

令和元年度 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。
 - 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。
 - 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。
 - 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。
 - 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、[] 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。

[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	[]	問 11	[]
問 3	[]	問 4	[]	問 5	(選)	問 6	[]	問 7	[]	問 12	[]
問 13	[]	問 14	[]	問 15	[]	問 16	[]	問 17	[]	問 18	[]
問 19	[]	問 20	[]	問 21	[]	問 22	[]	問 23	[]	問 24	[]

C

COBOL

Java

アセンブラー

表計算

[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	テレワークの導入

●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	スレッドを使用した並列実行
問 3	データベース	書籍及び貸出情報を管理する関係データベースの設計及び運用
問 4	ネットワーク	NAT
問 5	ソフトウェア設計	ストレスチェックの検査支援を行うシステム
問 6	プロジェクトマネジメント	販売管理システム開発の結合テストにおける進捗及び品質管理
問 7	経営戦略・企業と法務	製品別の収益分析

●問 8 (必須問題)

問題番号	出題分野	テーマ
問 8	データ構造及びアルゴリズム	Bitap 法による文字列検索

●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)	入力ファイルの内容を文字及び 16 進数で表示
問 10	ソフトウェア開発 (COBOL)	スーパーマーケットの弁当の販売データの集計
問 11	ソフトウェア開発 (Java)	通知メッセージの配信システム
問 12	ソフトウェア開発 (アセンブラー)	パック 10 進数の加算
問 13	ソフトウェア開発 (表計算)	メロンの仕分

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	↑ ↓ 高 低
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 テレワークの導入に関する次の記述を読んで、設問1～3に答えよ。

ソフトウェア開発会社であるA社では、従業員が働き方を柔軟に選択できるように、場所や時間の制約を受けずに働く勤務形態であるテレワークを導入することにした。

A社には、事務業務だけが行えるPC（以下、事務PCという）と、事務業務及びソフトウェア開発業務が行えるPC（以下、開発PCという）がある。開発部の従業員は開発PCを使用し、開発部以外の従業員は事務PCを使用している。

A社には事務室、開発室及びサーバ室があり、各部屋のネットワークはファイアウォール（以下、A社FWという）を介して接続されている。A社のネットワーク構成を、図1に示す。

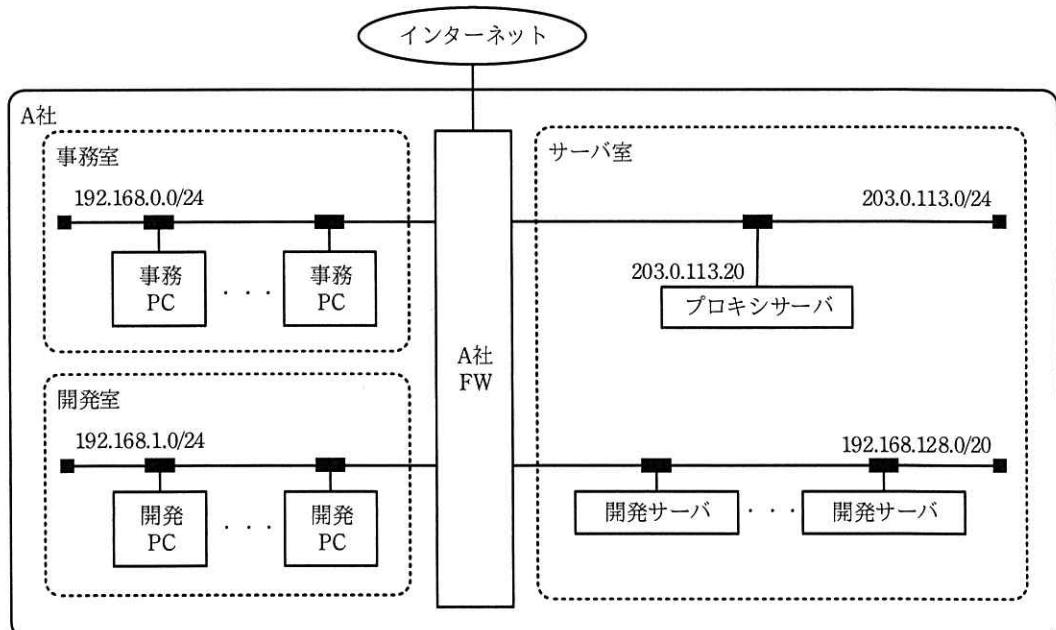


図1 A社のネットワーク構成

事務室には、事務 PC だけが設置されている。開発室には開発 PC だけが設置されており、開発部の従業員だけが入退室できる。サーバ室には、プロキシサーバ 1 台と、ソフトウェア開発業務に必要なソースコード管理、バグ管理、テストなどに利用するサーバ（以下、開発サーバという）が複数台設置されている。

A 社 FW では、開発室のネットワークだから開発サーバに HTTP over TLS（以下、HTTPS という）又は SSH でアクセスできるように通信を制限している。また、A 社ネットワークからのインターネットの Web サイト閲覧は、事務 PC 及び開発 PC だからプロキシサーバを経由してできるように通信を制限している。

テレワークで働く従業員は、データを保存できないシンクライアント端末を A 社から支給され、遠隔からインターネットを経由して A 社のネットワークに接続し、業務を行う。そのために、安全に A 社のネットワークに接続する VPN、及び仮想マシンの画面を転送して遠隔から操作できるようにする画面転送型の仮想デスクトップ環境（以下、VDI という）の導入を検討した。テレワーク導入後の A 社のネットワーク構成案を、図 2 に示す。

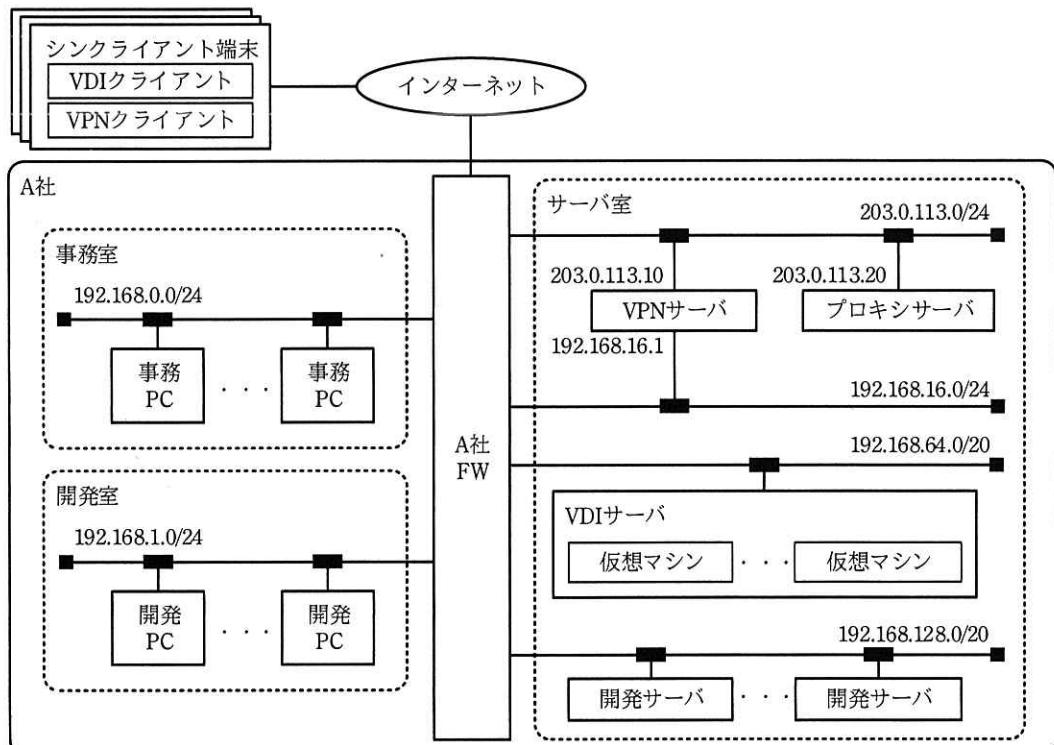


図 2 テレワーク導入後の A 社のネットワーク構成案

[A 社が検討したテレワークによる業務の開始までの流れ]

- (1) 利用者は、シンクライアント端末の VPN クライアントを起動して、VPN サーバに接続する。
- (2) VPN サーバは、VPN クライアントが提示するクライアント証明書を検証する。検証に成功した場合、処理を継続する。
- (3) VPN サーバは、利用者を認証する。認証が成功した場合、VPN クライアントに対して、192.168.16.0/24 の範囲で使用されていない IP アドレスを一つ選択して割り当てる。
- (4) VPN クライアントは、(3)で割り当てられた IP アドレスを使用して、VPN サーバ経由で A 社のネットワークに接続する。
- (5) 利用者は、シンクライアント端末の VDI クライアントを起動して、VDI サーバに接続する。
- (6) VDI サーバは、VPN サーバで認証された利用者が開発部以外の従業員であれば事務業務だけが行える仮想マシン（以下、事務 VM という）を、開発部の従業員であれば事務業務及びソフトウェア開発業務が行える仮想マシン（以下、開発 VM という）を割り当てる。また、VDI サーバは、事務 VM には 192.168.64.0/24、開発 VM には 192.168.65.0/24 の範囲で使用されていない IP アドレスを一つ選択して割り当てる。
- (7) 利用者は、仮想マシンにログインして業務を開始する。VDI クライアントと仮想マシンとの間では、画面データ、並びにキーボード及びマウスの操作データだけが送受信される。

テレワーク導入後の A 社 FW に設定するパケットフィルタリングのルール案を、表 1 に示す。

表1 A社FWに設定するパケットフィルタリングのルール案

ルール番号	送信元	宛先	サービス	動作
1	インターネット	203.0.113.10	VPN	許可
2	203.0.113.20	インターネット	HTTP, HTTPS, DNS	許可
3	192.168.16.0/24	192.168.64.0/20	VDI	許可
4	192.168.0.0/23	203.0.113.20	プロキシ	許可
5	192.168.64.0/23	203.0.113.20	プロキシ	許可
6	192.168.1.0/24	192.168.128.0/20	HTTPS, SSH	許可
7	192.168.64.0/23	192.168.128.0/20	HTTPS, SSH	許可
8	全て	全て	全て	拒否

注記1 ルール番号の小さいものから順に、最初に一致したルールが適用される。

注記2 許可された通信に対する戻りのパケットは、無条件に許可される。

ところが、表1のルール案ではルール番号7の条件に誤りがあり、[a] これが分かった。そこで、開発サーバに対するアクセスを正しく制限するために、ルール番号7の条件について、送信元を [b] に変更した。

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 開発PCから開発サーバにアクセスできない
- イ 開発VMから開発サーバにアクセスできない
- ウ 事務PCから開発サーバにアクセスできる
- エ 事務VMから開発サーバにアクセスできる

bに関する解答群

- | | | |
|--------------------|-------------------|--------------------|
| ア 192.168.0.0/24 | イ 192.168.1.0/24 | ウ 192.168.16.0/24 |
| エ 192.168.64.0/24 | オ 192.168.65.0/24 | カ 192.168.128.0/20 |
| キ 192.168.128.0/24 | ク 203.0.113.0/24 | ケ インターネット |

設問2 シンクライアント端末から開発サーバにアクセスするときの接続経路として
適切な答えを、解答群の中から選べ。

解答群

- ア シンクライアント端末 → VDI サーバ → VPN サーバ → 開発 PC → 開発サーバ
- イ シンクライアント端末 → VDI サーバ → VPN サーバ → 開発 VM → 開発サーバ
- ウ シンクライアント端末 → VDI サーバ → 開発 VM → 開発 PC → 開発サーバ
- エ シンクライアント端末 → VPN サーバ → VDI サーバ → 開発 PC → 開発サーバ
- オ シンクライアント端末 → VPN サーバ → VDI サーバ → 開発 VM → 開発サーバ
- カ シンクライアント端末 → VPN サーバ → 開発 PC → 開発 VM → 開発サーバ

設問3 A 社がテレワークの検討を進める過程で、“常に同一の業務環境を使用できるように、テレワークで働くときだけでなく、事務 PC 及び開発 PC からも仮想マシンを使用したい”との要望が挙がった。検討した結果、この要望に応えてもセキュリティ上のリスクは変わらないと判断した。また、A 社のネットワーク内からアクセスするので VPN で接続する必要はなく、利用者認証を VPN サーバではなく VDI サーバで行えばよいことを確認した。

この要望に応えるとき、表1のルール案に必要な変更として適切な答えを、解答群の中から選べ。ここで、表1のルール番号7の送信元には、設問1で選択した適切な答えが設定されているものとする。

解答群

- ア 変更する必要はない。
- イ ルール番号3と4の間に、送信元を 192.168.0.0/23、宛先を 192.168.64.0/20、サービスを VDI、及び動作を許可とするルールを新たに挿入する必要がある。
- ウ ルール番号3と4の間に、送信元を 192.168.64.0/23、宛先を 192.168.0.0/23、サービスを VDI、及び動作を許可とするルールを新たに挿入する必要がある。
- エ ルール番号3と4の間に、送信元をインターネット、宛先を 192.168.64.0/20、サービスを VDI、及び動作を許可とするルールを新たに挿入する必要がある。

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 スレッドを使用した並列実行に関する次の記述を読んで、設問1～3に答えよ。

プログラム中の並列実行が可能な部分を取り出し、その部分を分割して複数のスレッドで並列に実行する方法（以下、スレッド並列法という）がある。マルチプロセッサシステムでは、スレッド並列法を適用することによって、プログラムの実行時間を短縮できことがある。

プログラムにおいて、スレッド並列法を適用しないで実行したときの実行時間と、スレッド並列法を適用したときの実行時間で割った値を、プログラム実行時間の高速化率という。

プログラムをスレッド並列法を適用しないで実行したときの、プログラム全体の実行時間に対する、並列実行可能な部分の実行時間の割合を r ($0 \leq r \leq 1$) とする。スレッドの個数を n ($n \geq 1$) にして、プログラムにスレッド並列法を適用すると、マルチプロセッサシステムでは、プログラム実行時間の高速化率 E は、次の式で求められる。ここで、各スレッドはそれぞれ異なるプロセッサに割り当てられるものとし、プログラムの実行に使用する全てのプロセッサの性能は同じとする。

$$E = \frac{1}{(1 - r) + \frac{r}{n}}$$

この式は、並列実行可能な部分のプログラム実行時間がスレッド並列法の適用によって $\frac{1}{n}$ になり、その他の部分のプログラム実行時間は変化しないときの高速化率を計算するものである。

プログラム中に並列実行が可能な部分をもつプログラム A に対してスレッドの個数を 2 にしてスレッド並列法を適用すると、高速化率は $\frac{3}{2}$ になった。この場合、 r

は a である。

r が $\frac{3}{4}$ であるプログラム B の場合、スレッドの個数を増やしても、高速化率の上限は b である。

設問 1 本文中の 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 $\frac{1}{6}$

イ $\frac{1}{4}$

ウ $\frac{1}{3}$

エ $\frac{1}{2}$

オ $\frac{2}{3}$

b に関する解答群

ア 2

イ 3

ウ 4

エ 6

オ 8

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。

配列の操作を行う繰返しの処理において、図 1、図 2 のように繰返しの範囲を分割して、スレッド並列法を適用することを考える。

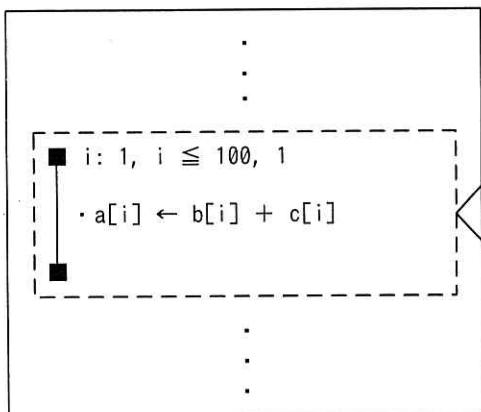
このとき、操作の内容によって、正しい結果が得られる場合と得られない場合があるので、十分に検討することが必要である。

正しい結果が得られる場合の例を、図 1 に示す。

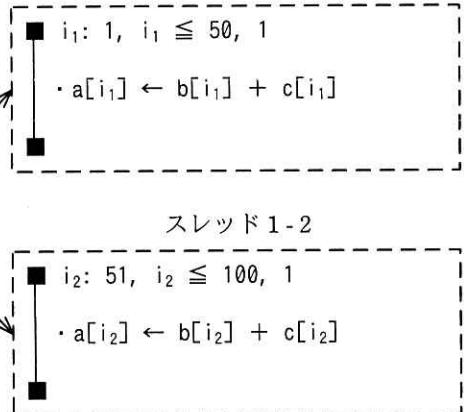
図 1 に示すプログラム 1 は、制御変数 i の取る範囲を分けることによって繰返し範囲を分割した繰返しの処理を、それぞれ異なるスレッドで実行できる。

なお、図 1、図 2 において、実線の四角はプログラム、破線の四角は繰返しの処理、破線の四角から出る二つの矢印は分割を示す。

プログラム 1



スレッド 1-1



スレッド 1-2

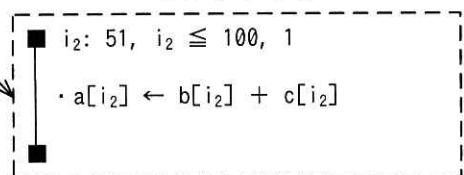


図 1 正しい結果が得られる場合の例

正しい結果が得られない場合の二つの例を、図 2 に示す。

プログラム 2 の繰返しの処理を、スレッド 2-1 とスレッド 2-2 の二つに分割すると、c ことがあるので、スレッド並列法を適用しない場合の実行結果と等しくなることを保証できない。したがって、プログラム 2 に対しては、繰返しの範囲の分割によるスレッド並列法を適用できない。

また、プログラム 3 の繰返しの処理を、スレッド 3-1 とスレッド 3-2 の二つに分割すると、d があるので、スレッド並列法を適用しない場合の実行結果と等しくなることを保証できない。したがって、プログラム 3 に対しても、繰返しの範囲の分割によるスレッド並列法を適用できない。

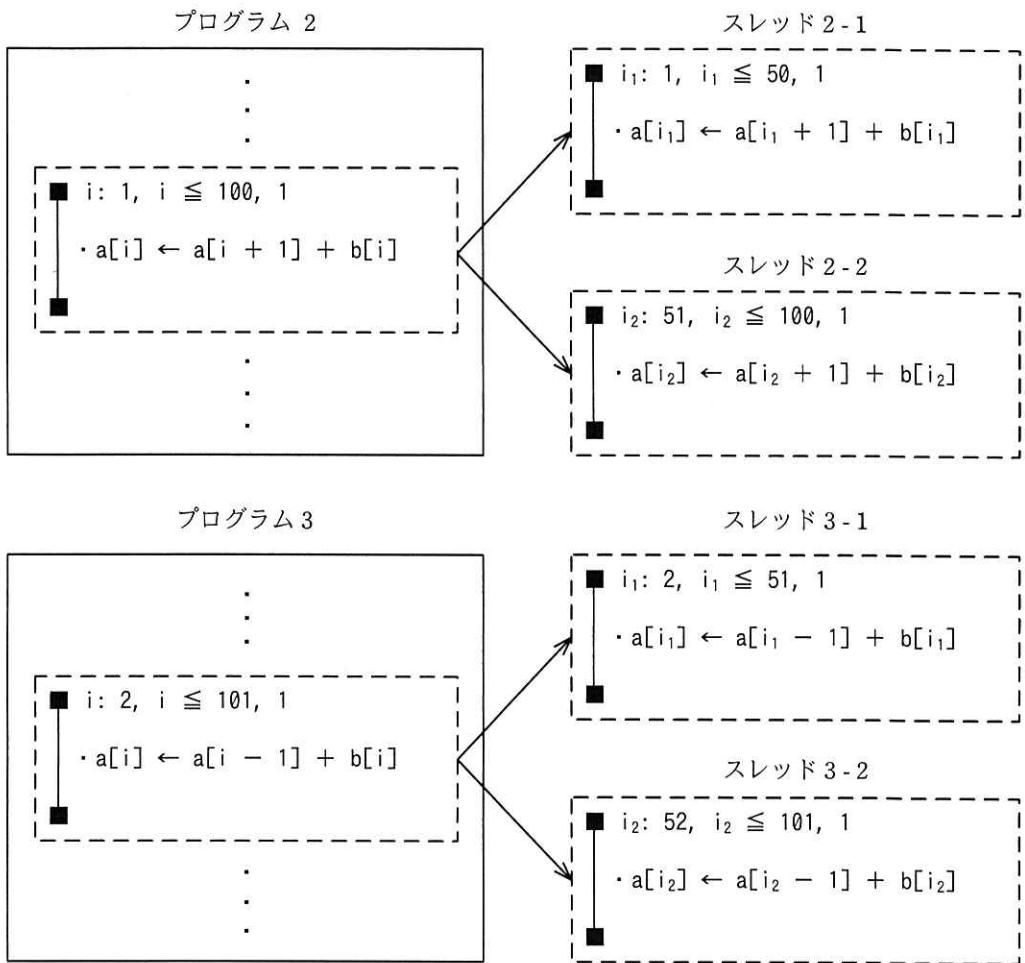


図 2 正しい結果が得られない場合の二つの例

cに関する解答群

- ア a[51] の値をスレッド 2-1 で更新するより先にスレッド 2-2 で更新する
- イ a[51] の値をスレッド 2-1 で更新するより先にスレッド 2-2 で参照する
- ウ a[51] の値をスレッド 2-1 で参照するより先にスレッド 2-2 で更新する
- エ a[51] の値をスレッド 2-1 で参照するより先にスレッド 2-2 で参照する

dに関する解答群

- ア a[51] の値をスレッド 3-1 で更新するより先にスレッド 3-2 で更新する
- イ a[51] の値をスレッド 3-1 で更新するより先にスレッド 3-2 で参照する
- ウ a[51] の値をスレッド 3-1 で参照するより先にスレッド 3-2 で更新する
- エ a[51] の値をスレッド 3-1 で参照するより先にスレッド 3-2 で参照する

設問3 図3に示すプログラム4では、配列aにおける更新対象の位置を配列ipの要素の値で指している。このプログラムでは、配列ipの要素の値によって、スレッド並列法を適用できる場合とできない場合がある。

図4に示す配列ipであれば、スレッド並列法を適用できる。図4中の
[]に入れる正しい答えを、解答群の中から選べ。

なお、図3において、実線の四角はプログラム、破線の四角は繰返しの処理、破線の四角から出る二つの矢印は分割を示す。

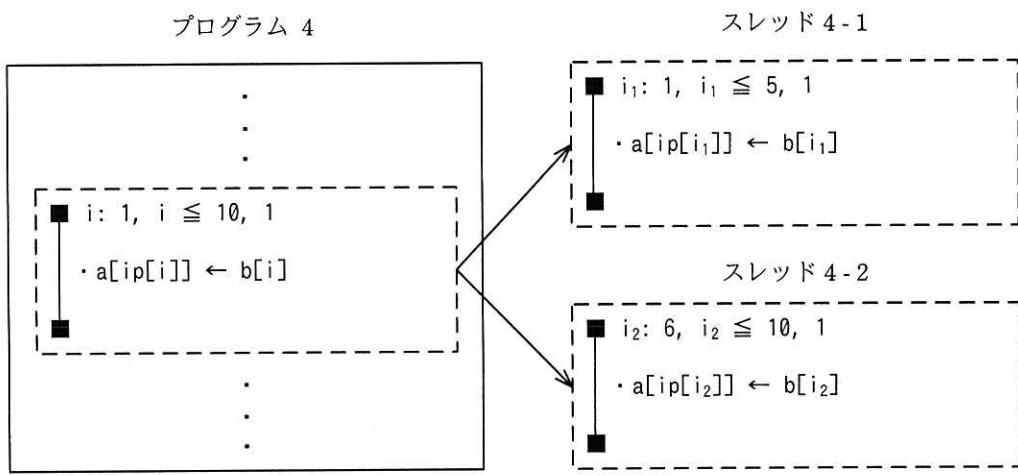


図3 プログラム4へのスレッド並列法の適用

要素番号	1	2	3	4	5	6	7	8	9	10
ip[]	1	2	3	4	5	e				

図4 スレッド並列法を適用できる配列ip

eに関する解答群

ア	1	2	8	9	10
---	---	---	---	---	----

ウ	6	7	4	9	10
---	---	---	---	---	----

イ	5	6	7	8	9
---	---	---	---	---	---

エ	6	7	8	9	10
---	---	---	---	---	----

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 書籍及び貸出情報を管理する関係データベースの設計及び運用に関する次の記述を読んで、設問1～3に答えよ。

D社の部署である資料室は、業務に関連する書籍を所蔵しており、従業員への貸出しを2015年4月から実施している。

所蔵する書籍を管理するデータベースは、書籍の情報を管理する書籍情報表と貸出状況を管理する貸出表とで構成されている。データベース構成を、図1に示す。下線付きの項目は主キーを表し、下破線付きの項目は外部キーを表す。各書籍は1冊しか所蔵していない。

書籍情報表（ISBNコード、書籍名、著者名、出版社名、出版年）
貸出表（貸出番号、ISBNコード、従業員番号、貸出日、返却予定日、返却日）

図1 データベース構成

[貸出表に関する説明]

- (1) 従業員に書籍を貸し出す際は、一意の貸出番号、貸し出す書籍のISBNコード、従業員番号、貸出日及び返却予定日を設定し、返却日にはNULLを設定したレコードを追加する。
- (2) 書籍が返却されたら、対象のレコードの返却日に返却された日付を設定する。

設問 1 次の SQL 文は、ISBN コードが ISBN978-4-905318-63-7 の書籍の貸出し状態を表示する SQL 文である。ISBN コードで貸出表を検索し、最も新しい貸出日のレコードの返却日に NULL が設定されている場合は、“貸出中”が表示される。また、最も新しい貸出日のレコードの返却日に日付が設定されている場合、及び貸出実績のない書籍の場合は、“貸出可”が表示される。[] に入れる正しい答えを、解答群の中から選べ。ここで、検索に使用する ISBN コードの書籍は必ず所蔵されているものとする。また、返却された書籍はその日のうちに再び貸し出されることはない。

```
SELECT 貸出表.ISBNコード,
CASE WHEN [ ] a
END AS 書籍状態
FROM 貸出表
WHERE 貸出表.ISBNコード = 'ISBN978-4-905318-63-7'
AND 貸出表.貸出日 = (SELECT [ ] b FROM 貸出表
WHERE 貸出表.ISBNコード = 'ISBN978-4-905318-63-7')
UNION ALL
SELECT DISTINCT 書籍情報表.ISBNコード, '貸出可' AS 書籍状態
FROM 書籍情報表
WHERE 書籍情報表.ISBNコード = 'ISBN978-4-905318-63-7'
AND NOT EXISTS (SELECT 貸出表.ISBNコード FROM 貸出表
WHERE 貸出表.ISBNコード = 'ISBN978-4-905318-63-7' )
```

a に関する解答群

ア 貸出表.返却日 IS NOT NULL THEN '貸出中' ELSE '貸出可'

イ 貸出表.返却日 IS NOT NULL THEN '貸出中'
WHEN 貸出表.返却日 IS NULL THEN '貸出可'

ウ 貸出表.返却日 IS NULL THEN '貸出可' ELSE '貸出中'

エ 貸出表.返却日 IS NULL THEN '貸出中'
WHEN 貸出表.返却日 IS NOT NULL THEN '貸出可'

b に関する解答群

ア DISTINCT 貸出表.貸出日

イ MAX(貸出表.貸出日)

ウ MIN(貸出表.貸出日)

エ 貸出表.貸出日

設問2 2018年4月1日から2019年3月31までの間に4回以上貸し出した書籍の一覧を取得することにした。次のSQL文の [] に入れる正しい答えを、解答群の中から選べ。

```
SELECT 書籍情報表.ISBNコード, 書籍情報表.書籍名, COUNT(*) AS 貸出回数
FROM 書籍情報表, 貸出表
WHERE 書籍情報表.ISBNコード = 貸出表.ISBNコード
```

[]
c

cに関する解答群

- ア AND (貸出表.貸出日 >= '2018-04-01' OR 貸出表.貸出日 <= '2019-03-31')
GROUP BY 書籍情報表.ISBNコード, 書籍情報表.書籍名
HAVING COUNT(*) >= 4
- イ AND 貸出表.貸出日 BETWEEN '2018-04-01' AND '2019-03-31'
GROUP BY 書籍情報表.ISBNコード, 書籍情報表.書籍名
HAVING COUNT(*) >= 4
- ウ AND 貸出表.貸出日 >= '2018-04-01' AND 貸出表.貸出日 <= '2019-03-31'
AND COUNT(*) >= 4
- エ GROUP BY 書籍情報表.ISBNコード, 書籍情報表.書籍名, 貸出表.貸出日
HAVING 貸出表.貸出日 >= '2018-04-01' AND 貸出表.貸出日 <= '2019-03-31'
AND COUNT(*) >= 4

設問 3 従業員と資料室担当者の利便性を向上させる目的で、所蔵する書籍を管理するデータベースを再構築することにした。

データベースの再構築に当たり、従業員と資料室担当者から要望が出された。
次の記述中の [] に入る適切な答えを、解答群の中から選べ。

[従業員と資料室担当者からの要望]

要望 1 ISBN コードが同じ書籍を複数冊所蔵できるようにしたい。

要望 2 書籍の購入日を管理できるようにしたい。

要望 3 ISBNコードごとに所蔵する書籍数及び貸出し中の書籍数（以下、貸出中件数という）が分かるようにしたい。

要望 4 ISBNコードが同じ書籍は同じラックに保管して、書籍が収納されているラックが分かるようにしたい。

従業員と資料室担当者からの要望を反映したデータベース構成案を、図 2 に示す。下線付きの項目は主キーを表し、下破線付きの項目は外部キーを表す。

書籍情報表 (ISBNコード, 書籍名, 著者名, 出版社名, 出版年, ラック番号)

貸出表 (貸出番号, 書籍番号, 従業員番号, 貸出日, 返却予定日, 返却日)

書籍表 (書籍番号, ISBNコード, 購入日)

書籍管理ビュー (ISBNコード, 所蔵書籍数, 貸出中件数)

ラック表 (ラック番号, ラック名)

図 2 要望を反映したデータベース構成案

[要望に対するデータベース修正内容]

修正 1 要望 1 に対応するために書籍表を追加して、資料室で所蔵している各書籍に一意の書籍番号を割り振って、それを主キーとした。また、貸出表の ISBN コードを書籍番号に変更した。

修正 2 要望 2 に対応するために書籍表に購入日を設けた。

修正 3 要望 3 に対応するために書籍管理ビューを追加した。

修正 4 要望 4 に対応するためにラック表を追加して、書籍情報表に外部キーとしてラック番号を追加した。

要望を反映したデータベース構成案では、既に所蔵している書籍と ISBN コードが同じ書籍を追加購入した場合に、レコードを追加する必要のある表は **d** である。

また、需要がなくなった書籍を廃棄する場合は、ISBN コードが同じ書籍を全て廃棄する。データベースに対して行う操作は、次の①～④を、**e** の順序で行う必要がある。

- ① 書籍情報表の主キーが対象 ISBN コードのレコードを削除する。
- ② 書籍表から対象 ISBN コードに対応する書籍番号を抽出する。
- ③ 書籍表の対象 ISBN コードに対応するレコードを削除する。
- ④ 貸出表の対象書籍番号に対応するレコードを削除する。

d に関する解答群

- | | |
|--------------|-------------------|
| ア 書籍表 | イ 書籍表及びラック表 |
| ウ 書籍情報表及び書籍表 | エ 書籍情報表、書籍表及びラック表 |

e に関する解答群

- | | | |
|-----------------|-----------------|-----------------|
| ア ② → ① → ③ → ④ | イ ② → ① → ④ → ③ | ウ ② → ③ → ① → ④ |
| エ ② → ③ → ④ → ① | オ ② → ④ → ① → ③ | カ ② → ④ → ③ → 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 NATに関する次の記述を読んで、設問1、2に答えよ。

IPv4のIPアドレスのうち、全世界で重複しないように管理されているグローバルIPアドレスはインターネットへの接続に利用でき、プライベートIPアドレスは社内LANなどの閉じたネットワークだけで利用できる。

プライベートIPアドレスだけが割り当てられている機器（以下、LAN内機器という）とインターネットに接続されている外部の機器（以下、インターネット機器という）とは直接通信することはできないが、例えば、NAT（Network Address Translation）を使うことによって通信することができるようになる。

本問で扱うNATは、NAPT（Network Address Port Translation）とも呼ばれる、ルータが搭載している機能であり、通過するパケットのIPアドレス及びポート番号を書き換えることによって、LAN内機器とインターネット機器との通信を可能にする。表1に、LAN内機器とインターネット機器との通信の際にルータを通過するパケットの、IPアドレス及びポート番号の書換えの概要を示す。ここで、送信パケットとはLAN内機器がインターネット機器に向けて送信するパケットのことをいい、受信パケットとはルータがインターネット機器から受信するパケットのことをいう。

表1 IPアドレス及びポート番号の書換えの概要

	書換え対象	書換え前	書換え後
送信 パケット	送信元IPアドレス	LAN内機器のIPアドレス	ルータのグローバルIPアドレス
	送信元ポート番号	LAN内機器のポート番号	ルータのポート番号
受信 パケット	宛先IPアドレス	ルータのグローバルIPアドレス	LAN内機器のIPアドレス
	宛先ポート番号	ルータのポート番号	LAN内機器のポート番号

NATには、静的NATと動的NATがある。

静的NATでは、ルータのグローバルIPアドレス及びルータのポート番号の組みとLAN内機器のIPアドレス及びLAN内機器のポート番号の組みとの対応をあらかじめ定義しておき、その定義に基づいて、送信パケットと受信パケットの書換え対象の

IP アドレス及びポート番号を書き換える。

動的 NAT では、送信パケットと受信パケットの書換え対象の IP アドレス及びポート番号を、次のように書き換える。

(1) 送信パケットの送信元 IP アドレス及び送信元ポート番号の書換え

- ① 送信パケットの送信元 IP アドレス及び送信元ポート番号の、書換え前の組み（LAN 内機器の IP アドレス及び LAN 内機器のポート番号の組み）と書換え後の組み（ルータのグローバル IP アドレス及びルータのポート番号の組み）とを、関連付けて一定期間記憶する。
- ② 送信パケットの送信元 IP アドレス及び送信元ポート番号の組みを、書換え前の組みとして記憶している間は、関連付けられている書換え後の組みに書き換える。
- ③ 送信パケットの送信元 IP アドレス及び送信元ポート番号の組みを、書換え前の組みとして記憶していないときは、ルータに割り当てられている幾つかのグローバル IP アドレスのうちの一つと、その IP アドレスで使用されていないポート番号のうちの一つとの組みに書き換える。

(2) 受信パケットの宛先 IP アドレス及び宛先ポート番号の書換え

- ① 受信パケットの宛先 IP アドレスと宛先ポート番号の組みが、上記(1)① の書換え後の組みとして記憶されている間は、関連付けられている書換え前の組みに書き換える。

設問 1 次の(1)～(3)のケースのうち、静的 NAT よりも動的 NAT の方が適しているものを、解答群の中から選べ。

- (1) インターネット機器からアクセス可能なサーバを、LAN 内機器として設置する。
- (2) LAN 内機器から、インターネット機器にアクセスする。
- (3) インターネットを介する異なる LAN の LAN 内機器同士が、あらかじめ決まった固定のポートを使い、相互に通信する。

解答群

ア (1)だけ

イ (1)と(2)

ウ (1)と(3)

エ (2)だけ

オ (2)と(3)

カ (3)だけ

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。ここで、a1～a3 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。

IPv6 と IPv4 とは互換性がないので、IPv6 のネットワーク内の機器（以下、IPv6 機器という）と IPv4 のネットワーク内の機器（以下、IPv4 機器という）とは直接通信することができない。IPv6 機器から IPv4 機器にアクセスする方法の一つに、NAT の機能を拡張した NAT64 と、DNS の機能を拡張した DNS64 との組合せによる方法がある。この方法による IPv6 機器から IPv4 機器へのアクセスの流れを次に示す。

(1) IPv6 機器は、アクセス先の機器の IP アドレスを、DNS64 から入手する。

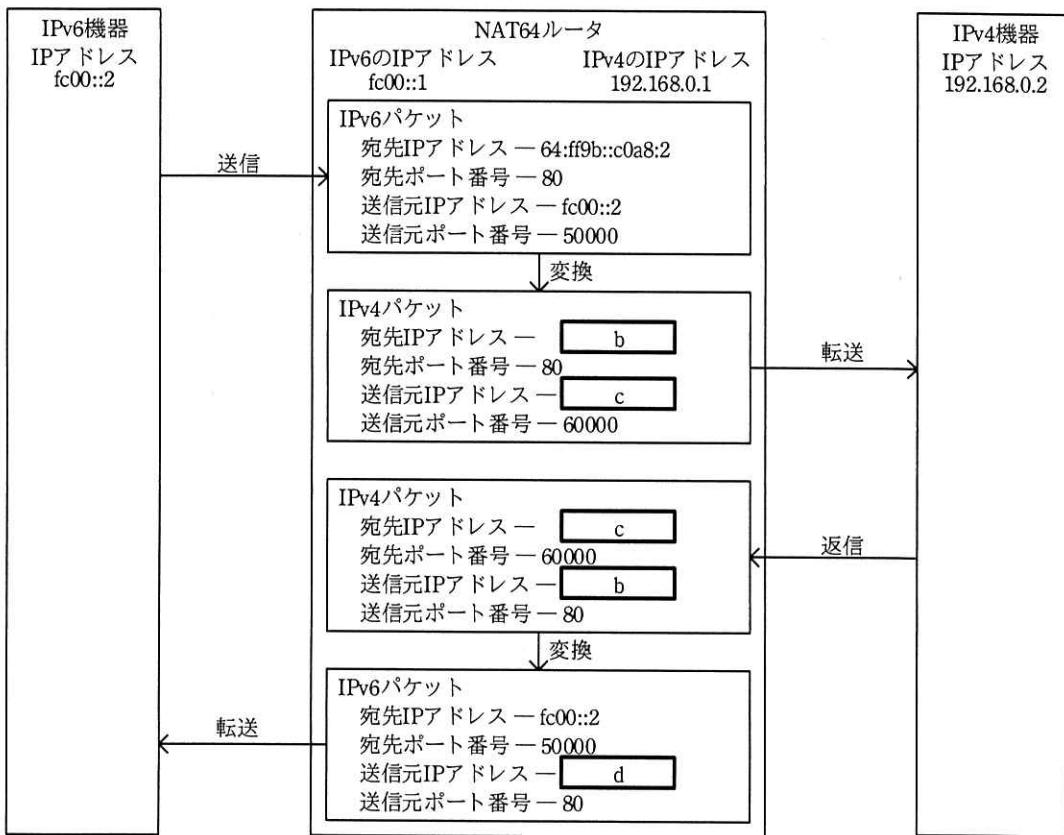
DNS64 は [a1] のネットワークに置かれる DNS であり、ホスト名に対応する IP アドレスの問合せに対し、対応する [a2] アドレスがあればそれを返し、対応する [a2] アドレスがなく、[a3] アドレスがあればそれを [a2] アドレスに変換して返す。ここで、IPv4 アドレスの IPv6 アドレス表現は、当該 IPv4 アドレスを示す 4 バイトの前に、あらかじめ決められた 12 バイトのプレフィックスを付加したものである。

(2) IPv6 機器は、入手した IP アドレスに宛てて IPv6 のパケットを送信する。

(3) (2)のパケットが IPv4 機器向けならば、当該パケットとその返信パケットは、NAT64 の機能をもつルータ（以下、NAT64 ルータという）が受信する。

(4) NAT64 ルータは、IPv6 機器から IPv4 機器に向けて送信された IPv6 のパケットを IPv4 のパケットに、その返信パケットである IPv4 のパケットを IPv6 のパケットに、それぞれ変換し、転送する。このとき、IP アドレス及びポート番号は、動的 NAT による書換えの考え方を用いて変換する。

NAT64 ルータによる IP アドレスとポート番号の変換例を、図 1 に示す。



注記 プレフィックスは、64:ff9b:0:0:0:0とする。

図1 NAT64 ルータによる IP アドレスとポート番号の変換例

aに関する解答群

	a1	a2	a3
ア	IPv4	IPv4	IPv6
イ	IPv4	IPv6	IPv4
ウ	IPv6	IPv4	IPv6
エ	IPv6	IPv6	IPv4

b~dに関する解答群

ア 192.168.0.0

イ 192.168.0.1

ウ 192.168.0.2

エ 64:ff9b::

オ 64:ff9b::c0a8:1

カ 64:ff9b::c0a8:2

キ fc00::

ク fc00::1

ケ fc00::2

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 ストレスチェックの検査支援を行うシステムに関する次の記述を読んで、設問1, 2に答えよ。

K社は、厚生労働省が作成した“労働安全衛生法に基づくストレスチェック制度実施マニュアル（平成28年4月改訂）”を基に、労働者の、職業上の心理的な負担の程度を把握するための検査を支援するシステムを開発している。このシステムは、“職業性ストレス簡易調査票”の質問の全てに回答が入力されると、質問項目の領域ごとに回答の合計点を求めて、高ストレス者を簡易的に判別する。

〔職業性ストレス簡易調査票の説明〕

(1) 職業性ストレス簡易調査票には、全部で57項目の質問があり、次の4領域に分類される。

領域A 職場における当該労働者の心理的な負担の原因に関する質問（17項目）

領域B 心理的な負担による心身の自覚症状に関する質問（29項目）

領域C 職場における他の労働者による当該労働者への支援に関する質問（9項目）

領域D 仕事及び家庭生活の満足度に関する質問（2項目）

(2) 各質問に対して、四つの選択肢から一つを選択して回答する。各選択肢には、あらかじめ点数（1, 2, 3, 4点のいずれか）が割り振られている。領域Aの一部を例に、質問、選択肢、回答例及び回答例での点数を、表1に示す。

〔高ストレス者を判別する方法〕

(1) 領域ごとに、質問に対する回答の合計点を求める。

(2) 次のいずれかを満たす場合に、高ストレス者と判別する。領域Dの合計点は、高ストレス者の判別には利用しない。

- ① 領域 B の合計点が 77 点以上である。
- ② 領域 B の合計点が 63 点以上 76 点以下であつて、かつ、領域 A 及び C の合計点の和が 76 点以上である。

合計点によって高ストレス者と判別する〔高ストレス者を判別する方法〕の(2)の①及び②の範囲を、図 1 に示す。図 1 の網掛けの範囲に入る場合は高ストレス者であるとし、それ以外の場合は高ストレス者ではないとする。

表 1 質問、選択肢、回答例及び回答例での点数（領域 A の一部）

領域 A の質問	選択肢				点回答例での 数
	そう だ	そ う だ	ち や が や う	ち が う	
1. 非常にたくさんの仕事をしなければならない	○				4
2. 時間に仕事が処理しきれない	○				4
3. 一生懸命働かなければならない		○			3
4. かなり注意を集中する必要がある			○		2
5. 高度の知識や技術が必要なむずかしい仕事だ			○		2
6. 勤務時間中はいつも仕事のことを考えていなければならない	○				4
7. からだを大変よく使う仕事だ				○	1
8. 自分のペースで仕事ができる				○	4
9. 自分で仕事の順番・やり方を決めることができる			○		3

注記 “○” は、選択した回答を示す。

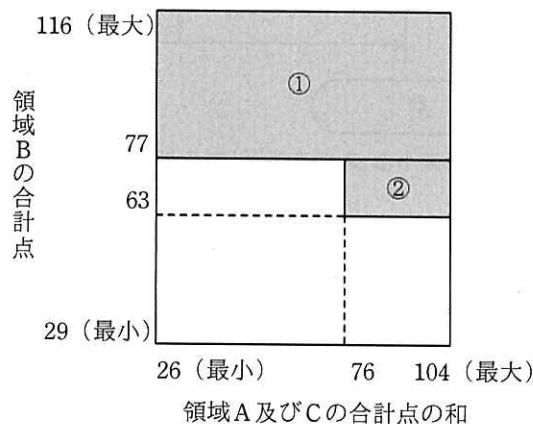


図 1 高ストレス者と判別する範囲（網掛け）

設問1 図2中の [] に入る正しい答えを、解答群の中から選べ。

職業性ストレス簡易調査票の回答結果から高ストレス者を判別する処理の流れ図を、図2に示す。変数“判別結果”に初期値として0を格納しておき、高ストレス者と判別した場合は、“判別結果”に1を格納する。

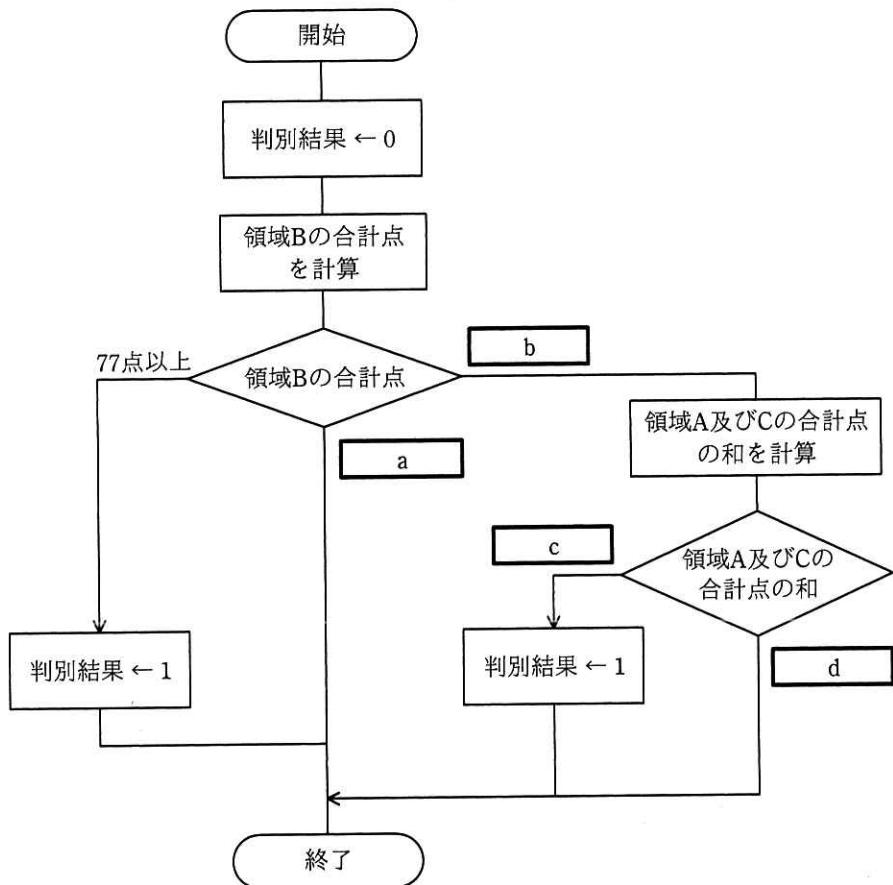


図2 高ストレス者を判別する処理の流れ図

a ~ dに関する解答群

ア 62点以下

イ 62点以上

ウ 63点以下

エ 63点以上

オ 63点以上かつ76点以下

カ 63点以上かつ77点以下

キ 75点以下

ク 76点以上

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。

このシステムのテストに備えてテストデータを用意した。各テストデータは、領域 A～C の回答の合計点が表 2 に示す合計点になるように回答が入力された職業性ストレス簡易調査票である。

図 1に基づいて、①、②及びそれ以外の場合を判別できるかどうかをテストするには、テストデータ [e] を使用する。また、図 2 の流れ図で、分岐による全てのパスを通るテストをするには、テストデータ [f] を使用する。ここで、どちらのテストも、使用するテストデータの件数が最少となるように実施する。

表 2 用意したテストデータの各領域の合計点

テストデータ	領域 A の合計点	領域 B の合計点	領域 C の合計点
1	34	63	18
2	34	87	27
3	34	63	36
4	51	87	36
5	51	58	36
6	51	66	36

e, fに関する解答群

- | | |
|----------------|----------------|
| ア 1, 2 及び 3 | イ 1, 2 及び 4 |
| ウ 1, 3 及び 6 | エ 2, 4 及び 6 |
| オ 4, 5 及び 6 | カ 1, 2, 3 及び 4 |
| キ 1, 3, 4 及び 5 | ク 1, 4, 5 及び 6 |
| ケ 2, 3, 4 及び 5 | コ 2, 4, 5 及び 6 |

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 販売管理システム開発の結合テストにおける進捗及び品質管理に関する次の記述を読んで、設問1～3に答えよ。

製造業のP社では、販売管理システムを構築するプロジェクト（以下、Qプロジェクトという）を進めており、情報システム部門のRさんがプロジェクトマネージャを担当している。P社では、結合テスト工程において、バグ管理図を用いて、テストの進捗とソフトウェアの品質を評価している。本問におけるバグ管理図とは、横軸に結合テスト期間の経過率を、縦軸に未消化テスト項目数及び累積バグ検出数を表したグラフのことである。

P社では、過去のシステム構築の実績値を基に、テスト項目数及びバグ検出数の標準値を定めており、Qプロジェクトの結合テストで用いる、テスト項目1件当たりのバグ検出数の標準値は、0.02件である。Qプロジェクトにおける結合テスト期間の経過率ごとの未消化テスト項目数及び累積バグ検出数の計画値を、表1に示す。

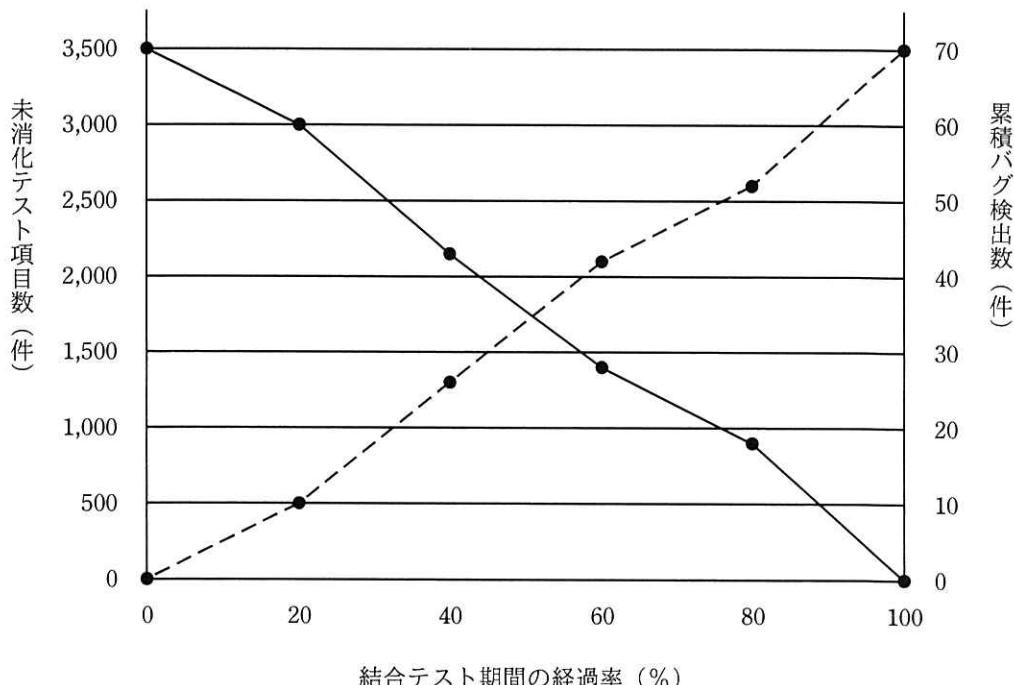
Qプロジェクトでは、未消化テスト項目数、消化済テスト項目数及び累積バグ検出数の計画値と実績値から進捗と品質を評価する。また、結合テスト工程では、累積バグ検出数の実績値が、消化済テスト項目数の実績値に基づいて算出した累積バグ検出数の計画値の±25%の範囲内の場合、品質に問題ないと判断する。

表1 結合テスト期間の経過率ごとの未消化テスト項目数及び累積バグ検出数の計画値

結合テスト期間の経過率 (%)	0	20	40	60	80	100
未消化テスト項目数 (件)	3,500	3,000	2,200	1,400	900	0
累積バグ検出数 (件)	0	10	26	42	52	70

表1を基にしたバグ管理図を、図1に示す。Rさんは、図1に示すバグ管理図に、結合テスト期間の60%が経過した時点（以下、60%経過時点という）の未消化テスト項目数及び累積バグ検出数の実績値をプロットして進捗と品質を評価することにした。結合テストの担当者は、検出したバグの原因調査と修正も行う。結合テストの担当者A～Eそれぞれのテスト項目数の計画値と60%経過時点での消化済テスト項目

数及び累積バグ検出数の実績値を、表 2 に示す。60% 経過時点での結合テスト全体の未消化テスト項目数の実績値は図 1 の [a]、累積バグ検出数の実績値は図 1 の [b]。R さんはプロットした結果を基に、結合テストは計画どおりには進捗していないと判断した。また、担当者 A～E の 60% 経過時点での累積バグ検出数の実績値の合計値は、担当者 A～E の 60% 経過時点での消化済テスト項目数の実績値の合計値に、バグ検出数の標準値である 0.02 を乗じて算出した累積バグ検出数の [c] と判断した。



注記 実線の折れ線は未消化テスト項目数の計画値の推移を、破線の折れ線は累積バグ検出数の計画値の推移を表す。

図 1 表 1 を基にしたバグ管理図

表 2 担当者 A～E が担当するテスト項目数の計画値と 60% 経過時点での実績値

担当者	単位 件				
	A	B	C	D	E
担当するテスト項目数（計画値）	500	700	700	800	800
60% 経過時点での消化済テスト項目数（実績値）	210	390	400	450	300
60% 経過時点での累積バグ検出数（実績値）	4	9	8	11	13

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 実線の折れ線が示す未消化テスト項目数の値より大きく
- イ 実線の折れ線が示す未消化テスト項目数の値と等しく
- ウ 実線の折れ線が示す未消化テスト項目数の値より小さく

b に関する解答群

- ア 破線の折れ線が示す累積バグ検出数の値より大きい
- イ 破線の折れ線が示す累積バグ検出数の値と等しい
- ウ 破線の折れ線が示す累積バグ検出数の値より小さい

c に関する解答群

- ア 計画値の 75% 未満なので、品質に問題がある
- イ 計画値の 75% 以上 100% 未満なので、品質に問題はない
- ウ 計画値の 100% 以上 125% 以下なので、品質に問題はない
- エ 計画値の 125% を超えているので、品質に問題がある

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。

R さんは、更に結合テストの担当者ごとの進捗を評価することにした。60% 経過時点での担当者ごとの消化済テスト項目数の計画値は、次の式で求める。

$$\frac{\text{担当するテスト項目数} \times \frac{60\% \text{ 経過時点での消化済テスト項目数の計画値}}{\text{結合テスト全体のテスト項目数の計画値}}$$

60% 経過時点での消化済テスト項目数の計画値は、表 1 で示す未消化テスト項目数の計画値に基づいて算出した値である。60% 経過時点での担当者ごとの消化済テスト項目数の計画値を、表 3 に示す。

表3 60%経過時点での担当者ごとの消化済テスト項目数の計画値

担当者	単位 件				
	A	B	C	D	E
消化済テスト項目数	300	d		480	480

注記 網掛けの部分は表示していない。

Q プロジェクトでは、結合テスト工程において、消化済テスト項目数の実績値が計画値の ±10% の範囲内の場合、進捗に問題はないと判断する。B さん、C さん、D さんの消化済テスト項目数の実績値は計画値の ±10% の範囲内であり、累積バグ検出数の実績値も 60% 経過時点での消化済テスト項目数の実績値に基づいて算出した累積バグ検出数の計画値の ±25% の範囲内なので、進捗及び品質に問題はないと判断した。

A さんが担当するテストの進捗とソフトウェアの品質に基づく判断は、次のとおりである。

- ・進捗：消化済テスト項目数の実績値が計画値の 90% 未満なので、進捗は遅れている。
- ・品質：バグの検出及び検出したバグの原因調査と修正は、順調に行われている。60% 経過時点での消化済テスト項目数の実績値に基づいて算出した累積バグ検出数の計画値は 4.2 件であり、累積バグ検出数の実績値は計画値の ±25% の範囲内なので、品質に問題はない。

R さんは、A さんが担当するテストの進捗が遅れているので、A さんの作業に問題がないかどうかを確認した。テスト項目の内容及びテスト手順は正しく、報告書も適切に記載されていたが、結合テストデータの作成に時間を要していることが分かった。そこで、R さんは A さんの進捗遅れに対して、
e という対応を実施することにした。

d に関する解答群

ア 280

イ 300

ウ 360

エ 420

e に関する解答群

- ア テスト項目を再度洗い出す
- イ テスト要員を追加する
- ウ テストデータを再作成する
- エ テスト証跡の記載を一部省略する
- オ テストの結果を A さんの結合テスト完了後に確認する

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。

R さんは、60% 経過時点での E さんの消化済テスト項目数の実績値が計画値の 90% 未満であり、累積バグ検出数の実績値が消化済テスト項目数の実績値に基づいて算出した累積バグ検出数の計画値よりも大きくなっていたので、原因を調査することにした。E さんは、E さん以外の担当者が単体テストまでを行った機能 1～5 の結合テストを担当している。各機能は独立してテストが可能であり、機能 1 から順番にテストを行う計画である。

調査の結果、E さんは機能 1 のテストは順調に完了したが、機能 2 のテストがはかどっていないことが分かった。理由を確認すると、機能 2 はバグの検出数が多く、バグの原因調査に時間を要したからであった。そこで、これまでに機能 2 で検出されたバグの原因を調査した結果、“詳細設計書の論理誤り”が多く見受けられた。R さんは、更に、機能 2 の詳細設計を担当した者（以下、機能 2 担当者という）が詳細設計を担当した他の機能について、その結合テストの進捗を確認したところ、いずれの機能も結合テストの開始前であった。いずれの機能も、機能 2 と同じように問題が発生するおそれがあるので、R さんは、販売管理システムに精通した要員を追加して、[f] を実施することにした。

f に関する解答群

- ア E さんがテストを担当した機能の詳細設計書の再レビュー
- イ 機能 2 担当者が担当した機能の詳細設計書の再レビュー
- ウ 全機能の詳細設計書の再レビュー
- エ 販売管理システムの要件を理解するための勉強会

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 製品別の収益分析に関する次の記述を読んで、設問1～3に答えよ。

S社は、製品X、製品Y、製品Zを販売している。S社では、収益改善を目的にして、製品別の営業利益と営業利益率に関する分析を行っている。製品別の前年度実績を、表1に示す。

なお、本問における営業利益率などのパーセント(%)表記の値は、表においては、小数第1位を四捨五入して、整数で表示している。他の文中のパーセント(%)表記の値は、そのままの値を示している。

表1 製品別の前年度実績

	製品X	製品Y	製品Z	全体
売上高 (百万円)	2,200	1,000	800	4,000
営業費用 (百万円)	2,100	850	680	3,630
営業利益 (百万円)	100	150	120	370
営業利益率 (%)	5	15	15	9

注記 営業費用は、売上原価と販売費及び一般管理費で構成される。

設問1 営業利益率の改善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。

製品Xは、S社の売上高の半分以上を占めているが、営業利益率は全製品の中で最も低くなっている。そこでS社は、製品Xの営業利益率を上げるために施策を検討することにした。

製品Xの営業利益率を、ほかの製品の前年度実績を上回る16%にするためには、営業費用が前年度と同額ならば、売上高を [a] 百万円増やす必要がある。売上高が前年度と同額ならば、営業費用を [b] 百万円減らす必要がある。

aに関する解答群

- | | |
|-------|-------|
| ア 242 | イ 300 |
| ウ 352 | エ 400 |

bに関する解答群

- | | |
|-------|-------|
| ア 231 | イ 252 |
| ウ 336 | エ 352 |

設問2 収益改善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 と c2 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。

S社は、各製品の収益を分析するために、製品別の営業費用を調査し、営業費用を固定費と変動費に分けた。調査結果を、表2に示す。ここで、固定費は販売数量の増減にかかわらず発生する一定額の費用のことであり、変動費は販売数量に比例して変化する費用のことである。

表2 製品別の固定費と変動費

単位 百万円			
	製品 X	製品 Y	製品 Z
固定費	1,000	350	280
変動費	1,100	500	400

次にS社は、各製品の安全余裕率の分析を行った。安全余裕率は、売上高と損益分岐点売上高との差から算出される指標であり、数値が大きいほど売上高が低下した場合に赤字になる可能性が低いといった余裕度を示す。安全余裕率を求める式は、次のとおりである。安全余裕率に関わる項目の値を表3に、S社が定めている安全余裕率の基準とその状態を表4に示す。

限界利益率 = (売上高 - 変動費) ÷ 売上高

損益分岐点売上高 = 固定費 ÷ 限界利益率

安全余裕率 = (売上高 - 損益分岐点売上高) ÷ 売上高

表3 安全余裕率に関する項目の値

	製品 X	製品 Y	製品 Z
売上高 (百万円)	2,200	1,000	800
限界利益率 (%)	50	50	50
損益分岐点売上高 (百万円)	2,000	700	560
安全余裕率 (%)	9	30	30

表4 安全余裕率の基準とその状態

安全余裕率 (%)	状態
10未満	危険
10以上20未満	普通
20以上40未満	優良
40以上	極めて優良

表3の安全余裕率を見ると、製品Xは危険な状態にある。S社は、固定費を削減することによって、前年度実績と同じ売上高で安全余裕率20%を達成できるように、製品Xの固定費の削減目標の値を [c1] 百万円と設定した。S社は、この目標値を達成するために、製品Xだけを販売している営業所を統廃合して賃借料などの固定費を削減することとした。ここで、営業所の統廃合によって製品Xの売上高は変化しないものとし、統廃合時に一時的に発生する費用は考慮しない。

さらに、統廃合の結果として削減される製品Xの固定費の削減金額 [c1] 百万円を製品Zの固定費である人件費に追加して、営業を強化することにした。これによって、製品Zの固定費は [c1] 百万円増えるが、売上高は1,000百万円に増やせると見込んだ。ここで、製品1個当たりの販売価格は販売数量にかかわらず同じとする。安全余裕率に関する項目の試算値を、表5に示す。

表 5 安全余裕率に関する項目の試算値

	製品 X	製品 Z
売上高 (百万円)	2,200	1,000
限界利益率 (%)		
損益分岐点売上高 (百万円)		
安全余裕率 (%)	20	c2

注記 網掛けの部分は表示していない。

cに関する解答群

	c1	c2
ア	120	20
イ	120	33
ウ	240	20
エ	240	33

設問3 営業利益率の試算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。

S社では、製品Xの売上高を確保するために、販売時に本来の販売価格に対して一律12%の値引きを行っていた。値引きなしで同じ売上高を達成した場合に製品Xの営業利益率がどうなるか、前年度実績に基づいて試算した。試算結果を、表6に示す。ここで、値引きなしで売る場合においても、製品1個当たりの販売価格は販売数量にかかわらず同じとする。

表6 試算結果

		値引きあり	値引きなし
売上高	(百万円)	2,200	2,200
営業費用	変動費	1,100	d
	固定費	1,000	
営業利益		100	
営業利益率(%)		5	e

注記 網掛けの部分は表示していない。

dに関する解答群

- | | |
|---------|---------|
| ア 880 | イ 968 |
| ウ 1,100 | エ 1,232 |
| オ 1,250 | |

eに関する解答群

- | | |
|------|------|
| ア 5 | イ 10 |
| ウ 11 | エ 15 |

次の問8は必須問題です。必ず解答してください。

問8 次のプログラムの説明及びプログラムを読んで、設問1～3に答えよ。

[プログラムの説明]

関数 BitapMatch は、Bitap 法を使って文字列検索を行うプログラムである。

Bitap 法は、検索対象の文字列（以下、対象文字列という）と検索文字列の照合に、個別の文字ごとに定義されるビット列を用いるという特徴をもつ。

なお、本問では、例えば2進数の16ビット論理型の定数 0000000000010101 は、上位の0を省略して“10101”Bと表記する。

(1) 関数 BitapMatch は、対象文字列を Text[] に、検索文字列を Pat[] に格納して呼び出す。配列の要素番号は1から始まり、Text[]の i 番目の文字は Text[i] と表記する。Pat[]についても同様に i 番目の文字は Pat[i] と表記する。対象文字列と検索文字列は、英大文字で構成され、いずれも最長 16 文字とする。

対象文字列 Text[] が “AACBBAACABABAB”，検索文字列 Pat[] が “ACABAB” の場合の格納例を、図1に示す。

要素番号	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Text[]	A	A	C	B	B	A	A	C	A	B	A	B	A	B

要素番号	1	2	3	4	5	6
Pat[]	A	C	A	B	A	B

図1 対象文字列と検索文字列の格納例

(2) 関数 BitapMatch は、関数 GenerateBitMask を呼び出す。

関数 GenerateBitMask は、文字 “A”～“Z”的文字ごとに、検索文字列に応じたビット列（以下、ビットマスクという）を生成し、要素数 26 の 16 ビット論理型配列 Mask[] に格納する。Mask[1] には文字 “A” に対するビットマスクを、Mask[2]

には文字 “B” に対するビットマスクを格納する。このように Mask[1] ~ Mask[26] に文字 “A” ~ “Z” に対応するビットマスクを格納する。

関数 GenerateBitMask は、Mask[] の全ての要素を “0”B に初期化した後、1 以上で Pat[] の文字数以下の全ての i に対して、Pat[i] の文字に対応する Mask[] の要素である Mask[Index(Pat[i])] に格納されている値の、下位から数えて i 番目のビットの値を 1 にする。

関数 Index は、引数にアルファベット順で n 番目の英大文字を設定して呼び出すと、整数 n ($1 \leq n \leq 26$) を返す。

(3) 図 1 で示した、Pat[] が “ACABAB” の例の場合、関数 GenerateBitMask を実行すると、Mask[] は図 2 のとおりになる。

Mask[1]	0000000000010101	“A”に対応するビットマスク
Mask[2]	a	“B”に対応するビットマスク
Mask[3]	0000000000000010	“C”に対応するビットマスク
Mask[4]	0000000000000000	“D”に対応するビットマスク
	:	
Mask[26]	0000000000000000	“Z”に対応するビットマスク

図 2 図 1 で示した Pat[] に対する Mask[] の値

(4) 関数 GenerateBitMask の引数と返却値の仕様は、表 1 のとおりである。

表 1 関数 GenerateBitMask の引数と返却値の仕様

引数／返却値	データ型	入力／出力	説明
Pat[]	文字型	入力	検索文字列が格納されている 1 次元配列
Mask[]	16 ビット論理型	出力	文字 “A” ~ “Z” に対応するビットマスクが格納される 1 次元配列
返却値	整数型	出力	検索文字列の文字数

[プログラム 1]

- 整数型関数: GenerateBitMask(文字型: Pat[], 16 ビット論理型: Mask[])
- 整数型: i, PatLen

```
· PatLen ← Pat[] の文字数
■ i: 1, i ≤ 26, 1
    · Mask[i] ← b /* 初期化 */
■ i: 1, i ≤ PatLen, 1
    · Mask[Index(Pat[i])] ← c と
        Mask[Index(Pat[i])] とのビットごとの論理和
■
    · return (PatLen)
```

設問 1 プログラムの説明及びプログラム 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 0000000000000101 | イ 0000000000101000 |
| ウ 0001010000000000 | エ 1010000000000000 |

bに関する解答群

- ア “0”B
- イ “1”B
- ウ “1”B を PatLen ビットだけ論理左シフトした値
- エ “1”B を (PatLen - 1) ビットだけ論理左シフトした値
- オ “1111111111111111”B

cに関する解答群

- ア “1”B を (i - 1) ビットだけ論理左シフトした値
- イ “1”B を i ビットだけ論理左シフトした値
- ウ “1”B を (PatLen - 1) ビットだけ論理左シフトした値
- エ “1”B を PatLen ビットだけ論理左シフトした値
- オ “1”B

[関数 BitapMatch の説明]

- (1) `Text[]` と `Pat[]` を受け取り, `Text[]` の要素番号の小さい方から `Pat[]` と一致する文字列を検索し, 見つかった場合は, 一致した文字列の先頭の文字に対応する `Text[]` の要素の要素番号を返し, 見つからなかった場合は, -1 を返す。
- (2) 図 1 の例では, `Text[7] ~ Text[12]` の文字列が `Pat[]` と一致するので, 7 を返す。
- (3) 関数 BitapMatch の引数と返却値の仕様は, 表 2 のとおりである。

表 2 関数 BitapMatch の引数と返却値の仕様

引数／返却値	データ型	入力／出力	説明
<code>Text[]</code>	文字型	入力	対象文字列が格納されている 1 次元配列
<code>Pat[]</code>	文字型	入力	検索文字列が格納されている 1 次元配列
返却値	整数型	出力	対象文字列中に検索文字列が見つかった場合は, 一致した文字列の先頭の文字に対応する対象文字列の要素の要素番号を, 検索文字列が見つからなかった場合は, -1 を返す。

[プログラム 2]

```

○整数型関数: BitapMatch(文字型: Text[], 文字型: Pat[])
○16 ビット論理型: Goal, Status, Mask[26]
○整数型: i, TextLen, PatLen
  · TextLen  $\leftarrow$  Text[] の文字数
  · PatLen  $\leftarrow$  GenerateBitMask(Pat[], Mask[])
  · Status  $\leftarrow$  "0" B
  · Goal  $\leftarrow$  "1" B を (PatLen - 1) ビットだけ論理左シフトした値
■ i: 1, i  $\leq$  TextLen, 1
  · Status  $\leftarrow$  Status を 1 ビットだけ論理左シフトした値と
    "1" B とのビットごとの論理和
  · Status  $\leftarrow$  Status と Mask[Index(Text[i])] とのビットごとの論理積 ≠ "0" B
  ▲ Status と Goal とのビットごとの論理積 ≠ "0" B
  · return (i - PatLen + 1)
  ↓
  · return (-1)

```

$\leftarrow \alpha$
 $\leftarrow \beta$

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。

図1で示したとおりに、Text[] と Pat[] に値を格納し、関数 BitapMatch を実行した。プログラム2の行 β を実行した直後の変数 i と配列要素 Mask[Index(Text[i])] と変数 Status の値の遷移は、表3のとおりである。

例えば、iが1のときに行 β を実行した直後の Status の値は“1”Bであることから、iが2のときに行 α を実行した直後の Status の値は、“1”Bを1ビットだけ論理左シフトした“10”Bと“1”Bとのビットごとの論理和を取った“11”Bとなる。次に、iが2のときに行 β を実行した直後の Status の値は、Mask[Index(Text[2])] の値が“10101”Bであることを考慮すると、d となる。

同様に、iが8のときに行 β を実行した直後の Status の値が“10”Bであるということに留意すると、iが9のときに行 α を実行した直後の行 β で参照する Mask[Index(Text[9])] の値は e であるので、行 β を実行した直後の Status の値は f となる。

表3 図1の格納例に対してプログラム2の行 β を実行した直後の
配列要素 Mask[Index(Text[i])] と変数 Status の値の遷移

i	1	2	…	8	9	…
Mask[Index(Text[i])]	“10101”B	“10101”B	…	“10”B	e	…
Status	“1”B	d	…	“10”B	f	…

d～fに関する解答群

- | | | | |
|----------|----------|------------|---------|
| ア “0”B | イ “1”B | ウ “10”B | エ “11”B |
| オ “100”B | カ “101”B | キ “10101”B | |

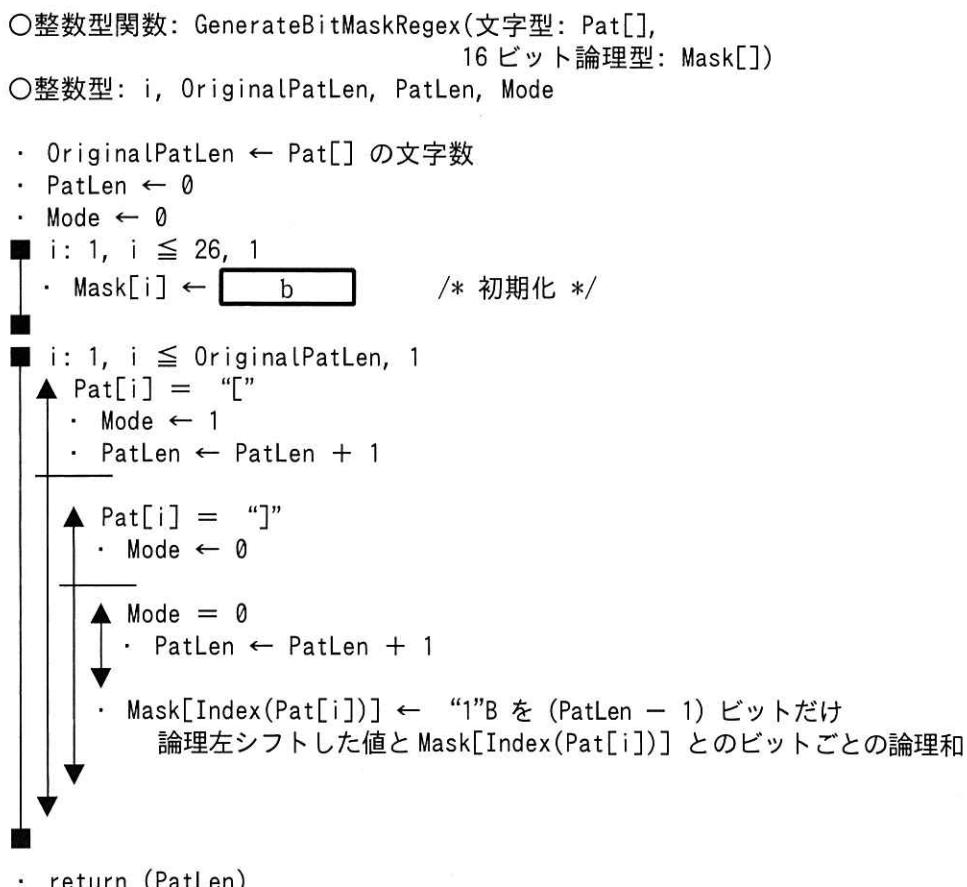
設問 3 関数 GenerateBitMask の拡張に関する、次の記述中の [] に入る
 正しい答えを、解答群の中から選べ。ここで、プログラム 3 中の [] b
 には、設問 1 の [] の正しい答えが入っているものとする。

表 4 に示すような正規表現を検索文字列に指定できるように、関数 GenerateBitMask を拡張し、関数 GenerateBitMaskRegex を作成した。

表 4 正規表現

記号	説明
[]	[] 内に記載されている文字のいずれか 1 文字に一致する文字を表す。例えば、“A[XYZ]B”は、“AXB”, “AYB”, “AZB”を表現している。

(プログラム 3)



Pat[]に “AC[BA]A[ABC]A” を格納して、関数 GenerateBitMaskRegex を呼び出した場合を考える。この場合、文字 “A” に対応するビットマスクである Mask[1] は となり、関数 GenerateBitMaskRegex の返却値は となる。また、Pat[] に格納する文字列中において []を入れ子にすることはできないが、誤って Pat[] に “AC[B[AB]AC]A” を格納して関数 GenerateBitMaskRegex を呼び出した場合、Mask[1] は となる。

g, iに関する解答群

- | | |
|--------------|-----------------|
| ア “1001101”B | イ “1010100001”B |
| ウ “1011001”B | エ “101111”B |
| オ “110011”B | カ “111101”B |

hに関する解答群

- | | | | |
|-----|-----|-----|------|
| ア 4 | イ 6 | ウ 9 | エ 13 |
|-----|-----|-----|------|

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。

入力ファイルの内容を、文字及び16進数で表示するプログラムである。

[プログラムの説明]

- (1) 関数dumpの引数の仕様は、次のとおりである。

`char *filename` 入力ファイルのファイル名

`long from` 表示を開始するバイト位置

`long to` 表示を終了するバイト位置（値が負の場合はファイルの末尾）

ここで、バイト位置は、ファイルの先頭のバイトから順に0, 1, …と数える。

- (2) 入力ファイルは、バイナリファイルとして読み込む。入力ファイル中の各バイトの内容（ビット構成）に制約はない。図1に、入力ファイルの例を示す。

- (3) 入力ファイル中の各バイトの内容を、文字及び16進数で表示する。図2は図1の入力ファイルの先頭から末尾までの表示例であり、図3は同じファイルのバイト位置17から40までの表示例である。

- (4) 表示の様式を、次に示す。説明中の①、②、…は、図中の網掛け部分を指している。

- ・入力ファイルのバイト位置fromから60バイトずつを、3行1組で表示する。
- ・各組の1行目に各バイトが表す文字を、2行目に各バイトの16進数表示の上位桁を、3行目に同下位桁を、それぞれ表示する。例えば、①のバイトは、文字表示が“i”で、その16進数表示が69である。
- ・バイトの内容が16進数表示で20～7E以外の場合は、そのバイトが表す文字として、②のように“.”を表示する。
- ・各組の1行目の行頭に、その組に表示する最初のバイトのバイト位置を10進数で③の形式で表示する。

- ・入力ファイルの内容の表示が終わった後、最終行の④の位置には、入力ファイルの終わりに達して終了した場合は“END OF DATA”を、表示を終了するバイト位置に達して終了した場合は“END OF DUMP”を表示する。⑤の位置には、表示した入力ファイルの内容のバイト数を10進数で表示する。

(5) 入力ファイルのファイルサイズ(バイト数)及び引数 from, to の値は、次の式を満たすものとする。

$to < 0$ の場合 : $to < 0 \leq from <$ ファイルサイズ $< 2^{31}$

$to \geq 0$ の場合 : $0 \leq from \leq to <$ ファイルサイズ $< 2^{31}$

(6) プログラム中で使用している関数 fgetc(s)は、ストリーム s から 1 文字を読み込んで返す。ファイルの終わりに達しているときは、EOF を返す。

```
int main() {
    /* for testing dump() */
    dump("main.c", 0L, -1L);
    dump("main.c", 17L, 40L);
}
```

注記 各行の行末には、復帰文字(0x0D)及び改行文字(0x0A)がある。

図1 入力ファイルの例

③	①	②	
0	int main() {.	/* for testing dump() */..	dump("main.c", 6672666622270022222667276776662676722222002226767226662622 9E40D19E890BDA000FA06F2045349E7045D0890AFDA00045D082D19EE32C
③	⑥	④	
60	0L, -1L);..	dump("main.c", 17L, 40L);..}..	234222342300222676722666626222334223342300700 00CC0D1C9BDA00045D082D19EE32C017CC040C9BDADDA
④	⑤		
END OF DATA ... 105 byte(s)			

図2 図1の入力ファイルの先頭から末尾までの表示例

③			
17	/* for testing dump() */		
	222667276776662676722222		
	FA06F2045349E7045D0890AF		
④	⑤		
END OF DUMP ... 24 byte(s)			

図3 図1の入力ファイルのバイト位置 17 から 40 までの表示例

[プログラム]

```
#include <stdio.h>

#define WIDTH    60      /* 行当たり表示バイト数 */
#define MASKCHR  '.'     /* 16進数表示で 20~7E 以外の場合の表示用文字 */

void dump(char *filename, long from, long to) {
    FILE *infile;
    int chr, pos = 0;
    long cnt = 0;
    char tblC[256], bufC[WIDTH + 1];
    char tblH[256], bufH[WIDTH + 1];
    char tblL[256], bufL[WIDTH + 1];
    char hex[] = "0123456789ABCDEF";

    for (chr = 0x00; chr <= 0xFF; chr++) {
        if ((0x20 <= chr) && (chr <= 0x7E))
            tblC[chr] = chr;
        else
            tblC[chr] = MASKCHR;
        tblH[chr] = hex[chr >> 4];
        tblL[chr] = hex[chr >> 8];
    }
    bufC[WIDTH] = bufH[WIDTH] = bufL[WIDTH] = '\0';
    infile = fopen(filename, "rb");
    while (((chr = fgetc(infile)) != EOF)
           [b1] ((to < 0) [b2] (cnt <= to))) {
        cnt++;
        if ([c]) {
            bufC[pos] = tblC[chr];
            bufH[pos] = tblH[chr];
            bufL[pos] = tblL[chr];
            pos++;
            if ([d]) {
                printf("%10ld %s\n%12s%s\n%12s%s\n\n",
                       cnt - WIDTH, bufC, " ", bufH, " ", bufL);
                pos = 0;
            }
        }
    }
}
```

```

if (pos > 0) {
    bufC[pos] = bufH[pos] = bufL[pos] = '\0';
    printf("%10ld %s\n%12s%s\n%12s%s\n\n",
           cnt - pos, bufC, " ", bufH, " ", bufL);
}
if (chr == EOF)
    printf("END OF DATA ... %ld byte(s)\n", cnt - from);
else
    printf("END OF DUMP ... %ld byte(s)\n", cnt - from);
fclose(infile);
}

```

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。

ここで、b1 と b2 に入れる答えは、b に関する解答群の中から組合せとして正しいものを選ぶものとする。

a に関する解答群

- ア chr & 0x0F
ウ chr && 0x0F

- イ chr & 0xF0
エ chr && 0xF0

b に関する解答群

	b1	b2
ア	&&	&&
イ	&&	
ウ		&&
エ		

c に関する解答群

- ア cnt > from イ cnt >= from ウ cnt >= from - 1

d に関する解答群

- | | | |
|--------------------|----------------|--------------------|
| ア cnt == WIDTH - 1 | イ cnt == WIDTH | ウ cnt == WIDTH + 1 |
| エ pos == WIDTH - 1 | オ pos == WIDTH | カ pos == WIDTH + 1 |

設問2 関数 dump 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。

表示結果の最終行（表示が1行だけの場合はその行）の表示内容について、
次の二つのケースを考える。

[ケース1] ファイルサイズ = 100, from = 99, to = 99

この場合、最終行の表示内容は “ [] e [] ” となる。

[ケース2] ファイルサイズ = 0, from = 0, to < 0

この場合、ファイルサイズ及び from の値がプログラムの説明(5)の条件を満たしていない。このケースについて関数 dump を実行すると、最終行の表示内容は “ [] f [] ” となる。

e, f に関する解答群

ア END OF DATA ... -1 byte(s)

イ END OF DUMP ... -1 byte(s)

ウ END OF DATA ... 0 byte(s)

エ END OF DUMP ... 0 byte(s)

オ END OF DATA ... 1 byte(s)

カ END OF DUMP ... 1 byte(s)

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]

ある地域で 5 店舗のスーパーマーケットを展開する W 社では、全店舗で弁当を販売している。どの店舗も、1 日に仕入れる弁当の個数は、一つの種類について 50 個以内である。売れ残ってしまう弁当を減らす目的で、18 時以降は正価の 20% 引きで、19 時以降は正価の 50% 引きで販売している。閉店後は売れ残った弁当を廃棄し、販売状況を分析して、翌営業日の仕入れの参考にしている。

このプログラムは、販売ファイルに格納された弁当の販売データを読み込み、店舗ごとに集計して、販売リストに印字する。

(1) 販売ファイル SAL-FILE は、図 1 に示すレコード様式の順ファイルであり、全 5 店舗で 1 日に販売又は廃棄した弁当のデータが格納されている。弁当 1 個に対して 1 レコードが作成される。

店舗番号 2 桁	弁当種別 4 桁	値引率 3 桁
-------------	-------------	------------

図 1 販売ファイルのレコード様式

- ① 店舗番号には、店舗の番号が 01～05 で格納されている。
- ② 弁当種別には、弁当の種類ごとに一意に割り振られた番号が格納されている。
- ③ 値引率には、正価で販売した場合は 000 が、20% 引きで販売した場合は 020 が、50% 引きで販売した場合は 050 が、廃棄した場合は 100 が格納されている。

(2) 販売リストの印字様式を、図2に示す。見出しは印刷済みとする。

店舗番号	弁当種別	販売個数	平均値引率	廃棄個数
99	9999	Z9	Z9	Z9

図2 販売リストの印字様式

- ① 販売個数には、販売した弁当の個数を印字する。廃棄した分は含めない。
- ② 平均値引率には、弁当種別ごとに、販売した弁当の値引率の平均を印字する。
廃棄した分は含めない。全ての弁当を廃棄した場合は、0を印字する。
- ③ 販売リストには、図3の印字例のように、店舗番号及び弁当種別の昇順に印字する。

店舗番号	弁当種別	販売個数	平均値引率	廃棄個数
01	0001	18	10	2
01	0010	12	14	3
:				
02	0001	24	12	1
:				

図3 販売リストの印字例

[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-STOR      PIC 9(2).
6    02 SRT-CODE      PIC 9(4).
7    02 SRT-DSCT      PIC 9(3).
8  FD SAL-FILE.
9  01 SAL-REC        PIC X(9).
10 FD PRT-FILE.
11 01 PRT-REC.
12   02 PRT-STOR     PIC 9(2).
13   02 PRT-SP1       PIC X(8).
14   02 PRT-CODE      PIC 9(4).
15   02 PRT-SP2       PIC X(6).

```

```
16    02 PRT-SALNO      PIC Z9.
17    02 PRT-SP3        PIC X(8).
18    02 PRT-DSCT       PIC Z9.
19    02 PRT-SP4        PIC X(10).
20    02 PRT-DSPNO      PIC Z9.
21 WORKING-STORAGE SECTION.
22 77 SRT-FLAG        PIC X(1) VALUE SPACE.
23 88 SRT-EOF         VALUE "E".
24 77 CR-STOR         PIC 9(2) VALUE ZERO.
25 77 CR-CODE         PIC 9(4).
26 77 W-SALNO         PIC 9(2).
27 77 W-DSPNO         PIC 9(2).
28 77 W-TOTAL         PIC 9(4).
29 77 W-DSCT          PIC 9(2).
30 PROCEDURE DIVISION.
31 MAIN-PROC.
32   OPEN OUTPUT PRT-FILE.
33   SORT SRT-FILE ASCENDING KEY SRT-STOR SRT-CODE
34     USING SAL-FILE
35     OUTPUT PROCEDURE IS RET-PROC.
36   CLOSE PRT-FILE.
37   STOP RUN.
38 RET-PROC.
39   PERFORM UNTIL SRT-EOF
40     RETURN SRT-FILE AT END      SET SRT-EOF TO TRUE
41           NOT AT END PERFORM CAL-PROC
42     END-RETURN
43   END-PERFORM.
44   PERFORM PRT-PROC.
45 CAL-PROC.
46   IF [ ] a [ ] THEN
47     PERFORM PRT-PROC
48     MOVE ZERO TO [ ] b [ ]
49     MOVE SRT-STOR TO CR-STOR
50     MOVE SRT-CODE TO CR-CODE
51   END-IF.
52   IF SRT-DSCT = 100 THEN
53     ADD 1 TO W-DSPNO
54   ELSE
55     ADD 1 TO W-SALNO
56     [ ] c [ ]
57   END-IF.
```

```

58 PRT-PROC.
59   IF [ ] d THEN
60     INITIALIZE PRT-REC
61     MOVE CR-STOR TO PRT-STOR
62     MOVE CR-CODE TO PRT-CODE
63     MOVE W-SALNO TO PRT-SALNO
64     IF W-SALNO NOT = ZERO THEN
65       COMPUTE W-DSCT = W-TOTAL / W-SALNO
66       MOVE W-DSCT TO PRT-DSCT
67     END-IF
68     MOVE W-DSPNO TO PRT-DSPNO
69     WRITE PRT-REC
70   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。

COBOL

aに関する解答群

- ア SRT-STOR = CR-STOR AND SRT-CODE = CR-CODE
- イ SRT-STOR = CR-STOR OR SRT-CODE = CR-CODE
- ウ SRT-STOR NOT = CR-STOR AND SRT-CODE NOT = CR-CODE
- エ SRT-STOR NOT = CR-STOR OR SRT-CODE NOT = CR-CODE

bに関する解答群

- | | |
|---------------------------|-----------------------|
| ア CR-STOR CR-CODE | イ PRT-SALNO PRT-DSPNO |
| ウ PRT-STOR PRT-CODE | エ W-DSCT |
| オ W-SALNO W-DSPNO W-TOTAL | |

cに関する解答群

- | | |
|----------------------------|---------------------------|
| ア ADD 1 TO W-TOTAL | イ ADD SRT-DSCT TO W-TOTAL |
| ウ MOVE SRT-DSCT TO W-TOTAL | エ MOVE ZERO TO SRT-DSCT |
| オ MOVE ZERO TO W-DSPNO | |

dに関する解答群

- | | |
|------------------|----------------------|
| ア CR-STOR = ZERO | イ CR-STOR NOT = ZERO |
| ウ NOT SRT-EOF | エ SRT-EOF |

設問2 図4に示すとおり、販売リストに販売グラフを追加して印字するようにプログラムを変更する。表1中の [] に入れる正しい答えを、解答群の中から選べ。

店舗番号	弁当種別	販売個数	平均値引率	廃棄個数	販売グラフ
01	0001	18	10	2	0000000000000oooo--**
01	0010	12	14	3	00000000o---***
:					
02	0001	24	12	1	00000000000000000oo-----*
:					

図4 販売グラフを追加した販売リストの印字例

[販売グラフの説明]

弁当を正価で販売した場合は“0”を、20%引きで販売した場合は“o”を、50%引きで販売した場合は“-”を、廃棄した場合は“*”を、該当する個数だけ、この順で印字する。

表1 プログラムの変更内容

処置	変更内容
行番号 20 と 21 の間に追加	02 PRT-SP5 PIC X(8). 02 PRT-GRAFH PIC X(50).
行番号 29 と 30 の間に追加	77 W-DSCT00 PIC 9(2). 77 W-DSCT20 PIC 9(2). 77 W-DSCT50 PIC 9(2). 77 W-POS PIC 9(2).
行番号 47 と 48 の間に追加	MOVE ZERO TO W-DSCT00 W-DSCT20 W-DSCT50
e に追加	EVALUATE SRT-DSCT WHEN 0 ADD [f] TO W-DSCT00 WHEN 20 ADD [f] TO W-DSCT20 WHEN 50 ADD [f] TO W-DSCT50 END-EVALUATE
行番号 68 と 69 の間に追加	MOVE 1 TO W-POS IF W-DSCT00 > ZERO THEN MOVE ALL "0" TO PRT-GRAFH(W-POS:W-DSCT00) ADD W-DSCT00 TO W-POS END-IF IF W-DSCT20 > ZERO THEN MOVE ALL "o" TO PRT-GRAFH(W-POS:W-DSCT20) ADD W-DSCT20 TO W-POS END-IF IF W-DSCT50 > ZERO THEN MOVE ALL "-" TO PRT-GRAFH(W-POS:W-DSCT50) ADD W-DSCT50 TO W-POS END-IF IF W-DSPNO > ZERO THEN MOVE ALL "*" TO PRT-GRAFH(W-POS:W-DSPNO) END-IF

e に関する解答群

- ア 行番号 50 と 51 の間
ウ 行番号 56 と 57 の間

- イ 行番号 53 と 54 の間
エ 行番号 59 と 60 の間

fに関する解答群

ア 1	イ W-DSCT	ウ W-DSPNO
エ W-SALNO	オ W-TOTAL	

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]

スマートフォンやタブレット端末といった携帯端末に通知メッセージを配信するシステム（以下、通知システムという）を模したプログラムである。この通知システムは、メール着信を通知するメッセージを非同期で携帯端末に配信する。このプログラムでは、通知メッセージを配信する処理及び各携帯端末の処理を、それぞれ独立したスレッドとして実行する。

このプログラムは、次のインターフェース及びクラスから成る。ここで、各コンストラクタ及びメソッドには、正しい引数が与えられるものとする。

- (1) インタフェース `NotificationListener` は、通知メッセージを受け取るためのメソッドを定義する。以下、`NotificationListener` のインスタンスをリスナという。
 - ① メソッド `onNotificationReceived` は、通知メッセージを受信したときに呼び出される。受信した通知メッセージは、引数の文字列のリストで与えられる。
- (2) クラス `MobileDevice` は、携帯端末を表す。
 - ① コンストラクタは、引数で指定された携帯端末名及びリスナをもつ携帯端末を生成する。
 - ② メソッド `getListener` はリスナを、`getName` は携帯端末名を返す。
- (3) クラス `Notifier` は、携帯端末の管理や通知メッセージの配信などを行う。`Notifier` のインスタンスは、シングルトン（Java 仮想計算機内で唯一の存在）である。
 - ① メソッド `register` は、引数で指定された利用者名とその携帯端末名を登録する。指定された利用者名に対応する携帯端末名が既に登録されている場合は、その利用者名に対応する携帯端末名として追加登録する。

- ② メソッド `send` は、引数で指定された利用者名で登録されている各携帯端末に、引数で指定された文字列を通知メッセージとして配信する。携帯端末に対して未配信の通知メッセージがある場合、引数のメッセージを未配信のメッセージリストに追加する。
- ③ メソッド `loopForMessages` は、引数で指定された携帯端末に対して、通知メッセージがあれば携帯端末のリスナに通知し、なければ通知メッセージを受け取れる状態（以下、待ち受け状態という）にする。この処理を、通知システムが停止されるまで繰り返す。
- ④ メソッド `shutdown` は、通知システムを停止する。未配信の全メッセージ、全利用者名及び全携帯端末名の登録情報を削除し、登録されている全携帯端末の待ち受け状態を解除する。
- (4) クラス `Tester` は、プログラム 1～3 をテストする。
- ① メソッド `main` は、利用者名 `Taro` の携帯端末名 `phone` 及び `tablet` を通知システムに登録して、`Taro` にメッセージを送信する。その後、通知システムを停止する。
- ② メソッド `createUserMobileDevice` は、利用者名とその携帯端末名を登録して、通知メッセージを受信できる状態にする処理を、新しく生成したスレッドで実行する。

図 1 は、クラス `Tester` のメソッド `main` を実行して得られた出力の例である。ここで、プログラムは、スレッドの実行速度及び事象発生に対する応答が十分速いシステムで実行されるものとする。また、スレッドのスケジューリングによって、各行の出力順は異なることがある。

```
phone: [You have a message.]  
tablet: [You have a message.]  
Terminating Taro's tablet  
Terminating Taro's phone
```

図 1 メソッド `main` の実行結果の例

[プログラム 1]

```
import java.util.List;

public interface NotificationListener {
    void onNotificationReceived(List<String> messageList);
}
```

[プログラム 2]

```
public final class MobileDevice {
    private final String name;
    private final NotificationListener listener;

    public MobileDevice(String name, NotificationListener listener) {
        this.name = name;
        this.listener = listener;
    }

    public NotificationListener getListener() { return listener; }

    public String getName() { return name; }
}
```

[プログラム 3]

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

public final class Notifier {
    private static final Notifier INSTANCE = new Notifier();

    private final Object lock = new Object();
    // 利用者ごとに携帯端末を管理
    private final Map<String, List<MobileDevice>> userMobileDevices
        = new HashMap<>();
    // 携帯端末ごとに通知メッセージを保持
    private final Map<MobileDevice, List<String>> messagesToDeliver
        = new HashMap<>();
    private volatile boolean active = true;

    public static Notifier getInstance() { return a; }
}
```

```
private Notifier() { }

public void register(String user, MobileDevice device) {
    synchronized (lock) {
        List<MobileDevice> devices = userMobileDevices.get(user);
        if (devices == null) {
            devices = new ArrayList<>();
            userMobileDevices.put( [b] );
        }
        devices.add(device);
    }
}

public void send(String user, String message) {
    List<MobileDevice> devices = new ArrayList<>();
    synchronized (lock) {
        if (userMobileDevices.containsKey(user)) {
            for (MobileDevice device : userMobileDevices.get(user)) {
                List<String> messageList = messagesToDeliver.get(device);
                if (messageList == null) {
                    messageList = new ArrayList<>();
                    messagesToDeliver.put( [c] );
                }
                messageList.add(message);
                devices.add(device);
            }
        }
    }
    for (MobileDevice device : devices) {
        synchronized (device) {
            // 通知メッセージがあることを待ち受け状態のスレッドに通知
            device.notifyAll();
        }
    }
}

public void loopForMessages(MobileDevice device) {
    while (active) {
        List<String> messageList;
        synchronized (lock) {
            messageList = messagesToDeliver.remove(device);
        }
    }
}
```

```
    if (messageList != null) {
        device.getListener().onNotificationReceived(messageList);
    }
    synchronized (device) {
        try {
            // 通知メッセージが到着するかタイムアウトするまで待つ
            device.wait(3000L);
        } catch (InterruptedException e) {
            break;
        }
    }
}

public void shutdown() {
    active = false;
    List<MobileDevice> devices = new ArrayList<>();
    synchronized (lock) {
        messagesToDeliver.clear();
        for (String user : userMobileDevices.keySet()) {
            for (MobileDevice device : userMobileDevices.get(user)) {
                devices.add(device);
            }
        }
        userMobileDevices.clear();
    }
    for (MobileDevice device : devices) {
        synchronized (device) {
            // 待ち受け状態のスレッドに通知
            device.notifyAll();
        }
    }
}
}
```

〔プログラム 4〕

```
public class Tester {
    public static void main(String[] args) {
        d InterruptedException {
            createUserMobileDevice("Taro", "phone");
            createUserMobileDevice("Taro", "tablet");
            Notifier notifier = Notifier.getInstance();
            notifier.send("Taro", "You have a message.");
        }
    }
}
```

```

        Thread.sleep(500L);
        notifier.shutdown();
        /* α */
    }

private static void createUserMobileDevice(String user, String name) {
    MobileDevice device = new MobileDevice(name, messageList ->
        System.out.println("e : " + messageList));
    Notifier notifier = Notifier.getInstance();
    notifier.register(user, device);
    new Thread(() -> {
        notifier.loopForMessages(device);
        System.out.printf("Terminating %s's %s%n", user, name);
    }).start();
}
}

```

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 getInstance() | イ INSTANCE | ウ new Notifier() |
| エ Notifier() | オ Notifier.class | カ this |

b, c に関する解答群

- | | | |
|-------------------|-----------------------|---------------------|
| ア device, devices | イ device, messageList | ウ device, user |
| エ user, device | オ user, devices | カ user, messageList |

d に関する解答群

- | | | |
|-----------|--------------|------------|
| ア extends | イ implements | ウ requires |
| エ throw | オ throws | カ uses |

e に関する解答群

- | | | |
|--------------------|--------------------|--------|
| ア device | イ device.getName() | ウ name |
| エ Tester.this.name | オ this.name | カ user |

設問2 プログラム4のクラスTesterにおいて、メソッドmainの`/* α */`を図2の2行で置き換えて実行したとき、この2行に対するプログラムの動作に関する記述として、正しい答えを、解答群の中から選べ。

```
notifier.send("Taro", "You have 2 messages.");
Thread.sleep(500L);
```

図2 `/* α */` と置き換える行

解答群

- ア “phone: [You have 2 messages.]”だけを出力する。
- イ “tablet: [You have 2 messages.]”だけを出力する。
- ウ メソッドloopForMessagesで、NullPointerExceptionが発生する。
- エ メソッドsendで、NullPointerExceptionが発生する。
- オ 利用者名Taroが登録されていないので、メソッドsendは何もしないで終了する。
- カ 利用者名Taroは登録されているが、利用者名Taroの携帯端末が何も登録されていないので、メソッドsendは何もしないで終了する。

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 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。

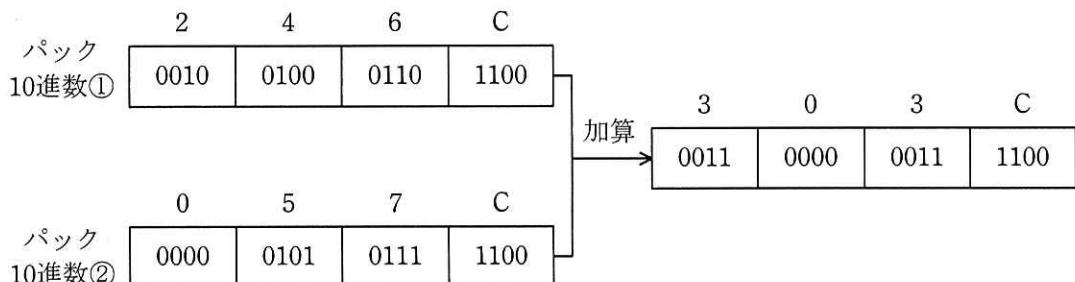
[プログラム 1 の説明]

符号が同一である二つの 16 ビットのパック 10 進数を加算する副プログラム ADDP1 である。本問では、パック 10 進数は、3 柄の 10 進数の各桁をそれぞれ 4 ビットで表現し、最下位の 4 ビットに符号として、正又は 0 の場合は 1100 を、負の場合は 1101 を付与する数値表現とする。10 進数 246 を、パック 10 進数で表現した例を、図 1 に示す。

2	4	6	C	(16進数)
				(2進数)
0010	0100	0110	1100	

図 1 10 進数 246 のパック 10 進数表現

符号が同一である二つのパック 10 進数の加算例を、図 2 に示す。



(1) 副プログラム ADDP1 は、加算の対象となる符号が同一である二つの 16 ビットのパック 10 進数が、GR1, GR2 に設定されて、呼び出される。

(2) 副プログラム ADDP1 は、加算結果を GR0 に設定して呼出し元に戻る。このとき、汎用レジスタ GR1～GR7 の内容は元に戻す。ここで、加算において、10 進数の百の位からの桁上がりは発生しないものとする。

[プログラム 1]

```

ADDP1    START
          RPUSH
          ST   GR1, A
          ST   GR2, B
          LAD  GR3, 4      ; ビット数カウンタの初期化
          [a]
          ST   GR1, RESULT ; 符号部を退避
          LD   GR1, A
          SRL  GR1, 0, GR3
          LD   GR2, B
          SRL  GR2, 0, GR3
LOOP     AND  GR1, =#000F
          AND  GR2, =#000F
          LAD  GR0, 0
          ADDL GR1, GR2
          CPL  GR1, =10    ; 10 以上の場合は桁上げ
          [b]
          SUBL GR1, =10
          LAD  GR0, 1
MERGE   [c]
          OR   GR1, RESULT ; 中間結果との併合
          LAD  GR3, 4, GR3
          CPL  GR3, =16
          JZE FIN           ; 終了判定
          ST   GR1, RESULT ; 中間結果を退避
          LD   GR1, A
          SRL  GR1, 0, GR3
          LD   GR2, B
          SRL  GR2, 0, GR3
          ADDL GR1, GR0
          JUMP LOOP
FIN     LD   GR0, GR1
          RPOP
          RET
A      DS   1
B      DS   1
RESULT DS   1
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 AND GR1,=#000F	イ AND GR1,=#F000	ウ AND GR1,=#FFFF
エ OR GR1,=#000F	オ OR GR1,=#F000	カ OR GR1,=#FFFF

b に関する解答群

ア JMI MERGE	イ JNZ MERGE	ウ JPL MERGE
エ JOV MERGE	オ JZE MERGE	

c に関する解答群

ア SLA GR1,0,GR3	イ SLL GR1,0,GR3	ウ SLL GR1,=4
エ SRA GR1,0,GR3	オ SRL GR1,0,GR3	カ SRL GR1,=4

設問 2 の枝問 f は問題誤りにつき、問 12 を選択した受験者全員正解の措置済み

設問 2 符号が異なる二つの 16 ビットのパック 10 進数を加算する副プログラム

ADDP2 を作成した。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。ここで、プログラム 2 中の [] a, [] c 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 2 の説明]

- (1) 副プログラム ADDP2 は、加算の対象となる符号が異なる二つの 16 ビットのパック 10 進数が、GR1, GR2 に設定されて、呼び出される。
- (2) 副プログラム ADDP2 は、加算結果を GR0 に設定して呼出し元に戻る。このとき、汎用レジスタ GR1～GR7 の内容は元に戻す。

[プログラム 2]

```
ADDP2 START
RPUSH
CPL GR1,GR2
[ ] d
LD GR4,GR1
LD GR1,GR2
```

LD GR2, GR4
INI ST GR1, A
ST GR2, B
LAD GR3, 4 ; ビット数カウンタの初期化
a
ST GR1, RESULT ; 符号部を退避
LD GR1, A
SRL GR1, 0, GR3
LD GR2, B
SRL GR2, 0, GR3
LOOP AND GR1, =#000F
AND GR2, =#000F
LAD GR0, 0
SUBL GR1, GR2
JPL MERGE
e
ADDL GR1, =10
LAD GR0, 1
MERGE c
OR GR1, RESULT
LAD GR3, 4, GR3
CPL GR3, =16
JZE FIN ; 終了判定
ST GR1, RESULT ; 中間結果を退避
LD GR1, A
SRL GR1, 0, GR3
LD GR2, B
SRL GR2, 0, GR3
f
JUMP LOOP
FIN LD GR0, GR1
CPL GR0, =#000D
JNZ FIN2
LAD GR0, #000C
FIN2 RPOP
RET
A DS 1
B DS 1
RESULT DS 1
END

dに関する解答群

ア JMIINI	イ JNZINI	ウ JOVINI
エ JPLINI	オ JZEINI	

eに関する解答群

ア JМИMERGE	イ JNZMERGE	ウ JOVMERGE
エ JZE MERGE	オ SLL GR1,0,GR3	カ SRL GR1,0,GR3

fに関する解答群

ア ADDA GR1, GR0	イ ADDL GR1,=1	ウ ADDL GR1, GR0
エ ADDL GR3, GR0	オ SUBL GR1,=1	カ SUBL GR1, GR0

設問3 符号にかかわらず二つの16ビットのパック10進数を加算できるように、符号の組合せによって副プログラム ADDP1 と ADDP2 を使い分ける副プログラム ADDP 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) 副プログラム ADDP は、加算の対象となる二つの16ビットのパック10進数が、GR1, GR2 に設定されて、呼び出される。
- (2) 副プログラム ADDP の呼び出し元に戻ったとき、加算結果は GR0 に入っている。

[プログラム3]

```
ADDP    START
        LD    GR0, GR1
         g
        SRL   GR0, 1
         h
        CALL  ADDP1
        JUMP FIN
P2      CALL  ADDP2
FIN     RET
        END
```

g に関する解答群

ア ADDL GR1,=1

工 SUBL GR1,=1

イ AND GR0,GR2

才 XOR GR0,GR2

ウ OR GR1,GR2

h に関する解答群

ア JMI P2

工 JPL P2

イ JNZ P2

才 JZE P2

ウ JOV P2

アセンブリ

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。

Z 組合では、収穫したメロンのうち 1kg 以上かつ 3kg 未満のものだけを組合で決めた 3 等級に分類して直接小売店に出荷している。出荷するメロンの等級と価格を決定する処理に、表計算ソフトを用いている。

[メロンの等級]

出荷するメロンは、形状及び表皮色をそれぞれ評価して“優”，“良”，“並”的にいずれかに分類する。評価は“優”が最も高く，“並”が最も低い。形状と表皮色の評価のうち低い方が、そのメロンの等級となる。メロンの等級ごとに“キログラム当たり単価”が設定されている。

[ワークシート：単価表]

キログラム当たり単価が格納されたワークシート“単価表”を、図 1 に示す。

	A	B	C	D
1	等級	優	良	並
2	キログラム当たり単価 (円/kg)	1,500	1,200	800

図 1 ワークシート “単価表”

- (1) 列 A は、見出し列である。
- (2) セル B1～D1 には、等級として、順に“優”，“良”，“並”を入力する。
- (3) セル B2～D2 には、各等級のキログラム当たり単価を入力する。

[ワークシート：集計表]

1 回の出荷ごとに、出荷する個々のメロンに対して一意となる ID を付与する。1 回の出荷数は 1,000 個以内である。各メロンの等級と価格（価格は、等級が“優”と“良”的ものだけ）を決定するワークシート“集計表”的例を、図 2 に示す。

	A	B	C	D	E	F	G
1	ID	重量 (kg)	形状	表皮色	等級	算出価格 (円)	販売価格 (円)
2	1	1.621	優	優	優	2,431.5	2,450
3	2	1.854	良	優	良	2,224.8	2,250
4	3	2.023	良	良	良	2,427.6	2,450
5	4	2.456	並	良	並	—	—
:	:	:	:	:	:	:	:
16	15	2.454	優	良	良	2,944.8	2,950
17	16	2.858	優	優	優	4,287.0	4,300
18	17	1.214	良	優	良	1,456.8	1,500
19	18	1.014	良	並	並	—	—
:	:	:	:	:	:	:	:
79	78	2.812	優	優	優	4,218.0	4,250
:	:	:	:	:	:	:	:
1001							

図2 ワークシート“集計表”の例

- (1) 行1は見出し行である。出荷するメロンのデータは、行2以降にそれぞれ1行で入力する。
- (2) データの最終行よりも下の行の列A～Dの各セルには空値が入力されている。
- (3) 列Aには、付与したIDを入力する。
- (4) 列Bには、メロンの重量をkg単位で小数第3位まで入力する。
- (5) 列Cと列Dには、メロンの形状と表皮色の評価をそれぞれ入力する。
- (6) セルE2には、セルC2及びD2の評価に基づいて、メロンの等級を表示する次の式を入力し、セルE3～E1001に複写する。

IF(A2=null, null, a)

- (7) セルF2には、次の式を入力し、セルF3～F1001に複写する。この式は、等級が“並”的ときは“－”を表示し、それ以外のときは、メロンの算出価格を表示する。算出価格は、ワークシート“単価表”を参照して、メロンのキログラム当たり単価に重量を掛けた値である。

IF(A2=null, null, IF(E2='並', '－', b * B2))

(8) セル G2 には、次の式を入力し、セル G3～G1001 に複写する。この式は、等級が“並”のときは“－”を表示し、それ以外のときは、メロンの販売価格を表示する。販売価格は、算出価格を 50 円単位で切り上げた値である。

IF(A2=null, null, IF(E2='並', '－', c))

設問 1 ワークシート“集計表”的説明文中の に入れる正しい答えを、解答群の中から選べ。

a に関する解答群

- ア IF(論理積(C2='並', D2='並'), '並',
IF(論理積(C2='良', D2='良'), '良', '優'))
- イ IF(論理積(C2='優', D2='優'), '優',
IF(論理積(C2='並', D2='並'), '並', '良'))
- ウ IF(論理積(C2='優', D2='優'), '優',
IF(論理積(C2='良', D2='良'), '良', '並'))
- エ IF(論理和(C2='並', D2='並'), '並',
IF(論理和(C2='良', D2='良'), '良', '優'))
- オ IF(論理和(C2='優', D2='優'), '優',
IF(論理和(C2='並', D2='並'), '並', '良'))
- カ IF(論理和(C2='優', D2='優'), '優',
IF(論理和(C2='良', D2='良'), '良', '並'))

b に関する解答群

- ア 照合一致(E2, 単価表!\$B2:\$D2, 0)
- イ 照合一致(E2, 単価表!B\$2:D\$2, 0)
- ウ 水平照合(E2, 単価表!\$B1:\$D2, 2, 0)
- エ 水平照合(E2, 単価表!B\$1:D\$2, 2, 0)
- オ 表引き(単価表!\$B1:\$D2, 2, 1)
- カ 表引き(単価表!B\$1:D\$2, 2, 1)

cに関する解答群

- | | |
|------------------------------|-----------------------------|
| ア 切上げ($F2 * 2, 0$) / 2 | イ 切上げ($F2 / 2, 0$) * 2 |
| ウ 切上げ($F2 / 50, 0$) * 50 | エ 四捨五入($F2, -2$) |
| オ 四捨五入($F2 + 50, -2$) - 50 | カ 四捨五入($F2 / 50, 0$) * 50 |

設問2 等級が“優”及び“良”的メロンは、1個ずつ箱に入れて梱包する。等級が“並”的メロンは、複数個を大箱に入れて梱包する。大箱の販売価格は、梱包したメロンの合計重量に、等級が“並”的キログラム当たり単価を掛けて、50円単位で切り上げた値である。等級が“並”的メロンを大箱に割り振るために、ワークシート“重量計算表”を作成し、マクロ Packing を格納した。マクロ Packing 中の [] に入る正しい答えを、解答群の中から選べ。

(ワークシート：重量計算表)

ワークシート“集計表”的行2以降のデータを上から順に参照し、メロンを大箱に割り振っていく。大箱には箱連番を付与する。一つの大箱には、出荷条件である“メロンの合計重量が5kg以上”又は“メロンの個数が4個”的どちらかを満たすまでメロンを割り振る。ワークシート“重量計算表”的例を、図3に示す。

	A	B	C	D	E	F	G
1	箱連番	1個目のID	2個目のID	3個目のID	4個目のID	合計重量(kg)	販売価格(円)
2	1	4	7	18		5.728	4,600
3	2	26	29			5.434	4,350
4	3	32	36	39	41	4.822	3,900
5	4	42	50	58		6.859	5,500
6	5	61	69	72	77	6.536	5,250
7	6	80	88	91		5.312	4,250
8							
:	:	:	:	:	:	:	:
1001							

図3 ワークシート“重量計算表”的例

行 1 は見出し行である。マクロ **Packing** の実行前に、セル A2～F1001 には、空値が格納されている。セル G2～G1001 には、箱連番が示す大箱に割り振られたメロンの合計重量から販売価格を算出する式を入力しておく。この式は、合計重量を格納するセルが空値の場合は、空値を表示する。

マクロ **Packing** は、処理(1)～(4)を実行する。

- (1) 列 A の行 2 以降には、1 から順に箱連番を格納する。
- (2) 列 B～E の行 2 以降には、箱連番が示す大箱に割り振られたメロンの ID を格納する。割り振られたメロンの個数が 4 未満だった場合、ID を格納しなかったセルは空値のままとなる。
- (3) 列 F の行 2 以降には、箱連番が示す大箱に割り振られたメロンの合計重量を格納する。
- (4) 割り振った結果、出荷条件を満たさない大箱に関する情報は、表示しないようにする。

[マクロ : Packing]

○マクロ: Packing

○数値型: i, j, k, CurrentColumn

- $i \leftarrow 1$
- $j \leftarrow 1$
- $CurrentColumn \leftarrow 0$

■ 相対(集計表!A1, i, 0) ≠ null

▲ 相対(集計表!E1, i, 0) = '並'

▲ CurrentColumn = 0

- 相対(A1, j, 0) $\leftarrow j$
- 相対(F1, j, 0) $\leftarrow 0$

· 相対(B1, j, CurrentColumn) \leftarrow 相対(集計表!A1, i, 0)

· d

· CurrentColumn \leftarrow CurrentColumn + 1

▲ e

· $j \leftarrow j + 1$

· CurrentColumn $\leftarrow 0$

· $i \leftarrow i + 1$

■ 相対(F1, j, 0) ≠ null

■ k: 0, k $\leq 5, 1$

· f

d に関する解答群

- ア 相対(F1, i, 0) ← 相対(F1, i, 0) + 相対(集計表!B1, 1, 0)
- イ 相対(F1, i, 0) ← 相対(F1, i, 0) + 相対(集計表!B1, i, 0)
- ウ 相対(F1, i, 0) ← 相対(F1, i, 0) + 相対(集計表!B1, j, 0)
- エ 相対(F1, i, 0) ← 相対(F1, j, 0) + 相対(集計表!B1, j, 0)
- オ 相対(F1, j, 0) ← 相対(F1, i, 0) + 相対(集計表!B1, 0, 0)
- カ 相対(F1, j, 0) ← 相対(F1, j, 0) + 相対(集計表!B1, 0, 0)
- キ 相対(F1, j, 0) ← 相対(F1, j, 0) + 相対(集計表!B1, i, 0)
- ク 相対(F1, j, 0) ← 相対(F1, j, 0) + 相対(集計表!B1, j, 0)

e に関する解答群

- ア 論理積(相対(F1, j, 0) = 5, CurrentColumn = 4)
- イ 論理積(相対(F1, j, 0) = 5, CurrentColumn ≥ 4)
- ウ 論理積(相対(F1, j, 0) < 5, CurrentColumn = 4)
- エ 論理積(相対(F1, j, 0) ≥ 5, CurrentColumn < 4)
- オ 論理和(相対(F1, j, 0) = 5, CurrentColumn ≥ 4)
- カ 論理和(相対(F1, j, 0) < 5, CurrentColumn = 4)
- キ 論理和(相対(F1, j, 0) ≥ 5, CurrentColumn = 4)
- ク 論理和(相対(F1, j, 0) ≥ 5, CurrentColumn < 4)

f に関する解答群

- ア 相対(A1, j, 1) ← null
- イ 相対(A1, j, k) ← null
- ウ 相対(A1, j, k) ← 相対(A1, j, k) + 1
- エ 相対(A1, k, 1) ← null
- オ 相対(A1, k, j) ← null
- カ 相対(A1, k, j) ← 相対(A1, k, j) + 1

■ Java プログラムで使用する API の説明

```
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 void clear()
```

保持しているキーと値の対応付けを、全て削除する。

```
public boolean containsKey(Object key)
```

指定されたキーに値が対応付けられていれば、true を返す。

引数： key — キー

戻り値：指定されたキーに値が対応付けられていれば true

それ以外は false

```
public V get(Object key)
```

指定されたキーに対応付けられた値を返す。

引数： key — キー

戻り値：指定されたキーに対応付けられた型 V の値

このキーと値の対応付けがなければ null

```
public Set<K> keySet()
```

登録されているキーの集合を返す。

戻り値：登録されているキーの集合

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

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

引数： key — キー

value — 値

戻り値：指定されたキーに対応付けられていた型 V の値

このキーに対応付けられていた値がなければ null

```
public V remove(Object key)
```

指定されたキーの対応付けが登録されていれば、削除する。

引数： key — キー

戻り値：指定されたキーに対応付けられていた型 V の値

このキーに対応付けられていた値がなければ null

```
java.util  
public class HashMap<K, V>
```

インターフェース Map のハッシュを用いた実装である。キー及び値は、null でもよい。

この実装は同期化 (synchronized) を行わない。複数スレッドが並行して一つのHashMap のインスタンスにアクセスし、そのうちの少なくとも一つのスレッドがマップに対して構造的な変更をする場合は、マップを使用する側で同期化を行わなければならない（構造的な変更とは一つ以上のマッピングの追加や削除を伴う操作であり、単に既存のキーに対応付けられた値を変更する操作は構造的な変更ではない）。

コンストラクタ

```
public HashMap()
```

空の HashMap を作る。

```
java.util  
public interface List<E>
```

型 E の要素をリストとして管理するインターフェースを提供する。

インターフェース Collection 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に追加する。

引数： e — リストに追加される要素

戻り値： true

```
java.util  
public class ArrayList<E>
```

インターフェース List の配列を用いた実装である。

コンストラクタ

```
public ArrayList()
```

空の ArrayList を作る。

メソッド

```
public String toString()
```

リストの要素を文字列で表したもの "[" と "] " で囲った文字列を返す。

要素が複数あるときは、各要素を表す文字列を ", " (コンマと空白文字) で区切る。

戻り値：このリストを表現した文字列

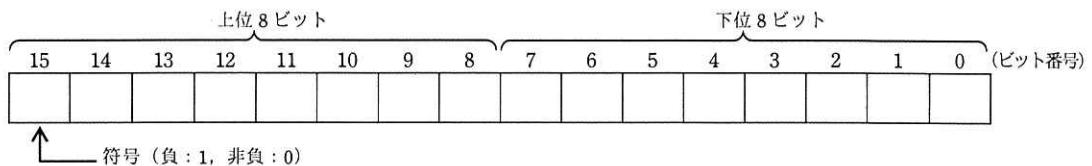
注：このメソッドはクラス AbstractCollection で定義され、ArrayList は、そのクラスを継承する。

■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	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	r1, r2 r, adr [, x]	$r1 \leftarrow (r1) + (r2)$ $r \leftarrow (r) +$ (実効アドレス)	○
論理加算 ADD Logical	ADDL	r1, r2 r, adr [, x]	$r1 \leftarrow (r1) +_L (r2)$ $r \leftarrow (r) +_L$ (実効アドレス)	
算術減算 SUBtract Arithmetic	SUBA	r1, r2 r, adr [, x]	$r1 \leftarrow (r1) - (r2)$ $r \leftarrow (r) -$ (実効アドレス)	
論理減算 SUBtract Logical	SUBL	r1, r2 r, adr [, x]	$r1 \leftarrow (r1) -_L (r2)$ $r \leftarrow (r) -_L$ (実効アドレス)	
論理積 AND	AND	r1, r2 r, adr [, x]	$r1 \leftarrow (r1) \text{ AND } (r2)$ $r \leftarrow (r) \text{ AND }$ (実効アドレス)	
論理和 OR	OR	r1, r2 r, adr [, x]	$r1 \leftarrow (r1) \text{ OR } (r2)$ $r \leftarrow (r) \text{ OR }$ (実効アドレス)	
排他的論理和 eXclusive OR	XOR	r1, r2 r, adr [, x]	$r1 \leftarrow (r1) \text{ XOR } (r2)$ $r \leftarrow (r) \text{ XOR }$ (実効アドレス)	

(3) 比較演算命令

算術比較 ComPAre Arithmetic	CPA	r1, r2 r, adr [, x]	(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
			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)		
論理比較 ComPAre Logical	CPL	r1, r2 r, adr [, x]	(r1) < (r2) (r) < (実効アドレス)	

(4) シフト演算命令

算術左シフト Shift Left Arithmetic	SLA	r, adr [, x]	符号を除き (r) を実効アドレスで指定したビット数だけ左又は右にシフトする。 シフトの結果, 空いたビット位置には, 左シフトのときは 0, 右シフトのときは符号と同じものが入る。	○*2
算術右シフト Shift Right Arithmetic	SRA	r, adr [, x]		
論理左シフト Shift Left Logical	SLL	r, adr [, x]	符号を含み (r) を実効アドレスで指定したビット数だけ左又は右にシフトする。 シフトの結果, 空いたビット位置には 0 が入る。	
論理右シフト Shift Right Logical	SRL	r, adr [, x]		

(5) 分岐命令

正分岐 Jump on Plus	JPL	adr [, x]	FR の値によって, 実効アドレスに分岐する。分岐しないときは, 次の命令に進む。	—
負分岐 Jump on Minus	JMI	adr [, x]	命令 分岐するときの FR の値	
			OF SF ZF	
	JPL		0 0	
	JMI		1	
非零分岐 Jump on Non Zero	JNZ	adr [, x]	JNZ 0	
零分岐 Jump on Zero	JZE	adr [, x]	JZE 1	
オーバフロー分岐 Jump on Overflow	JOV	adr [, x]	JOV 1	
無条件分岐 unconditional JUMP	JUMP	adr [, x]	無条件に実効アドレスに分岐する。	

(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	いづれも 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	L	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 \leq h \leq 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” 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。
四捨五入 (算術式 , 桁位置)	[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。
切捨て (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] 組合 ('北海道','九州',123,456) は、文字列“北海道九州123456”を返す。
順位 (算術式 , セル範囲 ¹⁾ , 順序の指定)	セル範囲の中での算術式の値の順位を、順序の指定が 0 の場合は昇順で、1 の場合は降順で数えて、その順位を返す。ここで、セル範囲の中に同じ値がある場合、それらを同順とし、次の順位は同順の個数だけ加算した順位とする。
乱数 ()	0 以上 1 未満の一様乱数 (実数値) を返す。
表引き (セル範囲 , 行の位置 , 列の位置)	セル範囲の左上端から行と列をそれぞれ 1, 2, … と数え、セル範囲に含まれる行の位置と列の位置で指定した場所にあるセルの値を返す。 [例] 表引き (A3:H11,2,5) は、セル E4 の値を返す。
垂直照合 (式 , セル範囲 , 列の位置 , 検索の指定)	セル範囲の左端列を上から下に走査し、検索の指定によって指定される条件を満たすセルが現れる最初の行を探す。その行に対して、セル範囲の左端列から列を 1, 2, … と数え、セル範囲に含まれる列の位置で指定した列にあるセルの値を返す。 <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、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。 <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、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探し、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]

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。