ。

D社は、Web上で会員制のコンサートチケット販売サイトを運営している。販売サイトのシステムは販売サブシステムと席予約サブシステムから構成され、販売サブシステムで購入申込み及び決済を処理し、席予約サブシステムで座席指定を処理する。本問では、販売サブシステムだけを取り扱う。

販売サブシステムで利用しているデータベースの表構成とデータの格納例を図1に示す。下線付きの項目は主キーである。

会員表

会員ID	氏名	電子メールアドレス
K00001	情報太郎	taro@example.com
⋮	⋮	⋮

商品表

コンサートID	コンサート情報	開催日時
C00001	クリスマスコンサート 2018 in 東京 出演: Xバンド…	2018-12-24 18:00:00
⋮	⋮	⋮

商品詳細表

コンサートID	席種	価格	発売席数
C00001	S	6000	500
⋮	⋮	⋮	⋮

決済表

販売ID	決済日	決済額
H000001	2018-09-29	12000
⋮	⋮	⋮

販売表

販売ID	会員ID	コンサートID	席種	席数	販売日	販売額	決済期限日
H000001	K00001	C00001	S	2	2018-09-02	12000	2018-10-01
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

図1 販売サブシステムで利用しているデータベースの表構成とデータの格納例

[コンサートの席の説明]

- (1) コンサートの席種には、S, A 及び B がある。
- (2) 各席種の価格（常に有料）及び発売席数は、コンサートごとに異なる。

[販売サブシステムの説明]

- (1) 販売サブシステムは取り扱うコンサートの席種ごとの販売可能な席数を管理する。
- (2) 会員が購入申込みを行うと、販売サブシステムは一意な販売 ID を生成して販売表にレコードを追加する。
- (3) 会員が支払手続を行うと、決済処理として販売サブシステムは販売 ID を主キーとするレコードを決済表に追加する。ここで、決済日はレコードを追加した日とする。
- (4) 販売サブシステムは決済期限日の翌日に、決済期限日を過ぎた販売表中のレコードと販売 ID が同じレコードが決済表にない場合、その購入申込みは取り消されたものとして、バッチ処理によって決済表に当該販売 ID を主キーとするレコードを追加する。このレコードの決済日は NULL で、決済額は -1 とする。
- (5) バッチ処理は、毎夜 0~4 時の販売サイトのシステムのメンテナンス時間帯に行う。
- (6) 会員が購入を申し込んだ席数が、その時点で販売可能な席数を上回る場合には、販売サブシステムは“販売終了”と表示し、この購入申込みを受け付けない。

設問 1 データベースのデータの整合性を保つために DDL で制約をつけてある。図 1 の表構成において、列名とその列に指定する制約の正しい組合せを、解答群の中から選べ。

解答群

	表名.列名	制約
ア	決済表. 決済額	検査制約
イ	決済表. 決済日	非 NULL 制約
ウ	商品詳細表. 席種	参照制約
エ	販売表. 会員 ID	一意性制約

設問2 “販売終了”の表示判定を行うために、販売できない席数を求める必要がある。販売できない席数を出力する SQL 文の [] に入る正しい答えを、解答群の中から選べ。ここで、コンサート ID は C00001、席種は S である。a1 と a2 に入る答えは、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。

```
SELECT SUM(販売表.席数)
  FROM 販売表 [ ] 決済表 ON 販売表.販売 ID = 決済表.販売 ID
 WHERE 販売表.コンサート ID = 'C00001'
   AND 販売表.席種 = 'S'
   AND [ ] a2
```

a に関する解答群

	a1	a2
ア	INNER JOIN	決済表.決済額 = -1
イ	INNER JOIN	決済表.決済額 >= 0
ウ	LEFT OUTER JOIN	(決済表.決済額 IS NULL OR 決済表.決済額 = -1)
エ	LEFT OUTER JOIN	(決済表.決済額 IS NULL OR 決済表.決済額 >= 0)
オ	RIGHT OUTER JOIN	(決済表.決済額 IS NULL OR 決済表.決済額 = -1)
カ	RIGHT OUTER JOIN	(決済表.決済額 IS NULL OR 決済表.決済額 >= 0)

設問3 決済期限日まで残り 3 日となっても支払手続が行われていない購入申込みがある会員に、支払手続を促す電子メールを送る。この会員の氏名、電子メールアドレス及び販売 ID を出力する SQL 文の [] に入る正しい答えを、解答群の中から選べ。NOW は SQL を実行した日の日付を返すユーザ定義関数であり、DATEDIFF はともに日付である二つの引数を受け取って第 1 引数から第 2 引数を引いた日数を整数値で返すユーザ定義関数である。

```
SELECT 会員表.氏名, 会員表.電子メールアドレス, 販売表.販売 ID
```

```
[ ] b
```

bに関する解答群

ア FROM 会員表, 販売表

WHERE DATEDIFF(販売表.決済期限日, NOW()) = 3

AND 販売表.会員 ID = 会員表.会員 ID

AND 販売表.販売 ID NOT IN (SELECT 販売 ID FROM 決済表)

イ FROM 会員表, 販売表

WHERE DATEDIFF(販売表.決済期限日, NOW()) = 3

AND 販売表.会員 ID = 会員表.会員 ID

AND 販売表.販売 ID IN

(SELECT 販売 ID FROM 決済表 WHERE 決済額 >= 0)

ウ FROM 会員表, 販売表, 決済表

WHERE DATEDIFF(販売表.決済期限日, NOW()) = 3

AND 販売表.会員 ID = 会員表.会員 ID

AND 販売表.販売 ID = 決済表.販売 ID

エ FROM 会員表, 販売表, 決済表

WHERE DATEDIFF(販売表.決済期限日, NOW()) = 3

AND 販売表.会員 ID = 会員表.会員 ID

AND 販売表.販売 ID = 決済表.販売 ID

AND 決済表.決済額 ◇ -1

設問4 会員への優待サービスのために、ポイント制度を導入する。そのために修正した会員表、決済表及び販売表の表構成を図2に示す。ポイント制度を導入するときに追加した列は0で初期化する。

会員表

会員 ID	氏名	電子メールアドレス	ポイント残高
-------	----	-----------	--------

決済表

販売 ID	決済日	決済額	付与ポイント
-------	-----	-----	--------

販売表

販売 ID	会員 ID	コンサート ID	席種	席数	販売日	販売額	決済期限日	使用ポイント
-------	-------	----------	----	----	-----	-----	-------	--------

図2 修正した会員表、決済表及び販売表の表構成

会員は購入申込み時に、1ポイント1円としてポイント残高の範囲で、販売額に充当するポイント数を指定する。販売サブシステムは、指定したポイント

数を使用ポイントに格納し、ポイント残高から減じる。会員は、販売額から使用ポイントを差し引いた金額を決済額として支払う。販売額の全額にポイントを充当した場合は、販売サブシステムは購入申込み時に支払手続が行われたものとし、決済処理として、決済表にレコードを追加する。

ポイント制度の導入時に追加したバッチ処理によって、前日に決済処理された販売 ID ごとに、その決済額が 20,000 円以上、10,000 円以上 20,000 円未満、10,000 円未満の場合に、それぞれ 3%, 2%, 1% のポイントを付与する。付与したポイント数は、付与ポイントに格納し、ポイント残高に加える。

決済表の付与ポイントを更新する正しい SQL 文を、解答群の中から選べ。

NOW, DATEDIFF は設問 3 で使用したユーザ定義関数と同じであり、FLOOR は引数の値以下で最大の整数値を返す関数である。

解答群

ア INSERT INTO 決済表(付与ポイント)
SELECT IF 決済額 >= 20000 THEN FLOOR(決済額 * 0.03)
ELSEIF 決済額 >= 10000 THEN FLOOR(決済額 * 0.02)
ELSE FLOOR(決済額 * 0.01) END
WHERE DATEDIFF(NOW(), 決済日) = 1

イ UPDATE 決済表 SET 付与ポイント = (
CASE 決済額 >= 20000 THEN FLOOR(決済額 * 0.03)
WHEN 決済額 >= 10000 THEN FLOOR(決済額 * 0.02)
ELSE FLOOR(決済額 * 0.01) END)
WHERE DATEDIFF(NOW(), 決済日) = 1

ウ UPDATE 決済表 SET 付与ポイント = (
CASE WHEN 決済額 >= 20000 THEN FLOOR(決済額 * 0.03)
WHEN 決済額 >= 10000 THEN FLOOR(決済額 * 0.02)
ELSE FLOOR(決済額 * 0.01) END)
WHERE DATEDIFF(NOW(), 決済日) = 1

エ UPDATE 決済表 SET 付与ポイント = (
IF 決済額 >= 20000 THEN FLOOR(決済額 * 0.03)
ELSEIF 決済額 >= 10000 THEN FLOOR(決済額 * 0.02)
ELSE FLOOR(決済額 * 0.01) END)
WHERE DATEDIFF(NOW(), 決済日) = 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4 ネットワークの障害分析と対策に関する次の記述を読んで、設間に答えよ。

G社は、ソフトウェア開発会社である。G社のネットワーク構成を図1に示す。

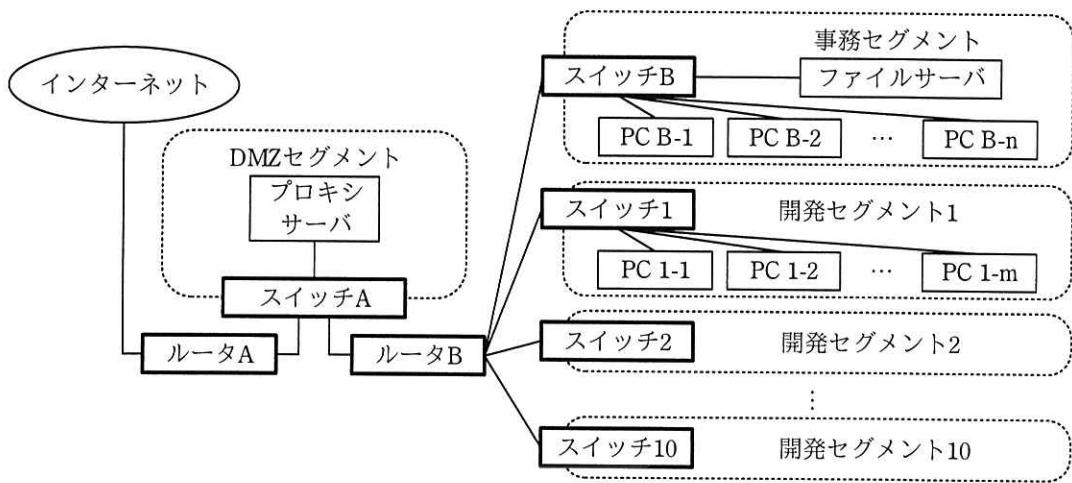


図1 G社のネットワーク構成

- (1) 図1中の各セグメントには、図に記された機器を含む複数の機器が配置されており、各機器はセグメントごとに用意されたスイッチにLANケーブルで接続されている。
- (2) 社員は、事務作業を事務セグメント内のPCを使用して行い、ソフトウェア開発作業を開発プロジェクトごとに設けられた開発セグメント内のPCを使用して行う。現在、開発セグメントは、開発セグメント1から開発セグメント10まである。
- (3) 事務セグメント内のPCは、リモートデスクトップ（手元のPCをクライアントとして、他のPCをGUIで遠隔操作する技術であり、操作される側がサーバになる）のクライアント機能（以下、リモートデスクトップ機能という）又はSSHを用いて、開発セグメント内のPCを遠隔操作できる。
- (4) 開発セグメント内のPCは、事務セグメント内のファイルサーバにアクセスできる。

- (5) 事務セグメント内の PC からインターネット上に公開された Web サイトを閲覧する際には、DMZ セグメント内のプロキシサーバを経由する。
- (6) ルータ B は、内蔵のパケットフィルタ型のファイアウォール機能によってセグメント間の通信の可否を制御しており、上記(3)～(5)に必要な通信だけを許可している。ルータ B のファイアウォール機能は、送信元ネットワークから、宛先ネットワークの指定したポート番号への通信の可否を制御するものであって、許可した通信の応答パケットの通過も許可する。

[障害の発生]

ある日、“事務セグメント内の PC B-1 から、リモートデスクトップ機能を用いて、開発セグメント 1 内の PC 1-1 を遠隔操作しようとしたが、接続できなかつた”と報告があつたので、障害箇所を特定するために原因の切分けを行つた。

初めに、事務セグメント内の PC B-2 から、PC 1-1 にリモートデスクトップ機能を用いて接続を試みたが、失敗した。

次に、PC B-1 から SSH を用いてログインした開発セグメント 1 内の PC 1-2 で ping コマンドを実行し、PC 1-1 から応答が返ってくることを確認した。

これらのことから、障害の原因として [a] や [b] の不具合が考えられたので、これらに不具合があるかどうかを調査して、原因を特定した。

[セグメントの追加]

新しい開発プロジェクトの立上げに伴い、開発セグメント 11 をネットワークに追加した。開発セグメント 11 は、開発セグメント 1～10 と同様にスイッチ 11 でルータ B と接続する。ルータ B のファイアウォールに追加した設定を表 1 に示す。ところが、表 1 の設定には誤りがあり、開発セグメント 11 に接続された PC に関して、[c] ことが分かつた。

事務セグメントと開発セグメント 11 のネットワークアドレスは、表 2 のとおりである。

表1 ルータBのファイアウォールに追加した設定

送信元ネットワーク	宛先ネットワーク	ポート番号（サービス）	可否
10.0.0.0/16	10.1.11.0/24	445（ファイル）	可
10.0.0.0/16	10.1.11.0/24	22（SSH）	可
10.0.0.0/16	10.1.11.0/24	3389（リモートデスクトップ）	可

表2 事務セグメントと開発セグメント11のネットワークアドレス

セグメント	ネットワークアドレス
事務セグメント	10.0.0.0/16
開発セグメント11	10.1.11.0/24

[障害発生の予防]

現在のネットワーク構成では、社内のセグメント間の全ての通信がルータBを経由するので、ルータBの過負荷によって社内ネットワークに障害が発生することが懸念される。そこで、ルータCを増設することによって負荷を分散させることとし、増設後の構成として構成案1と構成案2の二つを検討した。

構成案1及び構成案2における、ルータとスイッチの接続形態を図2に示す。

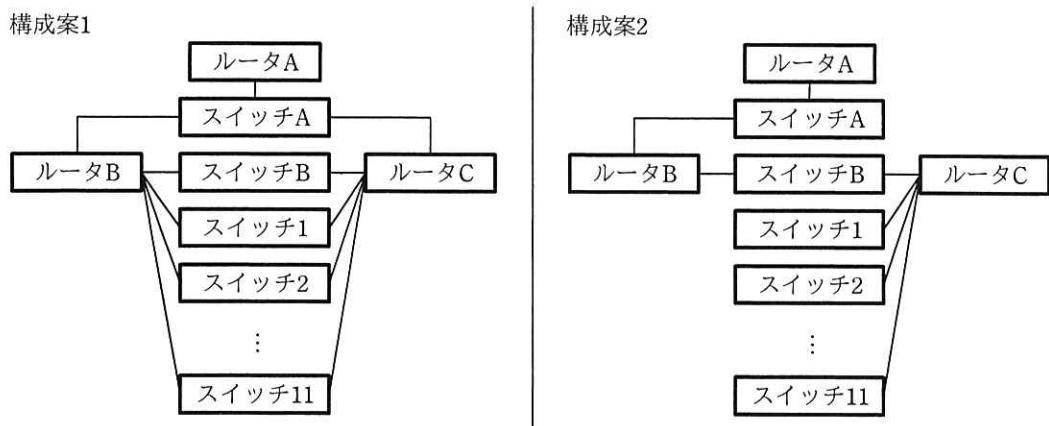


図2 ルータとスイッチの接続形態

構成案1では、セグメントごとにルータBかルータCのどちらかのルータを使用するように設定することによって、負荷を分散させる。このとき、ルータの冗長化技

術を用いて、一方のルータに障害が発生したときは、もう一方のルータが使用されるように構成する。

構成案 2 では、ルータ B とルータ C の役割を分けることによって、負荷を分散させる。ルータ C に、事務セグメントと開発セグメントの間の通信を中継する役割をもたせる。

G 社では ことや ことを重視して、構成案 1 を採用することにした。

設問 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 PC 1-1 の LAN ポート | イ PC 1-1 のソフトウェア |
| ウ スイッチ 1 | エ スイッチ B |
| オ 設定を含むルータ B のソフトウェア | カ ルータ B の LAN ポート |

c に関する解答群

- | |
|--|
| ア 事務セグメント内の PC から SSH を用いて当該 PC を遠隔操作できない |
| イ 事務セグメント内の PC からリモートデスクトップ機能を用いて当該 PC を遠隔操作できない |
| ウ 当該 PC から事務セグメント内のファイルサーバにアクセスできない |

d, e に関する解答群

- | |
|--|
| ア 可用性を高められる |
| イ 機密性を高められる |
| ウ 障害発生時に原因を特定しやすい |
| エ セグメント間で通信する際に経由する機器が少なくなる |
| オ ルータ B, C とスイッチ間をつなぐ LAN ケーブルの本数が少なくて済む |
| カ ルータ B とルータ C の負荷に大きな差が生じないように調整できる |

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 購買管理システムで行う処理に関する次の記述を読んで、設問1～3に答えよ。

自動車用アクセサリ類を製造しているK社では、購買部門が部品を発注する際に利用する購買管理システムを構築中である。システム部のC君は、購買ファイル更新可否チェック処理の開発を担当することになった。

購買ファイル更新可否チェック処理においては、部品の購入依頼情報を格納したファイル（以下、依頼ファイルという）中の各レコードについて、購買ファイルを更新できるかどうかをチェックする。更新することができないレコードは、更新対象外依頼ファイルに出力する。更新することができるレコードは、更新用依頼ファイルに出力し、一連の処理として実行する購買ファイル更新処理に引き渡す。ここで、依頼ファイルは、1日に1回、バッチ処理時間帯に製造部門から受け取る。依頼ファイルには、過去に受け取った購入依頼情報のレコードは含まれない。購買ファイル更新可否チェック処理の位置付けを、図1に示す。

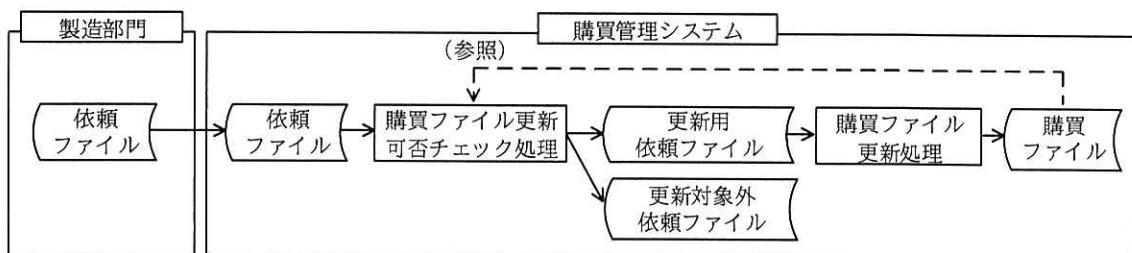


図1 購買ファイル更新可否チェック処理の位置付け

依頼ファイル及び購買ファイルのレコードの項目を表1に、その項目の説明を表2に示す。更新用依頼ファイル及び更新対象外依頼ファイルのレコードの項目は、依頼ファイルのレコードの項目と同じである。

表1 依頼ファイル及び購買ファイルのレコードの項目

ファイル名	項目
依頼ファイル	<u>依頼ID</u> , 依頼日時, 品番, 個数, 納期, 依頼者, 依頼種別
購買ファイル	<u>依頼ID</u> , 依頼日時, 品番, 個数, 納期, 依頼者, 購買ステータス

注記 下線付きの項目は主キーを表す。

表2 依頼ファイル及び購買ファイルのレコードの項目の説明

項目名	説明
依頼 ID	購入依頼情報を識別するために、購入依頼情報を新規に作成するときに、製造部門で採番して設定する一意な番号。購入依頼情報の変更又は削除のときは、新規に作成したときの依頼 ID を設定する。
依頼日時	依頼ファイルにレコードを格納した日時
品番	発注する部品の品番
個数	発注する部品の個数
納期	納品希望日
依頼者	製造部門の担当者名
依頼種別	“登録”，“変更”，“削除”がある。
購買ステータス	“購買受付”，“見積り中”，“発注済”，“納品済”がある。

- (1) 依頼種別には、依頼者が購入依頼情報を新規に作成するときは“登録”が、変更するときは“変更”が、削除するときは“削除”が設定される。
- (2) 購買ステータスには、購買部門が発注先に見積りを依頼する前は“購買受付”が、見積りを依頼して発注するまでの間は“見積り中”が、発注して納品されるまでの間は“発注済”が、納品後は“納品済”が設定される。

〔購買ファイル更新可否チェック処理の概要〕

- (1) 依頼ファイルのレコードの、依頼 ID と依頼種別を除く項目の内容は、正しいものとする。
- (2) 依頼 ID の昇順に整列された依頼ファイルのレコードを先頭から順に読み込んで、全てのレコードについて次の処理を行う。
 - ① 購買ファイルに同じ依頼 ID をもつレコードがない場合
依頼ファイルから読み込んだレコードの依頼種別が“登録”であれば、そのレコードを対象レコード出力処理を使って更新用依頼ファイルに出力する。
 - ② 購買ファイルに同じ依頼 ID をもつレコードがある場合
購買ステータスが“購買受付”又は“見積り中”で、依頼ファイルから読み込んだレコードの依頼種別が“変更”又は“削除”であれば、そのレコードを対象レコード出力処理を使って更新用依頼ファイルに出力する。
 - ③ ①と②の処理で更新用依頼ファイルに出力しなかったレコードを、対象外レコード出力処理を使って更新対象外依頼ファイルに出力する。

購買ファイル更新可否チェック処理の流れ図を、図2に示す。

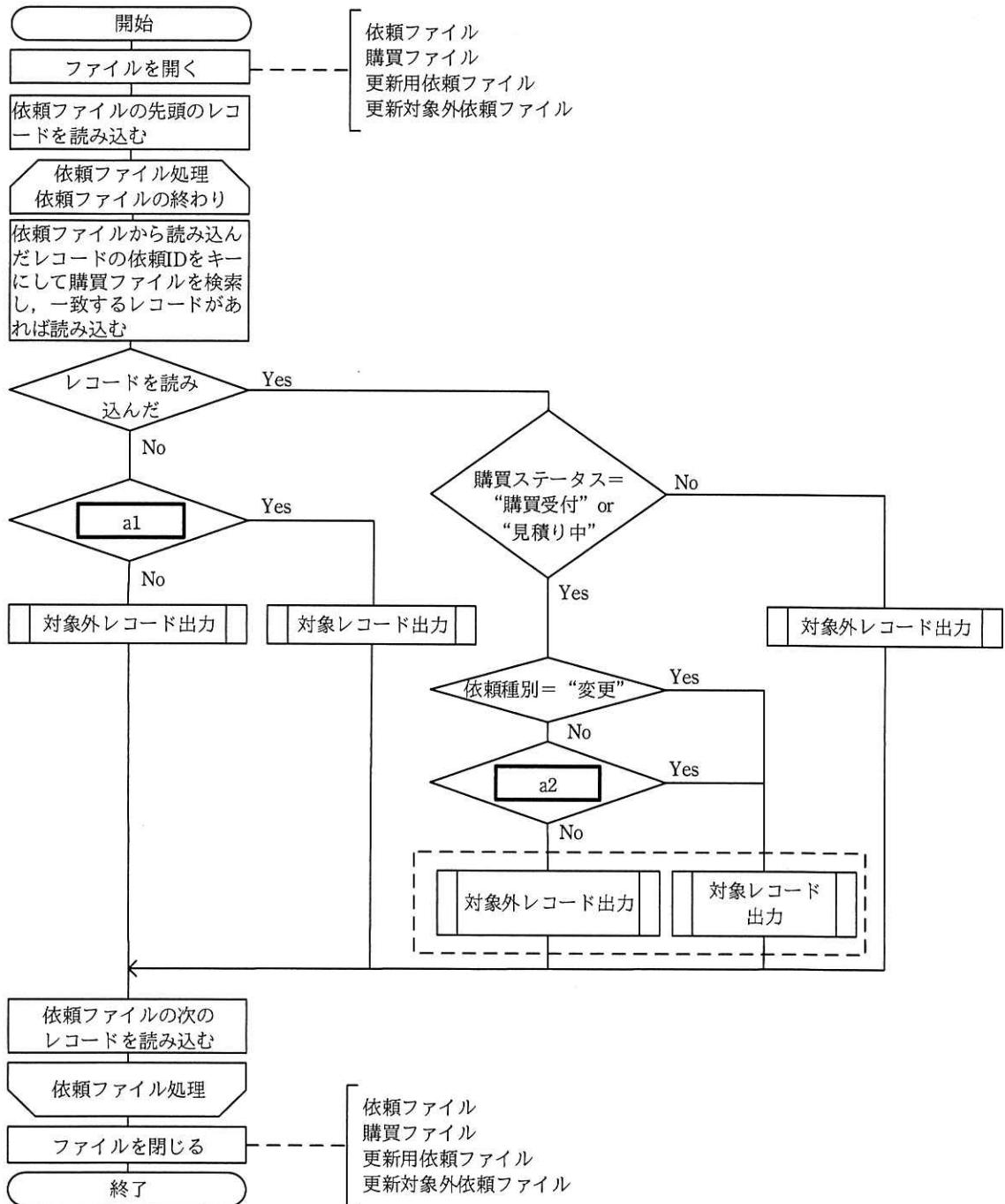


図2 購買ファイル更新可否チェック処理の流れ図

設問1 図2中の [] に入る適切な答えを、解答群の中から選べ。ここで、
a1とa2に入る答えは、aに関する解答群の中から組合せとして適切なもの
を選ぶものとする。

aに関する解答群

	a1	a2
ア	依頼種別 = “登録”	依頼種別 = “削除”
イ	依頼種別 = “登録”	依頼種別 ≠ “削除”
ウ	依頼種別 ≠ “登録”	依頼種別 = “削除”
エ	依頼種別 ≠ “登録”	依頼種別 ≠ “削除”
オ	依頼種別 = “変更”	依頼種別 = “削除”
カ	依頼種別 = “変更”	依頼種別 ≠ “削除”

設問2 購買ファイル更新可否チェック処理のテストケースを設計する。テストケースに漏れがないように購買ファイル更新可否チェック処理で出力するファイルに着目して、決定表を作成した。出力するファイルの決定表を表3に示す。表3中の [] に入る適切な答えを、解答群の中から選べ。

表3 出力するファイルの決定表

条件 ¹⁾		Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N
購買ステータス	購買受付	Y	Y	Y	—	—	—	—	—	—	—	—	—	—	—	—
	見積り中	—	—	—	Y	Y	Y	—	—	—	—	—	—	—	—	—
	発注済	—	—	—	—	—	—	Y	Y	Y	—	—	—	—	—	—
	納品済	—	—	—	—	—	—	—	—	—	Y	Y	Y	—	—	—
依頼種別	登録	Y	—	—	Y	b	—	Y	—	—	Y	—	c	Y	—	—
	変更	—	Y	—	—		—	—	Y	—	—	Y		—	Y	—
	削除	—	—	Y	—		Y	—	—	Y	—	—		—	—	Y
更新用依頼ファイルに出力		—	X	X	—		X	—	—	—	—	—		X	—	—
動作 ²⁾		X	—	—	X		—	X	X	X	X	X		—	X	X

注¹⁾ 依頼ファイルから読み込んだレコードと依頼IDが同じ購買ファイルのレコードがある。

²⁾ 更新対象外依頼ファイルに出力

b, c に関する解答群

ア	一	イ	一	ウ	一	エ	一	オ	一	カ	一
	Y		Y		Y		-		-		-
	-		-		-		-		Y		Y
	X		-		-		X		-		-
	-		X		-		-		X		-

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。

購買ファイル更新可否チェック処理のテストのためにテスト用レコードを作成した。購買ファイルには表 4 に示すテスト用レコードを、依頼ファイルには表 5 に示すテスト用レコードを、事前に格納しておいた。

表 4 購買ファイルのテスト用レコード

依頼 ID	依頼日時	品番	個数	納期	依頼者	購買ステータス
10000003	2018-10-19 10:00	A	100	2018-12-10	山田	購買受付
10000004	2018-10-19 11:00	B	300	2018-12-10	鈴木	見積り中
10000005	2018-10-19 11:00	D	100	2018-12-10	田中	見積り中
10000006	2018-10-19 11:00	C	200	2018-12-11	佐藤	発注済

表 5 依頼ファイルのテスト用レコード

依頼 ID	依頼日時	品番	個数	納期	依頼者	依頼種別
10000003	2018-10-20 10:00	A	100	2018-12-10	山田	削除
10000004	2018-10-20 11:00	B	400	2018-12-10	鈴木	登録
10000005	2018-10-20 11:05	D	200	2018-12-10	田中	変更
10000006	2018-10-20 11:10	C	200	2018-12-11	佐藤	削除
10000007	2018-10-20 11:15	E	300	2018-12-08	佐藤	登録
10000008	2018-10-20 12:00	F	300	2018-12-08	斎藤	変更

これらのテスト用レコードを用いて、購買ファイル更新可否チェック処理を実行した場合、図 2 の破線で囲んだ処理のうち、対象外レコード出力処理では依頼 ID が **d** のレコードが処理され、対象レコード出力処理では依頼 ID が **e** のレコードが処理される。

d に関する解答群

- | | |
|------------|------------|
| ア 10000003 | イ 10000004 |
| ウ 10000005 | エ 10000006 |
| オ 10000007 | |

e に関する解答群

- | | |
|-----------------------|-----------------------|
| ア 10000003 と 10000005 | イ 10000004 と 10000005 |
| ウ 10000004 と 10000006 | エ 10000005 と 10000008 |

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 プロジェクトのスケジュール作成に関する次の記述を読んで、設問1, 2に答えよ。

W社は、業務効率化を目的として、紙のりん議書を回付して行っている申請承認に関わる業務を電子化するプロジェクト（以下、りん議書電子化プロジェクトという）を立ち上げた。

[りん議書電子化プロジェクトの概要]

りん議書電子化プロジェクトでは、紙のりん議書を電子帳票化するシステム（以下、電子帳票システムという）を、W社の情報システム部が新規に開発する。電子帳票化されたりん議書の回付（以下、りん議書回付という）には、クラウドサービスプロバイダが提供している回付業務のクラウドサービス（以下、回付サービスという）を、比較検討の上、導入して利用する。

設問1 次の記述中の [] に入る正しい答えを、解答群の中から選べ。

プロジェクトマネージャのQ君は、りん議書電子化プロジェクトのスケジュールを、アローダイアグラムを用いて作成することにした。

アローダイアグラムの例を、図1に示す。

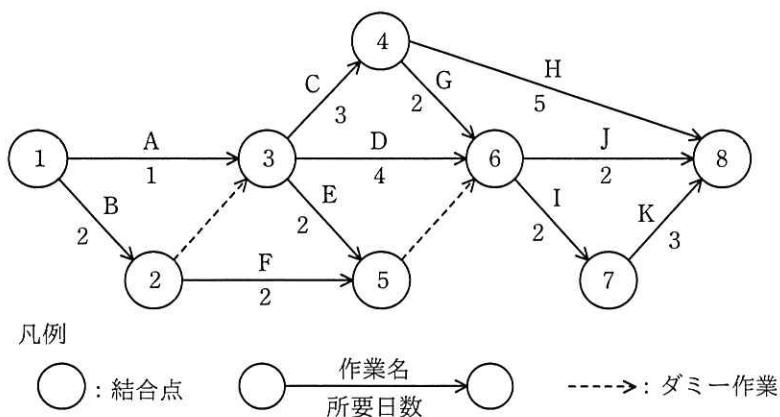


図1 アローダイアグラムの例

図 1 のクリティカルパスは [a] であり、全ての作業を完了するため必要な所要日数（以下、総所要日数という）は [b] 日である。また、作業 J について、最早開始日と最遅開始日は [c] であり、余裕日数は [d] 日である。ここで、結合点 1 から始まる作業の開始日は 0 日とする。また、余裕日数とは、当該作業に先行する作業が遅れなしに完了したとき、総所要日数を増加させることなく、当該作業が遅れてもよい日数である。

a に関する解答群

作業名	
ア	A, C, G, I, K
イ	A, D, I, K
ウ	B, C, G, I, K
エ	B, C, H
オ	B, D, I, K
カ	B, F, I, K

b に関する解答群

ア 9

イ 10

ウ 11

エ 12

オ 13

c に関する解答群

ア 4日と7日

イ 4日と8日

ウ 5日と8日

エ 5日と9日

オ 7日と10日

カ 7日と11日

d に関する解答群

ア 0

イ 1

ウ 2

エ 3

オ 4

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。

Q 君は、りん議書電子化プロジェクトのスケジュール作成に当たり、プロジェクトで実施する作業を、表1に示す作業一覧表にまとめ、作業の流れを整理中である。作成途中のアローダイアグラムを、図2に示す。

表1 作業一覧表

作業名	作業項目	所要日数	作業内容
A	プロジェクト計画作成	20	スケジュールや体制、要員配置など、プロジェクトの計画を作成する。
B	業務要件定義	30	紙のりん議書の電子帳票化及びりん議書回付に関する業務要件を定義する。
C	システム要件定義	20	電子帳票システムに関しては、紙のりん議書を電子帳票化する業務要件を満たすソフトウェア構成及びハードウェアとミドルウェアの構成を定義する。回付サービスに関しては、電子帳票システムとの連携要件 ¹⁾ を定義する。
D	ハードウェア、ミドルウェア調達	30	電子帳票システムのシステム要件に適合したハードウェアとミドルウェアを調達する。
E	ハードウェア設置、ミドルウェア設定	30	調達したハードウェアを設置し、外部設計で定義したミドルウェアのパラメタ ²⁾ 設定を行う。
F	回付サービスの候補選定	10	りん議書回付の業務要件に適合する回付サービスの候補を選定する。できるだけ多くの回付サービスを候補とするために、りん議書回付に関する業務要件だけを前提として選定を行い、電子帳票システムとの連携要件は前提としない。
G	回付サービスの適合性検証、決定	40	候補として選定した回付サービスについて、それぞれ業務要件との適合性検証、システム要件定義で定義した電子帳票システムとの連携要件の適合性検証を実施し、導入する回付サービスを決定する。
H	回付サービスの利用準備	40	回付サービスを利用するための各種パラメタ設定などを行う。
I	外部設計	20	システム要件を基に電子帳票システムのソフトウェアの外部設計、ミドルウェアのパラメタ定義を行う。
J	内部設計	20	外部設計を基に電子帳票システムの内部設計を行う。
K	プログラム開発、単体テスト	30	内部設計を基に電子帳票システムのプログラム開発と単体テストを行う。
L	結合テスト	20	電子帳票システムと回付サービスを連携させた結合テストを行う。
M	総合テスト	20	実運用を想定した総合的なテストを行う。
N	本番移行	10	各種のシステム資源を稼働環境に移行する。

注¹⁾ 連携要件とは、電子帳票システムとの連携に必要となる機能、データ項目及びインターフェースである。

²⁾ ミドルウェアの外部仕様は、公開されている。

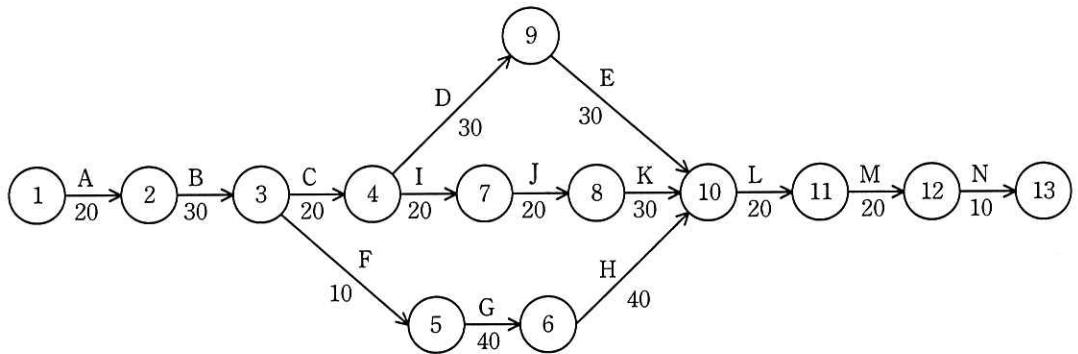


図2 作成途中のアローダイアグラム

作成途中のアローダイアグラムには、二つのダミー作業が欠けている。りん
議書電子化プロジェクトのスケジュールを正しく表現するためには、

e と f にダミー作業を追加する必要がある。

ダミー作業追加後のアローダイアグラムにおいて、りん議書電子化プロジェ
クトの総所要日数は、 g 日である。

e, fに関する解答群

- | | |
|------------------|-----------------|
| ア 結合点 2 から結合点 9 | イ 結合点 3 から結合点 9 |
| ウ 結合点 4 から結合点 5 | エ 結合点 4 から結合点 6 |
| オ 結合点 6 から結合点 7 | カ 結合点 6 から結合点 8 |
| キ 結合点 6 から結合点 12 | ク 結合点 7 から結合点 9 |

gに関する解答群

- | | | | | | |
|-------|-------|-------|-------|-------|-------|
| ア 160 | イ 170 | ウ 180 | エ 190 | オ 200 | カ 210 |
|-------|-------|-------|-------|-------|-------|

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 広告制作業務の現状把握と改善に関する次の記述を読んで、設問 1～3 に答えよ。

X 社は、新聞、雑誌、カタログ誌などの紙媒体に掲載する広告（以下、紙広告という）及びインターネット上の Web サイトに掲載する広告（以下、Web 広告という）を取り扱う広告制作会社であり、複数の拠点で業務を行っている。

各拠点では、担当するエリア内で営業活動を行い、エリア内の顧客から受け取った広告原稿を基に、広告を制作する。各拠点の体制を、図 1 に示す。

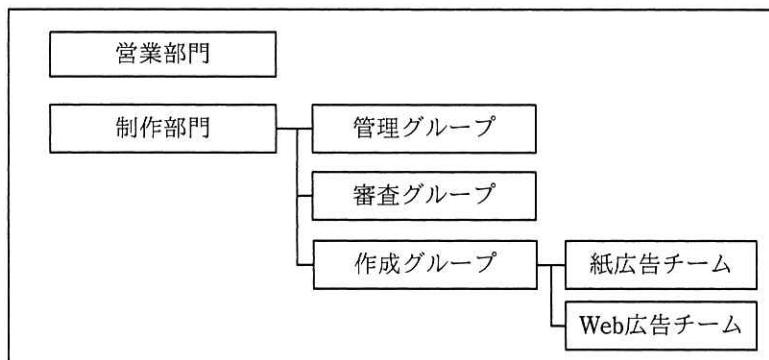


図 1 各拠点の体制

X 社では、広告原稿の受渡しを手渡しで行っている。制作部門では、営業部門から広告原稿をいつ受け取り、営業部門にいつ返却のために渡したかを、授受管理簿に記録する。授受管理簿には、広告原稿を用いて作業を行う担当者が誰であるかも、併せて記入している。授受管理簿の様式を、図 2 に示す。

通番	受付日	営業担当者	顧客名	紙広告 原稿 枚数	Web 広告 原稿 枚数	返却日	返却事由	作業担当者		
								審査 担当者	紙広告 作成 担当者	Web 広告 作成 担当者

図 2 授受管理簿の様式

[制作部門での作業の流れ]

(1) 管理グループの作業 1

- ① 営業部門から受け取った広告原稿を確認し、授受管理簿の通番、受付日、営業担当者、顧客名、紙広告原稿枚数、Web 広告原稿枚数の欄に記入する。ここで、広告原稿が紙広告又は Web 広告の一方だけの場合は、広告原稿のない方の原稿枚数は“一”とする。
- ② 広告原稿の形式的な点検（枚数の過不足、折れや汚れの有無など）を行う。
- ③ 点検の結果、不備がある場合は、返却のために広告原稿を営業部門に渡す。授受管理簿の返却日、返却事由の欄に記入する。返却事由は“原稿不備”とする。
営業部門に渡すときに、待ちは発生しない。
- ④ 点検の結果、不備がない場合は、作業担当者を決定し、審査担当者、紙広告作成担当者、Web 広告作成担当者の欄に記入する。ここで、広告原稿が紙広告又は Web 広告の一方だけの場合は、広告原稿のない方の作成担当者は“一”とする。記入した後、広告原稿を審査担当者に渡す。審査担当者に渡すときに、待ちは発生しない。

(2) 審査グループの作業

- ① 広告原稿の内容が X 社の広告制作基準に違反していないかどうかを審査し、審査結果を“適”又は“不適”として、審査票を起票する。
- ② 広告原稿と審査票を管理グループに渡す。管理グループに渡すときに、待ちは発生しない。

(3) 管理グループの作業 2

- ① 審査結果が“不適”であれば、審査票の写しをとり、審査票の原本は保管する。その写しを添えて、返却のために広告原稿を営業部門に渡す。授受管理簿の返却日、返却事由の欄に記入する。返却事由は“基準違反”とする。営業部門に渡すときに、待ちは発生しない。
- ② 審査結果が“適”であれば、審査票の原本を保管し、広告原稿を紙広告と Web 広告とに分けて、それぞれを決定している作成担当者に渡す。
 - ・他の作業を行っている場合は、作業の終了を待って渡す。
 - ・他の作業を行っていない場合は、すぐに渡す。

(4) 作成グループの作業

- ① 受け取った広告原稿を基に、広告を作成する。
- ② 作成完了後、作成担当者が、作成した広告（以下、作成済広告という）と広告

原稿を管理グループに渡す。管理グループに渡すときに、待ちは発生しない。

(5) 管理グループの作業 3

- ① 紙広告と Web 広告について、受け取った広告原稿と作成済広告がそろっているかどうかを授受管理簿と照合して確認する。
- ② 確認の結果、紙広告と Web 広告について、どちらかに不足がある場合は、必要なものが全てそろうまで待つ。
- ③ 確認の結果、紙広告と Web 広告について、必要なものが全てそろっている場合は、営業部門に広告原稿と作成済広告を渡す。授受管理簿の返却日、返却事由の欄に記入する。返却事由は“作成完了”とする。営業部門に広告原稿と作成済広告を渡すときに、待ちは発生しない。

設問 1 10月11日における授受管理簿を、図3に示す。通番6の行の広告原稿については、【制作部門での作業の流れ】(1)④の作業を実施し、完了したところである。図3中の [] に入る適切な答えを、解答群の中から選べ。

通番	受付日	営業担当者	顧客名	紙広告 原稿枚数	Web広告 原稿枚数	返却日	返却事由	作業担当者		
								審査担当者	紙広告作成担当者	Web広告作成担当者
1	10月4日	佐々木		2枚	2枚	10月10日	作成完了	佐藤	高橋	伊藤
2	10月5日	山口		3枚	—	10月5日	原稿不備			
3	10月5日	松本		—	3枚	10月6日	基準違反	佐藤	—	渡辺
4	10月6日	松本		5枚	—	10月10日	作成完了	鈴木	田中	—
5	10月7日	山口		—	2枚	10月10日	作成完了	佐藤	—	山本
6	10月11日	佐々木		—	5枚					

注記 網掛けの部分は表示していない。空白の部分は未記入であることを示す。

図3 10月11日における授受管理簿

解答群

ア		佐藤	—	渡辺
イ		佐藤	—	—
ウ	10月11日	原稿不備	佐藤	—
エ	10月11日	原稿不備		

設問2 作成グループにおいて、広告原稿の“紛失事故”騒動が発生した。“紛失事故”的原因は、Web広告作成担当者が、広告原稿を保管した場所を勘違いしていたことであった。広告原稿は翌日に発見されて事なきを得たが、X社の作業方法では、広告原稿をグループ間で受け渡すとき及び営業部門に渡すときに待ちが発生する場合がある。この場合に、一時的な保管が必要となって、“紛失事故”が誘発されるおそれがある。

[制作部門での作業の流れ] のうち、作成グループの作業以外の作業で、広告原稿の一時的な保管に伴って“紛失事故”が誘発されるおそれがある作業として適切な答えを二つ、解答群の中から選べ。

解答群

- | | |
|---------------|---------------|
| ア 管理グループの作業 1 | イ 管理グループの作業 2 |
| ウ 管理グループの作業 3 | エ 審査グループの作業 |

設問3 X社では、“紛失事故”が誘発されるおそれを低減させ、管理グループの作業を軽減させるために、広告原稿については、電子化を行って、サーバ上で共有することで、これまでの手渡しによる受渡しの作業をなくすことを構想している。広告原稿を電子化する時点の候補として、営業部門が制作部門に渡す直前（以下、営業出口という）と、制作部門が営業部門から受け取った直後（以下、制作入口という）が挙がっている。ここで、作成済広告の受渡しは、手渡しのままとする。

次の記述中の [] に入る適切な答えを、解答群の中から選べ。

図3の通番4に対応する制作部門での一連の作業において管理グループが行った作業を、表1に整理した。項目1の受渡し作業に関して、広告原稿、審査票、作成済広告のそれぞれについて、管理グループが受けの作業を行った件数と、管理グループが渡しの作業を行った件数の合計は9件であった。広告原稿の電子化をどこで行うかによって、項目1の受渡し作業のうち、手渡しによる作業の合計の件数は異なる。営業出口で行った場合は [a] 件になり、制作入口で行った場合は [b] 件になる。ここで、電子化を制作入口で行った場合には、営業部門への渡しは手渡しで行うものとする。

表1 管理グループが行った作業（図3の通番4）

項目番号	作業	件数
1 内 訳	受渡し	9
	広告原稿の受け（対営業部門）	
	広告原稿の渡し（対営業部門）	
	広告原稿の受け（制作部門内）	
	広告原稿の渡し（制作部門内）	
	作成済広告の受け（制作部門内）	
	作成済広告の渡し（対営業部門）	
	審査票の受け（制作部門内）	
2	広告原稿の形式的な点検	1
3	作業担当者の決定	1
4	審査票の原本を保管	1
5	授受管理簿の記入	

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 2

イ 3

ウ 4

エ 5

オ 6

カ 7

キ 8

ク 9

次の問8は必須問題です。必ず解答してください。

問8 次のプログラムの説明及びプログラムを読んで、設問1～3に答えよ。

整数式を受け取って、その値を返すプログラムである。例えば、例1に示す整数式を受け取ると、その値50を返す。

例1: $2 \times (34 - (5 + 67) \div 8)$

[プログラムの説明]

- (1) 整数式は、文字の列で与えられる。整数式は、次のもので構成される。
 - ・符号のない数字0～9の並び
 - ・演算子： +, -, ×, ÷
 - ・括弧： (,)
- (2) 引数Expression[]で整数式を、引数ExpLenで整数式の文字数を、それぞれ受け取る。
- (3) プログラム中の破線で囲んだ解析処理の部分では、受け取った整数式を解析し、計算に必要な情報を配列及び変数に設定する。
- (4) プログラム中の破線で囲んだ計算処理の部分では、(3)で設定した情報を用いて、整数式の値を計算する。整数式の値は、Value[0]に得られる。
- (5) 各配列の添字は、0から始まる。各配列の要素数は、十分に大きいものとする。
- (6) 受け取った整数式に誤りはないものとする。また、計算の過程で、あふれやゼロ除算は発生しないものとする。

[プログラム]

- 整数型関数: compute(文字型: Expression[], 整数型: ExpLen)
- 文字型: Operator[100]
- 整数型: OpCnt, Priority[100], Value[100]
- 文字型: chr
- 整数型: i, ip, nest

解析処理（詳細は「プログラム（解析処理の部分）」に示す）

計算処理（詳細は「プログラム（計算処理の部分）」に示す）

- return Value[0]

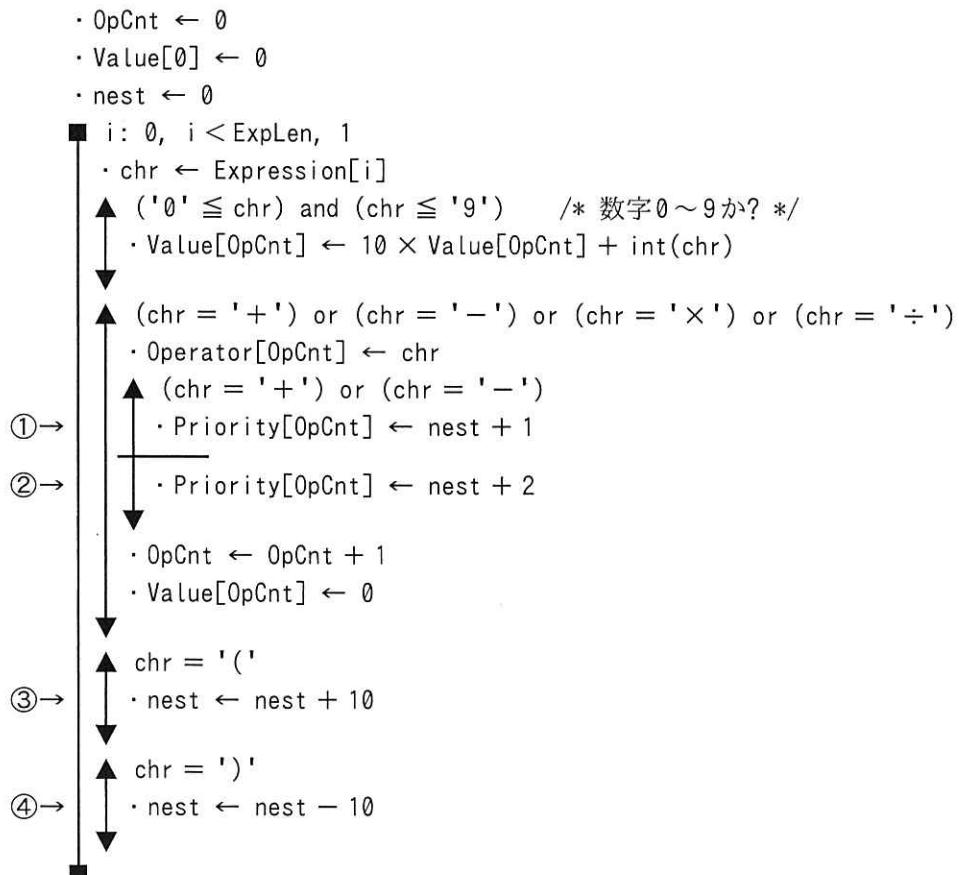
[プログラム（解析処理の部分）の説明]

- (1) Expression[] で渡された整数式を解析し、計算に必要な情報を配列 Operator[], Priority[], Value[] 及び変数 OpCnt に設定する。関数 int() は、引数の数字が表す値を整数型で返す。
- (2) 例 1 の整数式について、プログラム（解析処理の部分）を実行した直後の各配列及び変数の状態を、図 1 に示す。

Expression[]	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	...
	2	×	(3	4	-	(5	+	6	7)	÷	8)	...
Value[]	0	1	2	3	4											...
	2	34	5	67	8											...
Operator[]	×	-	+	÷												OpCnt
																4
Priority[]	2	11	21	12												...

図 1 プログラム（解析処理の部分）を実行した直後の状態

[プログラム（解析処理の部分）]



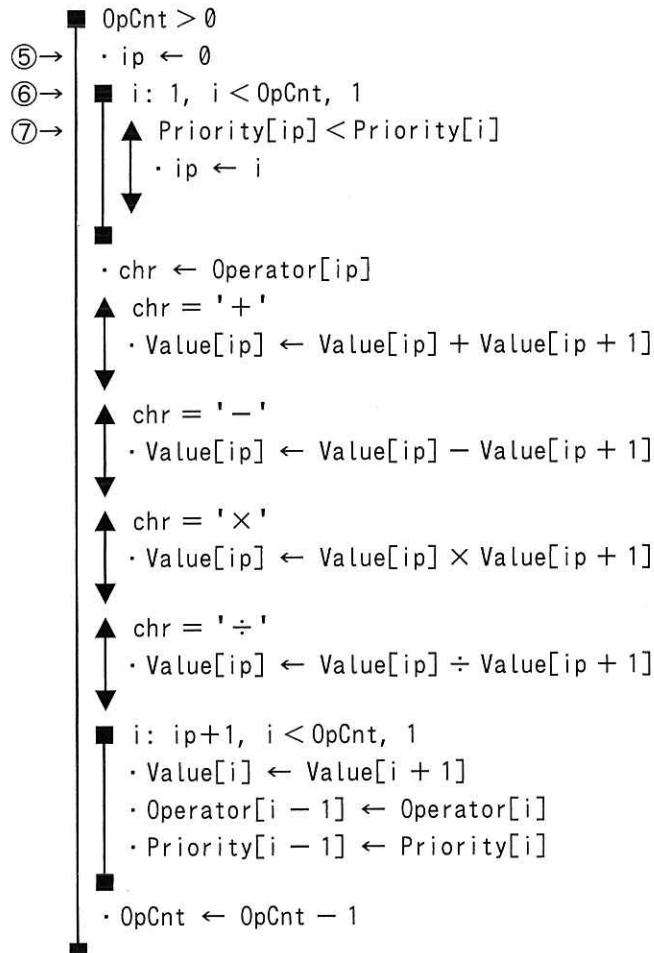
[プログラム（計算処理の部分）の説明]

- (1) 整式式の値を計算していく。図 1 に示す各配列及び変数の状態から、プログラム（計算処理の部分）の最外側の繰返しを 1 回実行した直後の各配列及び変数の状態を、図 2 に示す。

	0	1	2	3	---
Value[]	2	34	72	8	---
Operator[]	×	-	÷		OpCnt
Priority[]	2	11	12		

図 2 プログラム（計算処理の部分）の最外側の繰返しを 1 回実行した直後の状態

[プログラム（計算処理の部分）]



設問 1 プログラム（解析処理の部分）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。

プログラム（解析処理の部分）の行①～④で用いている定数について考察する。

まず、行③及び④の処理では、定数として 10 を用いているが、この定数は 10 である必要はない。このプログラムにおいては、定数が a であれば常に正しい演算順序が保証される。

また、行①及び②の処理では、定数として 1 及び 2 を用いているが、次に示すように書き換えることが可能である。ここで、`priLow` 及び `priHigh` は整数の定数を表し、その値は $priLow < priHigh$ とする。

①→ · `Priority[OpCnt] ← nest + priLow`

②→ · `Priority[OpCnt] ← nest + priHigh`

このように表現したとき、行③及び④の処理では、`nest` の値を増減する定数が b のときに限り正しい演算順序が保証されることになる。

a に関する解答群

ア 1 以上

イ 2 以上

ウ 11 以下

エ 12 以下

b に関する解答群

ア `priHigh` 以上

イ `priHigh + 1` 以上

ウ `priHigh - priLow` 以上

エ `priHigh - priLow + 1` 以上

設問2 優先順位の等しい演算子が複数個含まれている整式式の、演算の実行順序について考察する。プログラム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。ここで、c1～c3に入る答えは、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。

プログラム（計算処理の部分）では、優先順位の等しい演算子が複数個含まれている場合、演算を左から順に実行するようになっている。このプログラムでは、演算を左から順に実行するか右から順に実行するかは、行 [] c1 の内容が [] c2 か [] c3 かで決まる。

演算の実行順序によって、計算結果が異なることがある。例えば、次の四つの整式式のケースを考える。

$$\text{ケース 1: } (12 + 3 + 1) \times 4 \times 2$$

$$\text{ケース 2: } (12 + 3 + 1) \div 4 \div 2$$

$$\text{ケース 3: } (12 - 3 - 1) \times 4 \times 2$$

$$\text{ケース 4: } (12 - 3 - 1) \div 4 \div 2$$

これらのケースのうち、演算を左から実行しても右から実行しても、プログラムによる計算結果が等しくなるのは、ケース [] d である。

cに関する解答群

	c1	c2	c3
ア	⑤	$\cdot ip \leftarrow 0$	$\cdot ip \leftarrow OpCnt - 1$
イ	⑥	$i: 1, i < OpCnt, 1$	$i: OpCnt, i > 0, -1$
ウ	⑥	$i: 1, i < OpCnt, 1$	$i: OpCnt - 1, i > 0, -1$
エ	⑦	$Priority[ip] < Priority[i]$	$Priority[ip] \leq Priority[i]$

dに関する解答群

ア 1

イ 1及び2

ウ 1及び3

エ 1及び4

設問3 プログラムの動作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。

符号付き整定数（数字の並びの先頭に符号+又は-を付けた定数）を含む整数式を考える。符号付き整定数は、例2のように括弧で囲んで記述する。ただし、符号付き整定数の直前の文字が演算子でない場合は、例3のように括弧で囲まなくてもよい。

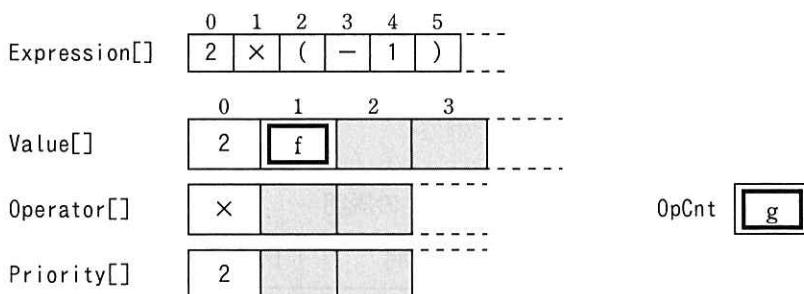
例2: $(+2) \times ((-3) + (-4))$

例3: $+2 \times (-3 + (-4))$

符号付き整定数を含む整数式 $2 \times (-1)$ についてプログラム（解析処理の部分）を実行した結果を、図3に示す。

このように、符号付き整定数を含む整数式を受け取ったとき、プログラムは

[e]。



注記 網掛け部分（値が格納されているとは限らない）は表示していない。

図3 整数式 $2 \times (-1)$ についてプログラム（解析処理の部分）を実行した結果

eに関する解答群

- ア 整数式が符号付き整定数で始まる場合に、正しい値を返さない
- イ 整数式中に符号-の付いた符号付き整定数がある場合に、正しい値を返さない
- ウ 整数式中に二つ以上の符号付き整定数が含まれる場合に、正しい値を返さない
- エ 正しい値を返す

f, gに関する解答群

- ア -1
- イ 0
- ウ 1
- エ 2

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。

本問は、図1に示す路線構成の鉄道模型における列車の運行をシミュレーションするプログラムである。表1は、図1の説明である。

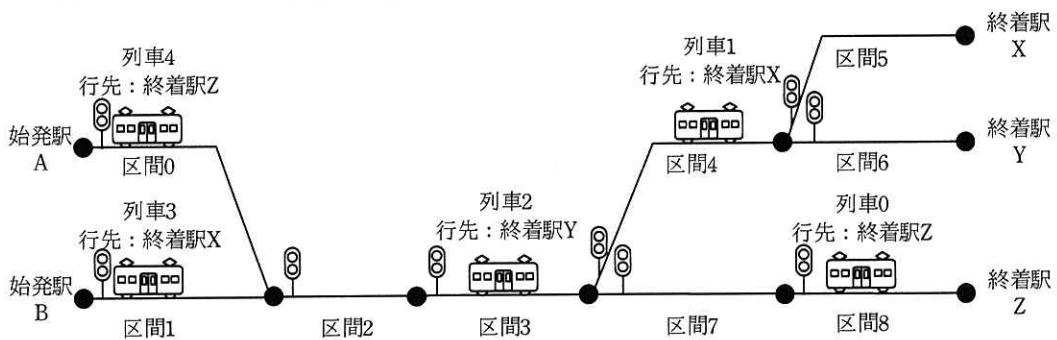


図1 鉄道模型の路線構成及び列車位置

表1 図1の説明

区間	区間の出口で接続する区間	区間内の列車（行先）
区間 0	区間 2	—
区間 1	区間 2	—
区間 2	区間 3	—
区間 3	区間 4	区間 7 列車 2 (終着駅 Y)
区間 4	区間 5	区間 6 列車 1 (終着駅 X)
区間 5	—	—
区間 6	—	—
区間 7	区間 8	—
区間 8	—	— 列車 0 (終着駅 Z)

[列車を進行させるルール]

- (1) 各列車は、あらかじめ決められた始発駅から、あらかじめ決められた終着駅を行先として、路線を後戻りすることなく進行する。
- (2) 一つの区間には、同時に一つの列車しかいることができない。
- (3) 各区間の入口には信号機があり、赤又は緑を表示する。
- (4) 各列車は、終着駅を出口とする区間にいるときは無条件に進行できる。終着駅を出口とする区間にいる列車が進行した場合、その列車は路線上から取り除かれる。
- (5) 各列車は、現在いる区間の出口で接続する区間（以下、次区間という）が次の条件をいずれも満たしたとき、次区間に進行できる。
 - ① 次区間の信号機の表示が緑である。
 - ② 次区間に進行すると、最終的には行先とする終着駅に到達できる。
- (6) 次の二つの処理を繰り返し実行して各列車を進行させる。
 - ① 路線を構成する全ての区間の信号機それぞれについて、区間内に列車がいるときは表示を赤に、列車がないときは表示を緑にする。
 - ② 路線を構成する全ての区間それぞれについて、次の処理を行う。
 - (ア) 終着駅を出口とする区間に列車がいる場合は、(4)に従って列車を進行させ、路線上から取り除く。
 - (イ) (ア)で述べた以外の区間に列車がいる場合は、(5)に従って進行できる列車を進行させ、進行した区間の信号機の表示を赤にする。

[プログラム 1 の説明]

- (1) 構造体 `block_info` は、鉄道模型の区間を表現する。

```
struct block_info {
    struct train_info* train;
    struct block_info* next[2];
    int signal;
};
```

`train`： 区間にいる列車を表現した、構造体 `train_info` へのポインタである。区間に列車がないとき、NULL を設定する。

next : 次区間を表現した、構造体 `block_info` へのポインタを格納する配列（配列の大きさは 2）である。終着駅を出口とする区間では、いずれの要素にも NULL を設定する。次区間が一つしかなければ、そのポインタは最初の要素に設定し、2番目の要素には NULL を設定する。

signal : 信号機に表示する色を示す。次のいずれかの定数値をとる。

RED	表示は赤
GREEN	表示は緑

(2) 構造体 `train_info` は、鉄道模型の列車を表現する。

```
struct train_info {  
    int number;  
    struct block_info* dest;  
};
```

number : 列車の番号である。

dest : 列車の終着駅を出口とする区間を表現した、構造体 `block_info` へのポインタである。

(3) 関数 `set_signals` の仕様は、次のとおりである。

機能 : 全ての区間の信号機それぞれについて、区間に列車がいるときは表示を赤に、列車がないときは表示を緑にする。

引数 : blocks 全ての区間を表現した、構造体 `block_info` の配列
nblocks blocks の要素数

[プログラム 1]

```
#include <stddef.h>  
  
#define RED (0)  
#define GREEN (1)  
  
struct train_info {  
    int number;  
    struct block_info* dest;  
};  
  
struct block_info {
```

```

    struct train_info* train;
    struct block_info* next[2];
    int signal;
};

void set_signals(struct block_info[], int);

void set_signals(struct block_info blocks[], int nblocks) {
    int i;
    struct block_info* block;
    for (i = 0; i < nblocks; i++) {
        block = &blocks[i];
        if ([ ] a [ ]) {
            block->signal = GREEN;
        }
        else {
            block->signal = RED;
        }
    }
}

```

C

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 block->next[0]->train == NULL
- イ block->next[1]->train == NULL
- ウ block->next[0]->train == NULL || block->next[1]->train == NULL
- エ block->next[0]->train == NULL && block->next[1]->train == NULL
- オ block->signal == RED
- カ block->train == NULL
- キ block->train != NULL

設問 2 関数 proceed は、進行できる全ての列車を進行させるプログラムである。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) 関数 proceed の仕様は、次のとおりである。

機能： 全ての区間それぞれについて、進行できる列車がいるかどうかを判定し、進行できると判定された列車を進行させる。終着駅を出口とする区間にいる列車を進行させた場合、その列車を路線上から取り除く。出口で他の区間と接続する区間にいる列車を進行させた場合、進行した区間の信号機の表示を赤にする。

引数： blocks 全ての区間を表現した、構造体 block_info の配列
nblocks blocks の要素数

- (2) 関数 proceed から呼び出される関数 find_block の仕様は、次のとおりである。

機能： 引数 block で示す区間が、引数 dest で示す区間に最終的に到達できる経路上にあるかどうかを判定する。

引数： block 判定対象の区間を表現した、構造体 block_info へのポインタ
dest 列車の終着駅を出口とする区間を表現した、構造体 block_info へのポインタ

返却値： 到達できるとき、1
到達できないとき、0

[プログラム 2]

```
#include <stddef.h>

void proceed(struct block_info[], int);
int find_block(struct block_info*, struct block_info*);

void proceed(struct block_info blocks[], int nblocks) {
    int i, j;
    struct block_info* block;
    for (i = nblocks - 1; i >= 0; i--) {
        block = &blocks[i];
        if (block->train == NULL) {
            continue;
        }
        if (b) {
            block->train = NULL;
            continue;
        }
    }
}
```

```

    }
    for (j = 0; j < 2; j++) {
        if (block->next[j] == NULL) {
            continue;
        }
        if (block->next[j]->signal == RED) {
            continue;
        }
        if (find_block(block->next[j], block->train->dest) == 1) {
            block->next[j]->train = block->train;
            block->next[j]->signal = RED;
            block->train = NULL;
            c;
        }
    }
}

int find_block(struct block_info* block, struct block_info* dest) {
    int i;
    if (block == dest) {
        return 1;
    }
    for (i = 0; i < 2; i++) {
        if (block->next[i] == NULL) {
            continue;
        }
        if (find_block(d) == 1) {
            return 1;
        }
    }
    return 0;
}

```

bに関する解答群

- | | |
|-------------------------------|-------------------------------|
| ア block->train->dest == NULL | イ block->train->dest != NULL |
| ウ block == block->train->dest | エ block != block->train->dest |

cに関する解答群

- | | | |
|---------|------------|----------|
| ア break | イ continue | ウ return |
|---------|------------|----------|

dに関する解答群

- | | |
|------------------------|-------------------------|
| ア dest, block | イ dest, block->next[i] |
| ウ block, dest | エ block, block->next[i] |
| オ block->next[i], dest | カ block->next[i], block |

設問 3 図 1 に示した鉄道模型の路線構成及び列車位置を表現した、構造体

block_info の配列を第 1 引数とし、その要素数 9 を第 2 引数として、プログラム 3 を実行した。プログラム 3 の実行が終了したときの状態について述べた次の記述中の [] に入る正しい答えを、解答群の中から選べ。ここで、blocks[n] は区間 n を示す。

[プログラム 3]

```
void start(struct block_info[], int);  
  
void start(struct block_info blocks[], int nblocks) {  
    int i;  
    set_signals(blocks, nblocks);  
    for (i = 0; i < 4; i++) {  
        proceed(blocks, nblocks);  
        set_signals(blocks, nblocks);  
    }  
}
```

列車 4 がいるのは区間 [] e である。また、区間 [] e の出口で接続する区間は [] f 。

eに関する解答群

- ア 0 イ 2 ウ 3 エ 7

fに関する解答群

- ア 一つあり、その信号機は緑を表示している
イ 一つあり、その信号機は赤を表示している
ウ 二つあり、それらの信号機はいずれも緑を表示している
エ 二つあり、それらの信号機はいずれも赤を表示している
オ 二つあり、それらの信号機は、一方が緑を、もう一方が赤を表示している

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 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]

従業員の社内資格の保有状況を管理するプログラムである。

情報サービス企業である P 社には社内資格として資格 1～4 があり、従業員に取得を奨励している。社内資格を取得するための試験（以下、資格試験という）は定期的に実施しており、全従業員の保有状況を保有資格ファイルで管理する。P 社の事業年度は 4 月から翌年 3 月までであり、上期（4 月～9 月）と下期（10 月～翌年 3 月）の 2 期から成る。資格試験に合格した従業員の情報は合格ファイルに 1 期分を蓄積する。上期の合格者を蓄積した合格ファイルは 10 月初めに、下期の合格者を蓄積した合格ファイルは 4 月初めにプログラムを実行して、保有資格ファイルに反映する運用である。従業員は、合格した資格試験を再度受験することはない。

- (1) 保有資格ファイルは、図 1 に示すレコード様式の索引ファイルであり、全従業員のレコードが格納されている。主キーは従業員番号である。プログラムでは、乱呼出し法で入出力している。

従業員番号 8 枠	保有状況			
	資格 1 8 枠	資格 2 8 枠	資格 3 8 枠	資格 4 8 枠

図 1 保有資格ファイルのレコード様式

資格 1～4 には、従業員番号で示される従業員が各資格試験に合格した日付として、西暦の年、月、日が、それぞれ 4 枠、2 枠、2 枠で格納されている。合格していない場合には、0 が格納されている。

- (2) 合格ファイルは、図 2 に示すレコード様式の順ファイルであり、資格試験に合格した従業員のレコードが格納されている。同じ従業員が複数の資格試験に合格した場合は、合格した資格試験ごとにレコードが格納されている。レコードは、順序

不同で格納されている。

従業員番号 8桁	資格種別 1桁	合格日 8桁
-------------	------------	-----------

図 2 合格ファイルのレコード様式

- ① 資格種別には、合格した資格試験を表す値が、資格 1 は 1 で、資格 2 は 2 で、資格 3 は 3 で、資格 4 は 4 で格納されている。
- ② 合格日には、合格した日付として、西暦の年、月、日が、それぞれ 4 桁、2 桁、2 桁で格納されている。
- (3) 保有資格ファイルには、合格ファイルの従業員番号で示される従業員のレコードが必ず存在する。

[プログラム]

(行番号)

```

1 DATA DIVISION.
2 FILE SECTION.
3 SD SRT-FILE.
4 01 SRT-REC.
5   02 SRT-NO      PIC X(8).
6   02 SRT-CD      PIC 9(1).
7   02 SRT-DATE    PIC 9(8).
8 FD QLF-FILE.
9 01 QLF-REC.
10  02 QLF-NO     PIC X(8).
11  02 QLF-INF.
12    03 QLF-DATE   PIC 9(8) OCCURS 4.
13 FD PAS-FILE.
14 01 PAS-REC.
15   02 PAS-NO     PIC X(8).
16   02 PAS-CD     PIC 9(1).
17   02 PAS-DATE   PIC 9(8).
18 WORKING-STORAGE SECTION.
19 77 SRT-FLAG      PIC X(1) VALUE SPACE.
20 88 SRT-EOF       VALUE "E".
21 77 CR-NO        PIC X(8) VALUE SPACE.

```

```

22 PROCEDURE DIVISION.
23 MAIN-PROC.
24   OPEN I-O QLF-FILE.
25   SORT SRT-FILE ASCENDING KEY SRT-NO
26     USING PAS-FILE
27     OUTPUT PROCEDURE IS RET-PROC.
28   CLOSE QLF-FILE.
29   STOP RUN.
30 RET-PROC.
31   PERFORM TEST BEFORE UNTIL SRT-EOF
32     RETURN SRT-FILE AT END      SET SRT-EOF TO TRUE
33           a
34     NOT AT END PERFORM UPD-PROC
35   END-RETURN
36   END-PERFORM.
37 UPD-PROC.
38   IF b THEN
39     PERFORM WRI-PROC
40     MOVE SRT-NO TO QLF-NO
41     READ QLF-FILE END-READ
42     MOVE QLF-NO TO CR-NO
43   END-IF.
44   c .
45 WRI-PROC.
46   IF CR-NO NOT = SPACE THEN
47     REWRITE QLF-REC
48   END-IF.

```

設問 1 プログラム中の [] 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 MOVE SPACE TO CR-NO | イ PERFORM UPD-PROC |
| ウ PERFORM WRI-PROC | エ REWRITE QLF-REC |

b に関する解答群

- | | |
|------------------|----------------------|
| ア CR-NO = SPACE | イ SRT-NO = CR-NO |
| ウ SRT-NO = SPACE | エ SRT-NO NOT = CR-NO |

cに関する解答群

- ア MOVE QLF-DATE(SRT-CD) TO QLF-DATE(SRT-CD + 1)
- イ MOVE SRT-DATE TO QLF-DATE(SRT-CD)
- ウ MOVE SRT-NO TO QLF-DATE(SRT-CD)
- エ MOVE ZERO TO QLF-DATE(SRT-CD)

設問2 P社では、社内資格を四つ全て取得した従業員を表彰している。合格ファイルの内容を保有資格ファイルに反映するとき、新たに表彰の対象となった従業員の従業員番号を表示するようにプログラムを変更する。表1中の
[]に入る正しい答えを、解答群の中から選べ。

表1 プログラムの変更内容1

処置	変更内容
行番号21と22の間に追加	77 W-CNT PIC 9(4).
[d]に追加	<pre> PERFORM VARYING W-CNT FROM 1 BY 1 UNTIL W-CNT > 4 OR QLF-DATE(W-CNT) = ZERO CONTINUE END-PERFORM IF [e] THEN DISPLAY "EMPLOYEE-NO:" QLF-NO " COMPLETE" END-IF </pre>

dに関する解答群

- | | |
|--------------|--------------|
| ア 行番号33と34の間 | イ 行番号35と36の間 |
| ウ 行番号42と43の間 | エ 行番号46と47の間 |

eに関する解答群

- | | |
|--------------------------|--------------------------|
| ア QLF-DATE(CR-NO) = ZERO | イ QLF-DATE(W-CNT) = ZERO |
| ウ QLF-INF NOT = ALL ZERO | エ W-CNT > 4 |

設問3 プログラムを表2に示すとおりに変更して、通常の運用どおり10月初めに実行した。行番号28と29の間に追加したDISPLAY文によって表示される内容に関する説明として適切な答えを、解答群の中から選べ。ここで、プログラムは設問2の変更を行う前のものとする。また、対象となる従業員の数は、10,000未満である。

表2 プログラムの変更内容2

処置	変更内容
行番号18と19の間に追加	01 W-LIC. 02 W-NUM PIC 9(4) OCCURS 4 VALUE ZERO.
行番号28と29の間に追加	DISPLAY W-NUM(1) ", " W-NUM(2) ", " W-NUM(3) ", " W-NUM(4).
行番号44と45の間に追加	ADD 1 TO W-NUM(SRT-CD).

解答群

- ア 当該事業年度の上期に、各資格試験に合格した従業員の数
- イ 当該事業年度の上期に、各資格試験を受験した従業員の数
- ウ 当該事業年度の上期までに、各資格試験に合格した従業員の数
- エ 当該事業年度の上期までに、各資格試験に合格していない従業員の数

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, 2 に答えよ。

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

[プログラムの説明]

図 1 のように、文書の書式を表すひな形に置換表を適用して、出力文書を得るプログラムである。

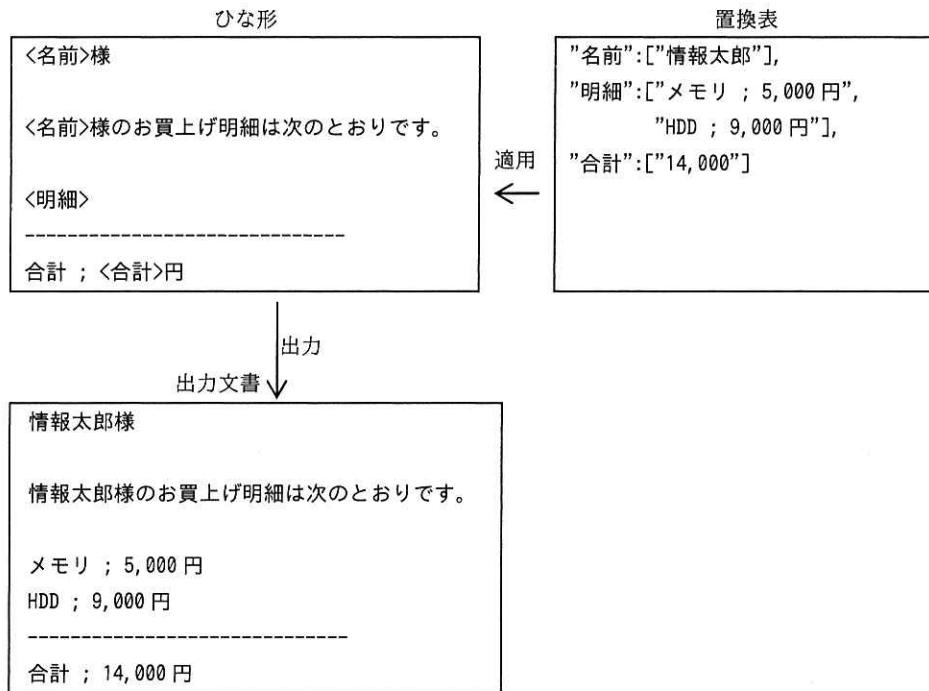


図 1 ひな形に置換表を適用して出力文書を得る例

- (1) ひな形は、0 個以上の置換指示と、0 個以上のそのまま出力される置換指示以外の部分が連なるテキストである。

置換指示は、キー名称を <と> で囲ったものである。ひな形中で、<と> は、置換指示としてキー名称を囲う用途にだけ使用できる。

- (2) 置換表は、キー名称とこれに対応する文字列の並びとの組みを一つ以上記述したテキストであり、それぞれの組みは、次の形式で記述する。

"キー名称":[文字列の並び]

組みを二つ以上記述するときは、コンマで区切る。

文字列の並びには、置換に用いる文字列（以下、置換用文字列という）を二重引用符で囲んだものを一つ以上記述する。置換用文字列を二つ以上記述するときは、コンマで区切る。例えば、置換用文字列が二つである場合は、次の形式で記述する。

"キー名称":["置換用文字列","置換用文字列"]

- (3) ひな形に置換表を適用すると、ひな形中の<キー名称>は、次の規則に従って置換される。

- ① 置換表のキー名称に対応する文字列の並びに含まれる置換用文字列が一つだけのときは、その置換用文字列で置換される。
- ② 置換表のキー名称に対応する文字列の並びに含まれる置換用文字列が二つ以上あるときは、各置換用文字列の間に改行を挟んだ上で、並び順に連結してできる文字列で置換される。

図 1 の例では、置換表中のキー名称名前に対応する文字列の並びに含まれる置換用文字列は情報太郎だけなので、ひな形中の<名前>は、情報太郎に置換される。

置換表中のキー名称明細に対応する文字列の並びには、置換用文字列としてメモリ；5,000円とHDD；9,000円とが含まれるので、ひな形中の<明細>は、メモリ；5,000円↵HDD；9,000円（↵は改行）に置換される。

このプログラムでは、ひな形を、0個以上の置換指示と0個以上の置換指示以外の部分が連なる文字ストリームとして扱う。個々の置換指示及び個々の置換指示以外の部分をフラグメントと呼ぶ。

インターフェース Fragment は、フラグメントを表す。

クラス Replacer は、置換指示を表す。

クラス PassThrough は、置換指示以外の部分を表す。

クラス `TemplateParser` のメソッド `parse` は、ひな形を表す文字ストリームからフラグメントのリストを構築し、クラス `Template` のインスタンスを生成して返す。ひな形に誤りはないものとする。

クラス `Template` は、ひな形をフラグメントのリストとして保持する。メソッド `apply` は、ひな形に置換表を適用して、出力文書を文字列で返す。置換表には、このひな形に含まれるキー名称とそれに対応する文字列の並びが含まれているものとする。

クラス `ReplacementTableParser` のメソッド `parse` は、置換表を表す文字ストリームから、キー名称とそれに対応する文字列の並びのマップを構築して、返す。置換表に誤りはないものとする（プログラムは省略）。

クラス `TemplateTester` はテスト用のプログラムである。テキストファイル `template.txt` が図 1 のひな形と同じ内容であって、テキストファイル `replacementTable.txt` が図 1 の置換表と同じ内容であるとき、実行結果は図 1 の出力文書と同じになる。

[プログラム 1]

```
import java.util.List;
import java.util.Map;

public interface Fragment {
    String replace(Map<String, List<String>> table);
}
```

[プログラム 2]

```
import java.util.List;
import java.util.Map;

public class Replacer [ ] Fragment {
    final String key;

    Replacer(CharSequence cs) { key = cs.toString(); }

    public String replace(Map<String, List<String>> table) {
        return String.join("\n", table.get(key));
    }
}
```

[プログラム 3]

```
import java.util.List;
import java.util.Map;

public class PassThrough [a] Fragment {
    final String str;

    PassThrough(CharSequence cs) { str = cs.toString(); }

    public String replace(Map<String, List<String>> table) {
        return str;
    }
}
```

[プログラム 4]

```
import java.io.IOException;
import java.io.Reader;
import java.util.ArrayList;
import java.util.List;

public class TemplateParser {
    static public Template parse(Reader reader) throws IOException {
        StringBuilder buf = new StringBuilder();
        List<Fragment> fragmentList = new ArrayList<>();
        int c;
        while ((c = reader.read()) >= 0) {
            switch (c) {
                case '<' :
                    fragmentList.add(new [b]);
                    buf = new StringBuilder();
                    break;
                case '>' :
                    fragmentList.add(new [c]);
                    buf = new StringBuilder();
                    break;
                default :
                    buf.append((char) c);
            }
        }
        fragmentList.add(new PassThrough(buf));
        return [d];
    }
}
```



```
    }
}

[プログラム 5]
import java.util.List;
import java.util.Map;

public class Template {
    List<Fragment> fragmentList;

    Template(List<Fragment> fragmentList) {
        this.fragmentList = fragmentList;
    }

    public String apply(Map<String, List<String>> table) {
        StringBuilder sb = new StringBuilder();
        for (Fragment fragment : fragmentList) {
            sb.append(fragment.replace("e"));
        }
        return sb.toString();
    }
}
```

```
[プログラム 6]
import java.io.FileReader;
import java.io.IOException;
import java.io.Reader;
import java.util.List;
import java.util.Map;

public class TemplateTester {
    public static void main(String... args) throws IOException {
        try {
            Reader tReader = new FileReader("template.txt");
            Reader rReader = new FileReader("replacementTable.txt")
        } {
            Template template = TemplateParser.parse(tReader);
            Map<String, List<String>> table =
                ReplacementTableParser.parse(rReader);
            System.out.print(template.apply(table));
        }
    }
}
```

設問 1 プログラム中の [] 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 extends イ interface ウ implements エ throws

b, c に関する解答群

- | | | |
|------------------|-----------------|--------------------|
| ア Fragment(buf) | イ Fragment(c) | ウ PassThrough(buf) |
| エ PassThrough(c) | オ Replacer(buf) | カ Replacer(c) |

d に関する解答群

- | | |
|------------------------------|------------------------|
| ア fragmentList | イ new Template(buf) |
| ウ new Template(fragmentList) | エ new Template(reader) |

e に関する解答群

- | | |
|------------|----------------|
| ア fragment | イ fragmentList |
| ウ sb | エ table |

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。

〈と〉は、置換指示としてキー名称を囲う用途以外では、ひな形中で使用することができない。そこで、これらの文字を他の用途でも使用できるように、次の 2 行をプログラム 4 のクラス TemplateParser の α の位置に挿入した。これによって、\に続く 1 文字 (\が複数個連続するときは奇数個目に続く 1 文字) は、置換指示以外の部分やキー名称の一部として扱われる。ここで、続く 1 文字は必ず読めるものとする。

```
case '\\':
    f;
```

f に関する解答群

- ア break
- イ buf.append((char) c)
- ウ buf.append((char) c); break
- エ buf.append((char) reader.read())
- オ buf.append((char) reader.read()); break

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, 2 に答えよ。

[プログラムの説明]

1970 年 1 月 1 日（以下、基準日という）から、指定された日付までの日数を求める副プログラム群である。日付は、全て西暦（グレゴリオ暦）の日付であり、基準日以降でなければならない。

- (1) プログラム 1 は、基準日から指定された日付までの日数を計算する副プログラム DAYOFFST である。日付は、GR2 に設定されたアドレスから始まる連続した 3 語に年、月、日を表す数値でこの順に格納され、誤りはないものとする。基準日を 0 日目とし、指定された日付までの日数を、符号のない数値（0～65535）として GR0 に設定して呼出し元に戻る。ただし、日付は基準日から 65535 日目までの日付で与えられる。
- (2) プログラム 2 は、うるう年を判定する副プログラム LEAPYEAR である。GR2 に設定された年が平年の場合は 0 を、うるう年の場合は 1 を、GR0 に設定して呼出し元に戻る。年が平年であるか、うるう年であるかの判定は、次の順に行う。
- ① 年が 4 で割り切れない場合、平年とする。
 - ② 年が 4 で割り切れ、かつ 100 で割り切れない場合、うるう年とする。
 - ③ 年が 400 で割り切れる場合、うるう年とする。
 - ④ ①～③のいずれでも決定しなかった場合、平年とする。
- (3) 副プログラム DIVISIBL は、GR2 に設定された整数値が GR3 に設定された整数値で割り切れる場合は 1 を、割り切れない場合は 0 を、GR0 に設定して呼出し元に戻る。プログラムのソースコードは、省略する。
- (4) 各副プログラムから戻るとき、汎用レジスタ GR1～GR7 の内容は元に戻す。

[プログラム 1]

(行番号)

```
1 DAYOFFST START
2          RPUSH
3          LD    GR5, 0, GR2 ; GR5: 年
4          LD    GR3, 1, GR2 ; GR3: 月
5          LD    GR1, 2, GR2 ; GR1: 日
6          SUBL GR1, =1      ; GR1で日数をカウント
7          a
8          ADDL GR1, -1, GR4 ; 1月1日からの日数（平年）を求める
9          CPA   GR3, =3     ; 月が3月以降のときうるう年を考慮
10         JMI   SKIP
11         LD    GR2, GR5
12         CALL  LEAPYEAR
13         ADDL GR1, GR0
14 SKIP    LD    GR2, =1970 ; 1970年から(年-1)年までの間（ただし,
15 LOOP   CPA   GR2, GR5 ; 年>1970），1年の日数を加算
16         b
17         CALL  LEAPYEAR
18         ADDL GR0, =365
19         ADDL GR1, GR0
20         ADDA GR2, =1
21         JUMP  LOOP
22 BREAK   LD    GR0, GR1
23 EXIT    RPOP
24        RET
25 ; ACCMDAYSは、平年の各月1日の1月1日からの日数（1月1日は0日目）
26 ACCMDAYS DC    0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334
27        END
```

アセンブリ

[プログラム 2]

(行番号)

```
1 LEAPYEAR START
2          RPUSH
3          SUBA GR0, GR0
4          LD    GR3, GR2
5          AND   GR3, =3
6          c
7          LD    GR3, =100
8          CALL  DIVISBL
9          d
```

10 JNZ FIN
11 LD GR3, =400
12 CALL DIVISBL
13 FIN RPOP
14 RET
15 END

設問1 プログラム中の 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 LAD GR4, ACCMDAYS | イ LAD GR4, ACCMDAYS, GR3 |
| ウ LAD GR4, ACCMDAYS, GR5 | エ LD GR4, ACCMDAYS |
| オ LD GR4, ACCMDAYS, GR3 | カ LD GR4, ACCMDAYS, GR5 |

bに関する解答群

- | | | |
|-------------|--------------|-------------|
| ア JMI BREAK | イ JNZ BREAK | ウ JOV BREAK |
| エ JPL BREAK | オ JUMP BREAK | カ JZE BREAK |

cに関する解答群

- | | | |
|------------|-----------|-----------|
| ア JMI FIN | イ JNZ FIN | ウ JOV FIN |
| エ JUMP FIN | オ JZE FIN | |

アセンブリ

dに関する解答群

- | | | |
|-------------------|-------------------|------------------|
| ア AND GR0,=#FFFFE | イ AND GR0,=1 | ウ OR GR0,=#FFFFE |
| エ OR GR0,=1 | オ XOR GR0,=#FFFFE | カ XOR GR0,=1 |

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。

副プログラム DAYOFFST に、基準日から 65536 日目以降 32767 年 12 月 31 日までの日付を与えた場合には、あふれが発生し、正しい結果を返すことができない。あふれが発生したことを呼出し元に通知するために、フラグレジスタ FR の OF が 1 の状態で呼出し元に戻るようにしたい。このためには、プログラム 1 の行番号 e の行の直後に f 命令を挿入し、ラベル EXIT の行に分岐する必要がある。

e に関する解答群

ア 8 イ 13 ウ 18 エ 19 オ 20

f に関する解答群

ア JMI イ JNZ ウ JOV エ JPL オ JZE

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]

Z 社は、店舗の窓口で自社商品に関する諸手続のサービスを行っている。来店した顧客が受付機のボタンを押すと、1 から始まる連番の受付番号を記した受付票が発行される。窓口では、受付番号の順に担当者が PC を用いて業務処理用サーバ（以下、業務サーバという）の機能を利用しながらサービスを行う。最近、顧客へのアンケートで“待ち時間が長い”との意見が増えたので、表計算ソフトを用いて待ち時間の状況などを分析することにした。

なお、本問において、関数“条件付個数”は、セル範囲に含まれる空白セルでない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。

[ワークシート：来店状況]

業務サーバには、顧客に対して行ったサービスに関するデータが受付番号ごとに記録される。この記録から分析に必要なデータを得るために、ある 1 日のデータを業務サーバから抽出、加工して、ワークシート“来店状況”に入力した。ワークシート“来店状況”的例を、図 1 に示す。

	A	B	C	D	E	F	G	H	I	J
1	窓口数	3								
2	受付番号	受付時刻	サービス種別	窓口番号	開始時刻	サービス時間	終了時刻	待ち時間	待ち人数	待ち時間順位
3	1	10:02	A	1	10:02	0:14	10:16	0:00	0	78
4	2	10:08	A	2	10:08	0:13	10:21	0:00	0	79
5	3	10:13	C	3	10:13	0:22	10:35	0:00	0	80
:	:	:	:	:	:	:	:	:	:	:
109	110	18:52	B	1	19:55	0:09	20:04	1:03	13	38
110										
:	:	:	:	:	:	:	:	:	:	:
252										

図 1 ワークシート“来店状況”的例

- (1) 列 A～F の行 3 以降に、業務サーバから抽出した、受付票の発行後にサービスを受けた顧客（以下、窓口利用者という）のデータが入力されている。
- データは、一つの受付番号に関するものが 1 レコードとして同一行に入力され、受付番号の昇順に整列されている。データの件数は 5 件以上 250 件以下であり、最後のデータが入力された行よりも下の行の各セルには空値が格納されている。
- (2) 業務サーバは、時間を分の単位の整数値で保持し、時刻を午前 0 時 00 分からの経過分数として保持する。時間及び時刻の計算にはこの整数値を用いる。時間及び時刻は、hh:mm（時:分）の形式で表示する。
- (3) 列 A の受付番号は受付票に記された番号であり、列 B の受付時刻はその受付票が発行された時刻である。受付機による受付票の発行は 10 時 00 分になると同時に開始し、19 時 00 分になると同時に終了する。
- (4) 列 C のサービス種別は、窓口利用者に対して行ったサービスの種別であり、A, B, C の三つがある。全てのサービスは、1 日に 1 回以上利用される。
- (5) 列 D の窓口番号は、サービスを行った窓口の番号である。窓口は三つあり、それらの窓口番号は 1, 2, 3 である。どの窓口でも全ての種別のサービスを行っている。
- (6) 列 E の開始時刻は窓口でサービスを開始した時刻、列 F のサービス時間はサービスに掛かった時間である。
- (7) セル G3 に、開始時刻にサービス時間を加えて終了時刻を求める次の式を入力し、セル G4～G252 に複写する。

$IF(A3=null,null,E3+F3)$

- (8) セル H3 に、受付時刻から開始時刻までの時間（以下、待ち時間という）を求める次の式を入力し、セル H4～H252 に複写する。

$IF(A3=null,null,E3-B3)$

- (9) セル I3 に、当該行の受付番号よりも小さい受付番号をもつ窓口利用者のうち、当該行の受付時刻にまだサービスが開始されていない人数（以下、待ち人数という）を求める次の式を入力し、セル I4～I252 に複写する。ここで、待ち時間が 0 の窓口利用者は、待ち人数に含めない。

$IF(A3=null,null,[a])$

- (10) セル J3 に、列 H の待ち時間の値を大きい順に順位付けしたときの当該行の順

位（以下、待ち時間順位という）を求める次の式を入力し、セル J4～J252 に複写する。ここで、待ち時間の等しいデータが複数あるときは、受付番号の小さいデータを上位とする。

IF(A3=null,null,条件付個数(H\$3:H\$252,>H3)
+条件付個数(H\$3:H3,=H3))

設問 1 は問題誤りにつき、問13を選択した受験者全員正解の措置済み

設問 1 ワークシート“来店状況”に関する記述中の 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 条件付個数(E3:E\$252,>B3) | イ 条件付個数(E\$3:E3,<B3) |
| ウ 条件付個数(E\$3:E3,>B3) | エ 条件付個数(G3:G\$252,>B3) |
| オ 条件付個数(G3:G\$252,<B3) | カ 条件付個数(G\$3:G3,>B3) |

[ワークシート：分析]

ワークシート“来店状況”的データを基に窓口利用者の状況を分析するために、ワークシート“分析”を作成し、セル A2～D12 に“時間帯別分析”，セル F2～I6 に“種別分析”，セル F9～I14 に“待ち時間分析（上位 5 件）”の表を格納した。ワークシート“分析”的例を、図 2 に示す。

	A	B	C	D	E	F	G	H	I					
1	時間帯別分析													
2	受付時間帯		受付 人数	平均待 ち時間	種別分析									
3	から	まで												
4	10:00	10:59	7	0:00										
5	11:00	11:59	9	0:00										
6	12:00	12:59	9	0:00										
7	13:00	13:59	12	0:00										
8	14:00	14:59	18	0:14										
9	15:00	15:59	21	0:50										
10	16:00	16:59	15	1:27										
11	17:00	17:59	9	1:32										
12	18:00	18:59	7	1:17										
13														
14														

	サービス種別	件数	構成比率	平均サービス時間
A	62	57.9%	0:14	
B	29	27.1%	0:08	
C	16	15.0%	0:24	

待ち時間分析（上位 5 件）			
順位	受付番号	受付時刻	待ち時間
1	92	16:59	1:38
2	97	17:24	1:37
3	99	17:34	1:37
4	91	16:57	1:35
5	93	17:02	1:35

図 2 ワークシート“分析”的例

設問2 ワークシート“分析”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) 集計の区分を1時間ごとの時間帯とし、セルA4～B12に、各時間帯の始まりの時刻（毎時の00分）と終わりの時刻（毎時の59分）を上から順に入力する。また、セルF4～F6にサービスの種別を、セルF10～F14に待ち時間の順位を示す数値1～5を入力する。
- (2) セルC4に、ワークシート“来店状況”的データから、受付時刻が当該行の時間帯にあるデータの個数を求める次の式を入力し、セルC5～C12に複写する。

[] b

- (3) セルD4に、ワークシート“来店状況”的データから、受付時刻が当該行の時間帯にあるデータの待ち時間の平均値を求める式を入力し、セルD5～D12に複写する。ここで、当該行の時間帯の受付人数が0のときの待ち時間の平均値は0とする。
- (4) セルG4に、ワークシート“来店状況”的データから、サービス種別が当該行のサービス種別と一致するデータの件数を求める式を入力し、セルG5～G6に複写する。また、セルH4に全サービス件数に対する当該行の件数の構成比率を求める式を入力し、セルH5～H6に複写する。
- (5) セルI4に、ワークシート“来店状況”的データから、サービス種別が当該行のサービス種別と一致するデータのサービス時間の平均値を求める次の式を入力し、セルI5～I6に複写する。

切捨て([] c ,0)

- (6) セルG10に、ワークシート“来店状況”的データから、待ち時間順位がセルF10の順位に該当するデータの受付番号を求める次の式を入力し、セルG11～G14に複写する。

[] d

- (7) セルH10に、ワークシート“来店状況”的データから、セルG10の受付番号に該当するデータの受付時刻を求める式を入力し、セルH11～H14に複写する。

(8) セル I10 に、ワークシート“来店状況”のデータから、セル G10 の受付番号に該当するデータの待ち時間を求める式を入力し、セル I11～I14 に複写する。

b に関する解答群

- ア 条件付個数(来店状況!B\$3:B\$252, >A4)
 - －条件付個数(来店状況!B\$3:B\$252, >A5)
- イ 条件付個数(来店状況!B\$3:B\$252, >A4)
 - －条件付個数(来店状況!B\$3:B\$252, >B4)
- ウ 条件付個数(来店状況!B\$3:B\$252, >B4)
 - －条件付個数(来店状況!B\$3:B\$252, >A4)
- エ 条件付個数(来店状況!B\$3:B\$252, <A4)
 - －条件付個数(来店状況!B\$3:B\$252, ≤B4)
- オ 条件付個数(来店状況!B\$3:B\$252, ≤B4)
 - －条件付個数(来店状況!B\$3:B\$252, <A4)

c に関する解答群

- ア 合計(来店状況!F\$3:F\$252) / G4 * H4
- イ 条件付合計(来店状況!C\$3:C\$252, =F4, 来店状況!F\$3:F\$252) / G4
- ウ 条件付合計(来店状況!C\$3:C\$252, =F4, 来店状況!F\$3:F\$252) / 合計(G\$4:G\$6)
- エ 条件付合計(来店状況!F\$3:F\$252, =F4, 来店状況!C\$3:C\$252) / G4
- オ 条件付合計(来店状況!F\$3:F\$252, =F4, 来店状況!C\$3:C\$252) / 合計(G\$4:G\$6)
- カ 平均(来店状況!F\$3:F\$252) * H4

d に関する解答群

- ア 照合検索(F10, 来店状況!A\$3:A\$252, 来店状況!J\$3:J\$252)
- イ 照合検索(F10, 来店状況!J\$3:J\$252, 来店状況!A\$3:A\$252)
- ウ 垂直照合(順位(F10, 来店状況!H\$3:H\$252, 1), 来店状況!A\$3:J\$252, 1, 0)
- エ 垂直照合(F10, 来店状況!A\$3:J\$252, 10, 0)
- オ 表引き(来店状況!A\$3:A\$252, 順位(F10, 来店状況!H\$3:H\$252, 1), 1)
- カ 表引き(来店状況!J\$3:J\$252, 照合一致(F10, 来店状況!A\$3:A\$252, 0), 1)

[マクロ : queue_simulation の説明]

待ち時間の短縮策を検討するために、窓口数を変えたときのサービスを行う窓口の窓口番号、開始時刻、終了時刻、待ち時間及び待ち人数の変化をシミュレーションするマクロ `queue_simulation` を作成し、ワークシート“来店状況”に格納した。シミュレーションに使う窓口数をワークシート“来店状況”的セル B1 に入力し、受付番号、受付時刻、サービス種別及びサービス時間は、ワークシート“来店状況”的セル A3 ~ F252 の内容をそのまま用いる。窓口数として、1~8 を指定できる。

- (1) 各窓口でサービスを受けている窓口利用者へのサービスが終了する時刻（以下、前者終了時刻という）を格納する配列 `end_time` を用意する。配列 `end_time` の添字の値 1~8 は窓口番号の 1~8 と対応する。各要素の初期値として 0 を格納する。
- (2) 処理する行（以下、処理行という）の位置を格納する変数 `work_line` を用意し、初期値を 1 とする。
- (3) ワークシート“来店状況”的行 3 のデータから処理を始め、全てのデータの処理が終わるまで、①~④の処理を繰り返す。
 - ① サービスを受ける窓口を決定するために、配列 `end_time` に格納されている前者終了時刻の中で最小の値をもつ要素を探し、その添字の値を窓口番号として処理行の列 D 的セルに格納する。ここで、前者終了時刻が等しい窓口が複数あるときは、窓口番号が最小の窓口を選択する。
 - ② ①で求めた前者終了時刻の最小の値が処理行の列 B の受付時刻よりも小さいときは受付時刻を、そうでなければ①で求めた前者終了時刻の最小の値を、開始時刻として処理行の列 E 的セルに格納する。
 - ③ ①で決定した窓口の前者終了時刻を更新するために、配列 `end_time` の当該窓口に対応する要素に、処理行の列 G の終了時刻を格納する。
 - ④ 変数 `work_line` に 1 を加える。

設問3 マクロ queue_simulation 中の に入る正しい答えを、解答群の中から選べ。

[マクロ : queue_simulation]

○マクロ: queue_simulation

○数値型: i, work_line, sw, end_time[9], min_time, min_no

■ i: 1, $i \leq B1, 1$

 ・ $\text{end_time}[i] \leftarrow 0$

■

 ・ $\text{work_line} \leftarrow 1$

 ・ $sw \leftarrow 0$

■ 論理積 ($sw = 0, \text{work_line} \leq 250$)

 ・ $\text{min_no} \leftarrow 1$

 ・ $\text{min_time} \leftarrow \text{end_time}[1]$

 ■ i: 2, $i \leq B1, 1$

 ↑

 ・ $\text{min_time} \leftarrow \text{end_time}[i]$

 ・ $\text{min_no} \leftarrow i$

■

 ・ 相対 (A2, work_line, 3) \leftarrow

 ↑ 相対 (A2, work_line, 1) $> \text{min_time}$

 ・ 相対 (A2, work_line, 4) \leftarrow 相対 (A2, work_line, 1)

 ・ 相対 (A2, work_line, 4) $\leftarrow \text{min_time}$

■

 ・

 ・ $\text{work_line} \leftarrow \text{work_line} + 1$

 ↑ 相対 (A2, work_line, 0) = null

 ・ $sw \leftarrow 1$

■

eに関する解答群

- ア `min_time < end_time[i]`
- イ `min_time < end_time[min_no]`
- ウ `min_time < 相対(A2, work_line, 1)`
- エ `min_time > end_time[i]`
- オ `min_time > end_time[min_no]`
- カ `min_time > 相対(A2, work_line, 1)`

fに関する解答群

- | | |
|----------------------------|---------------------------------|
| ア <code>end_time[i]</code> | イ <code>end_time[min_no]</code> |
| ウ <code>i</code> | エ <code>min_no</code> |
| オ <code>min_time</code> | カ <code>work_line</code> |

gに関する解答群

- ア `end_time[min_no] ← min_time`
- イ `end_time[min_no] ← 相対(A2, work_line, 4)`
- ウ `end_time[min_no] ← 相対(A2, work_line, 6)`
- エ `end_time[work_line] ← min_time`
- オ `end_time[work_line] ← 相対(A2, work_line, 4)`
- カ `end_time[work_line] ← 相対(A2, work_line, 6)`

■ Java プログラムで使用する API の説明

java.io

public abstract class Reader

文字ストリームを読み込むための抽象クラスである。

メソッド

public int read()

单一の文字を読み込む。

戻り値：読み込まれた文字（char値）をint型に変換した0から65535の範囲の値

ストリームの終わりに達した後は -1

例外： IOException — 入出力エラーが発生したとき

java.io

public class FileReader

テキストファイルを読み込むためのクラスである。抽象クラスReaderを継承する。

コンストラクタ

public FileReader(String fileName)

読み込み元のファイルの名前を指定して、新しいFileReaderを作成する。

引数： fileName — 読込み元のファイルの名前

例外： FileNotFoundException — 指定されたファイルが存在しないなどの理由で開くことができないとき。

FileNotFoundExceptionは、IOExceptionを継承する。

java.util

public interface List<E>

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

メソッド

public boolean add(E e)

指定された要素をリストの最後に追加する。

引数： e — リストに追加する要素

戻り値： true

```
java.util
public class ArrayList<E>
    インタフェース List の配列による実装である。
    メソッドの説明は、インターフェース List の項を参照。
```

コンストラクタ

```
public ArrayList()
    空のリストを作る。
```

```
java.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ー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。このキーに対応付けられた値がなければ、null
    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
            このキーに対応付けられた値がなければ null
```

```
java.lang
public interface CharSequence
    char 値のシーケンスを表すインターフェースを提供する。
```

```
java.lang
public class StringBuilder
    char 値の可変のシーケンスを表す。インターフェース CharSequence を実装する。
```

コンストラクタ

```
public StringBuilder()
    空のシーケンスを作る。
```

メソッド

```
public StringBuilder append(char c)
    このシーケンスの末尾に、指定された char 値を追加する。
    引数： c — このシーケンスに追加する char 値
    戻り値：このオブジェクトへの参照
```

```
public StringBuilder append(CharSequence cs)
    このシーケンスの末尾に、指定された char 値のシーケンスを追加する。
    引数： cs — このシーケンスに追加する char 値のシーケンス
    戻り値：このオブジェクトへの参照
```

```
java.lang
public final class String
    文字列を表す。インターフェース CharSequence を実装する。
```

メソッド

```
-----  
public static String join(CharSequence delimiter,
```

```
    Iterable<? extends CharSequence> elements)
```

`elements` で指定された反復子が返す全ての `char` 値のシーケンスを、間に `delimiter` で指定された `char` 値のシーケンスを挟んだ上で連結し、そこに含まれる全ての `char` 値を順に並べた文字列を返す。

例えば、`delimiter` が文字列 ":" で、`elements` が三つの文字列 "abc", "def", "ghi" を順に返す反復子であるとき、このメソッドは文字列 "abc:def::ghi" を返す。

`elements` が一つもシーケンスを返さないときは、空文字列を返す。`elements` が返すシーケンスが一つだけのときは、そのシーケンスに含まれる全ての `char` 値を、順に並べた文字列を返す。

引数： `delimiter` — `element` が返すシーケンスを連結する際に間に挟む `char` 値のシーケンス

`elements` — 連結対象の `char` 値のシーケンスを返す反復子

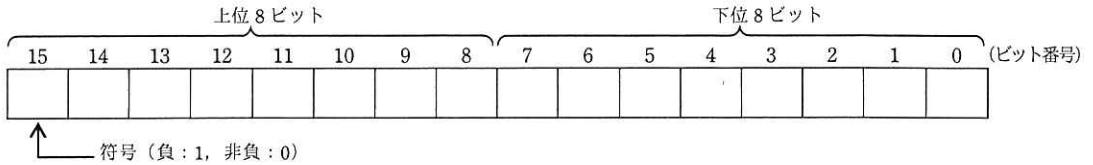
戻り値：連結したシーケンスに含まれる全ての `char` 値を順に並べた文字列

■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 ベ ラ ン ド		

(1) ロード、ストア、ロードアドレス命令

ロード LoaD	LD	r_1, r_2 $r, adr [, x]$	$r_1 \leftarrow (r_2)$ $r \leftarrow (\text{実効アドレス})$	○*1
ストア STore	ST	$r, adr [, x]$	$\text{実効アドレス} \leftarrow (r)$	-
ロードアドレス Load ADDress	LAD	$r, adr [, x]$	$r \leftarrow \text{実効アドレス}$	

(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	$r_1 \leftarrow (r_1) -_L (r_2)$	○*1
		$r, \text{adr } [, x]$	$r \leftarrow (r) -_L (\text{実効アドレス})$	
論理積 AND	AND	r_1, r_2	$r_1 \leftarrow (r_1) \text{ AND } (r_2)$	
		$r, \text{adr } [, x]$	$r \leftarrow (r) \text{ AND } (\text{実効アドレス})$	
論理和 OR	OR	r_1, r_2	$r_1 \leftarrow (r_1) \text{ OR } (r_2)$	
		$r, \text{adr } [, x]$	$r \leftarrow (r) \text{ OR } (\text{実効アドレス})$	
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	(r_1) と (r_2) 、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>$(r_1) > (r_2)$</td> <td>0</td> <td>0</td> </tr> <tr> <td>$(r) > (\text{実効アドレス})$</td> <td>0</td> <td>1</td> </tr> <tr> <td>$(r_1) = (r_2)$</td> <td>1</td> <td>0</td> </tr> <tr> <td>$(r) = (\text{実効アドレス})$</td> <td>1</td> <td>0</td> </tr> <tr> <td>$(r_1) < (r_2)$</td> <td>0</td> <td>1</td> </tr> <tr> <td>$(r) < (\text{実効アドレス})$</td> <td>0</td> <td>1</td> </tr> </tbody> </table>		比較結果	FR の値			SF	ZF	$(r_1) > (r_2)$	0	0	$(r) > (\text{実効アドレス})$	0	1	$(r_1) = (r_2)$	1	0	$(r) = (\text{実効アドレス})$	1	0	$(r_1) < (r_2)$	0	1	$(r) < (\text{実効アドレス})$	0	1
比較結果	FR の値																											
	SF	ZF																										
$(r_1) > (r_2)$	0	0																										
$(r) > (\text{実効アドレス})$	0	1																										
$(r_1) = (r_2)$	1	0																										
$(r) = (\text{実効アドレス})$	1	0																										
$(r_1) < (r_2)$	0	1																										
$(r) < (\text{実効アドレス})$	0	1																										
$r, \text{adr } [, x]$																												

(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。	○*2
算術右シフト Shift Right Arithmetic	SRA	$r, \text{adr } [, x]$	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。	
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。	
論理右シフト Shift Right Logical	SRL	$r, \text{adr } [, x]$	シフトの結果、空いたビット位置には 0 が入る。	

(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 MIinus	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	$\text{adr } [, x]$		

(6) スタック操作命令

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

(7) コール, リターン命令

コール CALL subroutine	CALL adr [, x]	SP \leftarrow (SP) $-_L$ 1, (SP) \leftarrow (PR), PR \leftarrow 実効アドレス	—
リターン RETurn from subroutine	RET	PR \leftarrow ((SP)), SP \leftarrow (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
 \leftarrow 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 (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
アセンブラ命令	ラベル	START	[実行開始番地]	プログラムの先頭を定義 プログラムの実行開始番地を定義 他のプログラムで参照する入口名を定義
		END		プログラムの終わりを明示
	[ラベル]	DS	語数	領域を確保
	[ラベル]	DC	定数 [, 定数] …	定数を定義
マクロ命令	[ラベル]	IN	入力領域, 入力文字長領域	入力装置から文字データを入力
	[ラベル]	OUT	出力領域, 出力文字長領域	出力装置へ文字データを出力
	[ラベル]	RPUSH		GR の内容をスタックに格納
	[ラベル]	RPOP		スタックの内容を GR に格納
機械語命令	[ラベル]			(「1.2 命令」を参照)

2.3 アセンブラ命令

アセンブラ命令は、アセンブラの制御などを行う。

- (1) START [実行開始番地]

START 命令は、プログラムの先頭を定義する。

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

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

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

END 命令は、プログラムの終わりを定義する。

(3)	DS	語数
-----	----	----

DS 命令は、指定した語数の領域を確保する。

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

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

DC 命令は、定数で指定したデータを（連続する）語に格納する。

定数には、10進定数、16進定数、文字定数、アドレス定数の 4種類がある。

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

2.4 マクロ命令

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

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

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

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

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

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

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

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

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

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

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

(3)

RPUSH	
-------	--

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

(4)

RPOP	
------	--

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

2.5 機械語命令

機械語命令のオペランドは、次の形式で記述する。

r, r1, r2 GR は、記号 GR0 ~ GR7 で指定する。

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

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

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

2.6 その他

- (1) アセンブラ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、^{けい}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” 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。
四捨五入 (算術式 , 桁位置)	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。
切捨て (算術式 , 桁位置)	セル範囲の中での算術式の値の順位を、順序の指定が0の場合には昇順で、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]

[× 用 紙]

6. 退室可能時間中に退室する場合は、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、アセンブラー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、会場での貸出ちは行っていません。
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。