【改訂版】2023 年度第 1 回から出題

サンプル問題

情報処理技術者能力認定試験

２ 級 第2部

解答時における注意事項

1. 次の表に従って解答してください。

問題番号	問１～問 18
選択方法	全問必須
試験時間	９０分

2. HB の黒鉛筆を使用してください。訂正する場合は，あとが残らないように消しゴムできれいに消し，消しくずを残さないでください。なお，ボールペンや万年筆等で記入した場合は，採点されません。
3. マークシート（解答用紙）の所定の欄に，級種，会場コード，受験番号を記入しマークしてください。また，会場名，氏名及びフリガナを所定の位置に記入してください。
4. 解答は，次の例題にならって，「解答マーク欄」にマークしてください。

〔例題〕　日本の首都はどこか。

ア　東京　　イ　京都　　ウ　大阪　　エ　福岡

正しい答えは“ア　東京”ですから，次のようにマークしてください。

例題 

指示があるまで開いてはいけません。
試験終了後，問題冊子を回収します。

受験会場	
受験番号	
氏　　名	

サーティファイ
情報処理能力認定委員会

改訂版

疑似言語の記述形式

疑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。

〔擬似言語の記述形式などの説明〕

(1) 宣言，注釈及び処理

記述形式		説明
○手続名又は関数名		**手続又は関数**を宣言する。
型名：変数名		**変数**を宣言する。
/* 注釈 */		**注釈**を記述する。
// 注釈		
処理	**変数 ← 式**	**変数**に**式**の値を代入する。
	手続名又は関数名(引数，・・・)	**手続又は関数**を呼び出し，**引数**を受け渡す。
	if **(条件式 1)** 　**処理 1** elseif **(条件式 2)** 　**処理 2** elseif **(条件式 n)** 　**処理 n** else 　**処理 n + 1** endif	選択処理 　**条件式**を上から評価し，最初に真になった**条件式**に対応する**処理**を実行する。以降の**条件式**は評価せず，対応する**処理**も実行しない。 　どの**条件式**も真にならないときは， 　**処理 n + 1** を実行する。 　各**処理**は，0 以上の文の集まりである。 　elseif と**処理**の組みは，複数記述することがあり，省略することもある。 　else と**処理 n + 1** の組みは一つだけ記述し，省略することもある。
	while**(条件式)** 　**処理** endwhile	前判定繰返し処理 　**条件式**が真の間，**処理**を繰り返し実行する。 　**処理**は，0 以上の文の集まりである。
	do 　**処理** while**(条件式)**	後判定繰返し処理 　**処理**を実行し，**条件式**が真の間，**処理**を繰り返し実行する。 　**処理**は，0 以上の文の集まりである。
	for **(制御記述)** 　**処理** endfor	繰返し処理 　**制御記述**の内容に基づいて，**処理**を繰り返し実行する。 　**処理**は，0 以上の文の集まりである。

(2) 演算子と優先順位

演算子の種類		演算子	優先順位
式		(), .	高
単項演算子		not, ＋, －	
二項演算子	乗除	×, ÷, mod	
	加減	＋, －	
	関係	＞, ＜, ≧, ≦, ＝, ≠	
	論理積	and	
	論理和	or	低

注記　演算子 **.** は，メンバ変数又はメソッドのアクセスを表す。
演算子 mod は，剰余算を表す。

(3) 論理型の定数

true（真），false（偽）

(4) 配列

配列の要素は，“[” と “]” の間にアクセス対象要素の要素番号を指定することでアクセスする。なお，二次元配列の要素番号は，行番号，列番号の順に “,” で区切って指定する。

“{” は配列の内容の始まりを，“}” は配列の内容の終わりを表す。ただし，二次元配列において，内部の “{” と “}” に囲まれた部分は，1 行分の内容を表す。

(5) 未定義，未定義の値

変数に値が格納されていない状態を，“未定義” という。変数に “未定義の値” を代入すると，その変数は未定義となる。

改訂版

問１　次の記述中の［　　］に入れる適切な字句を，解答群の中から選べ。

プログラムを実行すると，“［　　］”の順で出力される。

［プログラム］

```
整数型：x，y
x ← 5
y ← x ＋ 3
x を出力
y を出力
```

解答群

ア　3，5　　イ　5，3　　ウ　5，8　　エ　8，8

問２　次の記述中の［　　］に入れる適切な字句を，解答群の中から選べ。ここで，配列の要素番号は１から始まる。

プログラムを実行すると，“［　　］”と出力される。

［プログラム］

```
整数型: x, y, z
整数型の配列: a ← {3, 1, 8, 9, 6, 2, 4}
x ← a[4]
y ← a[6]
z ← x − y × a[1]
z を出力
```

解答群

ア　2　　イ　3　　ウ　9　　エ　21

問３　次のプログラム中の［　a　］，［　b　］に入れる適切な字句の組合せを，解答群の中から選べ。

関数 absNum は，引数で与えられた二つの整数値の差の絶対値を返す。

［プログラム］

```
〇整数型: absNum(整数型: x, 整数型: y)
  整数型: diff
  diff ← x − y
  if (diff が［　a　］より小さい)
    diff ←［　b　］
  endif
  return diff
```

解答群

	a	b
ア	−1	diff − x
イ	−1	diff × (−1)
ウ	0	diff − x
エ	0	diff × (−1)

問４　次のプログラム中の [　　] に入れる適切な字句を，解答群の中から選べ。ここで，配列の要素番号は 1 から始まる。

下記プログラムでは，要素数 10 の整数型の配列 w の要素に，先頭から順に 9，8，7，6，5，4，3，2，1，0 の値を設定する。プログラム実行後の配列要素を以下の図に示す。

	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]
w	9	8	7	6	5	4	3	2	1	0

図　プログラム実行後の配列要素

［プログラム］

```
整数型: i
整数型の配列: w ← {10 個の未定義の値}
for (i を 1 から 10 まで 1 ずつ増やす)
  [　　]
endfor
```

解答群

ア　w[i] ← 9 − i

イ　w[i] ← 10 − i

ウ　w[11 − i] ← i

エ　w[11 − i] ← i ＋ 1

問５　次のプログラム中の［　a　］～［　d　］に入れる適切な字句の組合せを，解答群の中から選べ。

関数 totalPrice は，引数で与えられた商品の金額 price と個数 quantity から，商品の合計金額を求める。さらに，合計金額が 1,000 円未満であれば送料 500 円，1,000 円以上 3,000 円未満であれば送料 200 円，3,000 円以上であれば送料無料とし，商品の合計金額に送料を加算した金額を返す。

［プログラム］

```
〇整数型: totalPrice(整数型: price, 整数型: quantity)
  整数型: total
  total ← price × quantity
  if (total が［　a　］より小さい)
    total ← total ＋［　b　］
  elseif (total が［　c　］より小さい)
    total ← total ＋［　d　］
  endif
  return total
```

解答群

	a	b	c	d
ア	1000	200	3000	500
イ	1000	500	3000	200
ウ	3000	200	1000	500
エ	3000	500	1000	200

問６　次の記述中の［　　］に入れる適切な字句を，解答群の中から選べ。ここで，配列の要素番号は１から始まる。

関数 linearSearch は，引数で文字型の配列 str と文字型の ch を受け取り，str の中から ch と一致する文字を探し，その位置（配列 str の要素番号）を返す。一致する文字がなかった場合は 0 を返す。

ここで，関数 linearSearch を以下のように呼び出すと，戻り値は［　　］となる。

```
linearSearch({"B", "B", "A", "D", "C", "A", "C", "E"}, "C")
```

［プログラム］

```
〇整数型: linearSearch(文字型の配列: str, 文字型: ch)
  整数型: i
  for (i を 1 から str の要素数 まで 1 ずつ増やす)
    if (str[i] が ch と等しい)
      return i
    endif
  endfor
  return 0
```

解答群

ア　0　　イ　2　　ウ　5　　エ　7

問７　次のプログラム中の［　a　］～［　d　］に入れる適切な字句の組合せを，解答群の中から選べ。ここで，配列の要素番号は１から始まる。

手続 printMinMax は，引数で与えられた整数型の配列 array（要素数 1 以上）の中の最小値と最大値を求めて，最小値，最大値の順に出力する。なお，array に格納されている値は，0～999 の範囲である。

［プログラム］

```
○printMinMax(整数型の配列: array)
  整数型: min, max, i
  min ← [ a ]
  max ← [ b ]
  for (i を 1 から array の要素数 まで 1 ずつ増やす)
    if (array[i] が [ c ] より小さい)
      [ c ] ← array[i]
    endif
    if (array[i] が [ d ] より大きい)
      [ d ] ← array[i]
    endif
  endfor
  min を出力
  max を出力
```

解答群

	a	b	c	d
ア	−1	1000	max	min
イ	−1	1000	min	max
ウ	1000	−1	max	min
エ	1000	−1	min	max

問８　次の記述中の［　　］に入れる適切な字句を，解答群の中から選べ。

スタックを操作するプログラムである。スタックは，後に入れたデータを先に取り出すデータ構造である。スタックの操作は，以下の表に示す四つの処理により制御される。なお，スタックに格納できるデータの領域は十分な大きさがある。

手続 stackData を呼び出したときは，［　　］の順に出力される。

表　スタック操作の処理

処理	戻り値	説明
init()	なし	スタックを初期化する。
push(文字列型: s)	なし	スタックに文字列 s をデータとして追加する。
pop()	文字列型	スタックの先頭にあるデータを取り出し，その値を出力する。
peek()	文字列型	スタックの先頭にあるデータを取り出さず，その値を出力する。

［プログラム］

```
〇stackData()
  init()
  push("M")
  push("J")
  pop()
  push("Y")
  push("G")
  peek()
  push("K")
  pop()
  pop()
  pop()
```

解答群

ア　"J", "G", "K", "G", "Y"

イ　"J", "G", "K", "Y", "M"

ウ　"M", "J", "J", "Y", "G"

エ　"M", "J", "Y", "G", "K"

問９　次のプログラム中の [　　] に入れる適切な字句を，解答群の中から選べ。

関数 countBit は，8 ビット型の引数 bits を受け取り，bits の各桁のうち，値が "1" の数を集計して整数値として返す。例えば，関数 countBit を「countBit(10011101)」として呼び出すと，戻り値は 5 となる。

なお，演算子 ∧ はビット単位の論理積，演算子 ≪ は論理左シフト，≫ は論理右シフトを表す。例えば，「value ≪ n」は value の値を n ビットだけ左に論理シフトする。

［プログラム］

```
〇整数型: countBit(8 ビット型: bits)
  整数型: cnt ← 0
  while (bits が 00000000 と等しくない)
    if ((bits ∧ 00000001) が 00000001 と等しい)
      cnt ← cnt ＋ 1
    endif
    [　　]
  endwhile
  return cnt
```

解答群

ア　bits ← bits ∧ 10000000

イ　bits ← bits ∧ 11111110

ウ　bits ← bits ≫ 1

エ　bits ← bits ≪ 1

〔 メ モ 用 紙 〕

問 10　次のプログラム中の [　　] に入れる適切な字句を，解答群の中から選べ。

クラス MergeArray は，要素として 0 ～ 99 の値をもつ，降順にソートされた二つの整数型の配列の値を降順に併合するクラスである。クラス MergeArray の説明を以下の図に示す。

例えば，配列 {80，60，50，40} と {75，65，45，35，20} を併合すると，結果は {80，75，65，60，50，45，40，35，20} となる。

コンストラクタ	説明
MergeArray(整数型の配列: a1, 整数型の配列: a2)	クラス MergeArray のインスタンスを生成する。

メソッド	戻り値	説明
merge()	なし	整数型配列 a1 と a2 の値を降順に併合する。
nextData(整数型: n)	配列要素値	n が 1 のときは，配列 a1 の未併合の先頭要素の値を返す。 n が 2 のときは，配列 a2 の未併合の先頭要素の値を返す。 ただし，未併合の要素がない場合は −1 を返す。
putData(整数型: d)	なし	d を併合結果配列の末尾に追加する。

図　クラス MergeArray の説明

［プログラム］

```
〇merge()
  整数型: d1 ← nextData(1)
  整数型: d2 ← nextData(2)
  while (          )
    if (d1 が d2 以上)
      putData(d1)
      d1 ← nextData(1)
    else
      putData(d2)
      d2 ← nextData(2)
    endif
  endwhile
```

解答群

ア　(d1 が −1 と等しい) and (d2 が −1 と等しい)

イ　(d1 が −1 と等しい) or (d2 が −1 と等しい)

ウ　(d1 が −1 と等しくない) and (d2 が −1 と等しくない)

エ　(d1 が −1 と等しくない) or (d2 が −1 と等しくない)

問 11　次の記述中の [a] , [b] に入れる適切な字句の組合せを，解答群の中から選べ。ここで，配列の要素番号は 1 から始まる。

リングバッファを操作するプログラムである。リングバッファは，読込み位置 rp と書込み位置 wp，および要素数 6 の文字列型の配列 buffer から構成される。

リングバッファへのデータの書込みは，手続 write により行われる。手続 write は，wp が示す配列要素番号の位置にデータを書き込み，wp を 1 加算する。ここで，wp がリングバッファの配列要素数を超えた場合は，wp は 1 に戻る。

リングバッファからのデータの読込みは，関数 read により行われる。関数 read は，rp が示す配列要素番号の位置のデータを戻り値として出力し，rp を 1 加算する。ここで，rp がリングバッファの配列要素数を超えた場合は，rp は 1 に戻る。なお，rp と wp の初期値は 1 である。

手続 job を呼び出したとき，最終的に rp は [a] ，wp は [b] となる。

［プログラム］

```
〇job()
  write("OSAKA")
  write("TOKYO")
  write("NAGOYA")
  read()
  read()
  write("FUKUOKA")
  write("YOKOHAMA")
  read()
  write("SAITAMA")
  write("SENDAI")
  write("KAWASAKI")
  read()
```

解答群

	a	b
ア	4	2
イ	4	3
ウ	5	2
エ	5	3

問 12　次のプログラム中の [　　] に入れる適切な字句を，解答群の中から選べ。

関数 editComma は，文字列型の引数 value（数字のみで構成）を受け取り，1 の位から 3 桁区切りにコンマ文字（“,”）を挿入した文字列を返す。例えば，

editComma("123") の戻り値は，"123"
editComma("1234") の戻り値は，"1,234"
editComma("12345678") の戻り値は，"12,345,678"

となる。

［プログラム］

```
〇文字列型: editComma(文字列型: value)
  整数型: cnt
  文字型: ch
  文字列型: edited ← ""
  cnt ← [　　]
  if (cnt が 0 と等しい)
    cnt ← 3
  endif
  while (value の文字数が 0 より大きい)
    ch ← value の先頭文字
    value ← value の 2 文字目から末尾までの文字列
    if (cnt が 0 と等しい)
      edited の末尾に "," を追加
      cnt ← 3
    endif
    edited の末尾に ch を追加
    cnt ← cnt − 1
  endwhile
  return edited
```

解答群

ア　3 − (value の文字数 mod 3)
イ　4 − (value の文字数 mod 3)
ウ　value の文字数 mod 3
エ　value の文字数 mod 4

問 13　次のプログラム中の [　　] に入れる適切な字句を，解答群の中から選べ。

関数 serializeSeconds は，時（0〜23），分（0〜59），秒（0〜59）が格納されたクラス Time のインスタンス hms を引数で受け取り，0 時 0 分 0 秒からの秒数を整数型で返す。

クラス Time のメンバ変数の説明を以下の表に示す。なお，各メンバ変数は外部から直接アクセス可能とする。

表　クラス Time のメンバ変数の説明

メンバ変数	型	説明
hour	整数型	時（0〜23）
minute	整数型	分（0〜59）
second	整数型	秒（0〜59）

また，関数 makeTime は，整数型の seconds（0 時 0 分 0 秒からの秒数）を引数で受け取り，時分秒に変換してクラス Time のインスタンスを返す。ここで，整数型 ÷ 整数型の商は，小数点以下を切り捨てた整数となる。

［プログラム］

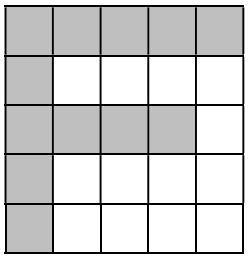
```
〇整数型: serializeSeconds(Time: hms)
  return hms.hour × 3600 ＋ hms.minute × 60 ＋ hms.second

〇Time: makeTime(整数型: seconds)
  Time: hms ← Time() /* クラス Time のインスタンスを生成 */
  hms.hour ← seconds ÷ 3600
  seconds ← seconds mod 3600
  hms.minute ← seconds ÷ 60
  hms.second ← [　　]
  return hms
```

解答群

ア　hms.hour mod 24
イ　seconds − hms.minute
ウ　seconds × 60
エ　seconds mod 60

問 14　次の記述中の [　　] に入れる適切な字句を，解答群の中から選べ。ここで，配列の要素番号は 1 から始まる。

5 行 5 列の図形がある。その図形データを文字型の二次元配列に，“ □ ” と“ ■ ”の文字を使用して表現する。図形の m 行 n 列の要素は，二次元配列の m 行 n 列目の要素に設定する。図形と二次元配列の対応の例を以下の図に示す。



```
{{"■", "■", "■", "■", "■"},
 {"■", "□", "□", "□", "□"},
 {"■", "■", "■", "■", "□"},
 {"■", "□", "□", "□", "□"},
 {"■", "□", "□", "□", "□"}}
```

図　図形と二次元配列の対応の例

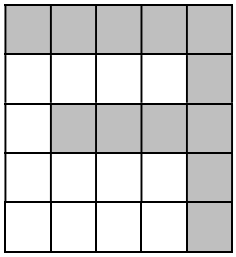
関数 move は，5 行 5 列の図形を表現した文字型の二次元配列 matrix を引数に受け取り，ある規則に従って，二次元配列の要素を移動した文字型の二次元配列 moved を返す。上記の例の図形を引数として渡した場合，moved が表現する図形は，解答群中の [　　] となる。

［プログラム］

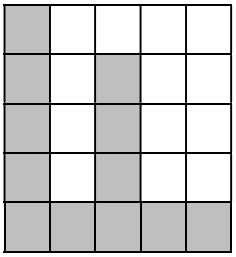
```
〇文字型の二次元配列: move(文字型の二次元配列: matrix)
  文字型の二次元配列: moved ← {{"", "", "", "", ""},
                               {"", "", "", "", ""},
                               {"", "", "", "", ""},
                               {"", "", "", "", ""},
                               {"", "", "", "", ""}}
  /* 要素数 5 の文字型配列を，要素として五つもつ二次元配列 */
  整数型: i, j
  for (i を 1 から 5 まで 1 ずつ増やす)
    for (j を 1 から 5 まで 1 ずつ増やす)
      moved[i, j] ← matrix[j, 6 − i]
    endfor
  endfor
  return moved
```

解答群

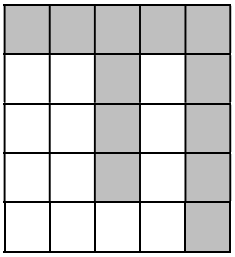
ア



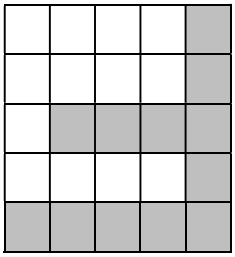
イ



ウ



エ



問 15　A 社は，所在地域の口コミ飲食店情報を収集して会員に提供する Web サービス（以下，A サービスという）を展開している。A サービスを利用するには会員登録をする必要があり、その際に英小文字からなる最大で 6 文字のパスワードを設定する必要がある。

近年，A サービスと同様の Web サービスが外部から攻撃を受けて不正侵入の被害を受けたり，利用者のログインパスワードを推測され，アカウントに不正アクセスされたりする事例が多発している。パスワードとして使用する文字列が短すぎたり容易に推測されやすかったりすると，パスワードを攻撃者に推測され，なりすましの被害に遭いやすいため，A 社の情報システム部では，A サービスのパスワードの強度（パスワードの候補数）を見直すことにした。

A 社情報システム部では，従来のパスワードに対して以下の表に示すパスワードを想定し，その強度を比較した。

表　パスワードの説明と強度比較

項番	パスワードの説明	強度
(1)	英小文字だけで最大 6 文字	（省略）
(2)	英小文字だけで最大 8 文字	a
(3)	英小文字及び英大文字で最大 6 文字	b
(4)	英小文字，英大文字及び数字で最大 16 文字	(2) や (3) と比較して非常に大きい値になる。

注記　強度は，そのパスワードの文字数のうち最大のものを使用したときのものである。

A 社は，表の内容を基にして A サービスのパスワードに使用できる文字種を増やすことにした。

設問　表中の［　a　］，［　b　］に入れる適切な字句の組合せを，解答群の中から選べ。

解答群

	a	b
ア	26^8	26^6
イ	26^8	52^6
ウ	8^{26}	6^{26}
エ	8^{26}	6^{52}

問16　B社は，会員間でメッセージを交換できるWebサービス（以下，Bサービスという）を提供している。Bサービスを利用するには会員登録をする必要があり，その際に利用者IDとパスワードを設定する。

これまでは,利用者が設定した固定長のパスワードはパスワードファイルに平文で保存されていた。しかし，この方法ではパスワードファイルが漏えいすると，Bサービスの全ての利用者のパスワードがそのまま外部に知られてしまうため,危険であるとB社の中で問題視された。そこで，Bサービスの運用担当のC氏は，パスワードファイルの漏えい時に不正ログインが発生しないようにするための対策を行った。

設問　C氏が行った対策として最も適切なものを，解答群の中から選べ。

解答群

ア　ソルト（十分に長いランダムな文字列）をパスワードに連結し保存する。
イ　パスワードに使用できる文字として英小文字（a～z），英大文字（A～Z），数字（0～9）以外に，記号を使用できるようにする。
ウ　平文のパスワードではなく，ハッシュ関数を用いて求めたハッシュ値をパスワードファイルに保存する。
エ　利用者IDの桁数を増やす。

問 17　D 社は小型家電の通信販売会社である。D 社では，インターネット上に公開している通信販売用の Web サイト（以下，D サイトという）を用いて，会員へのネット通販サービスを提供している。

D サイトの会員になるには会員登録が必要になる。その際に，D 社は個人情報の利用目的を提示し，会員登録をしようとしている人は利用目的に同意しなければならない。同意した人は，D サイトの利用者登録画面にて利用者 ID，パスワード，メールアドレス，住所，氏名，電話番号などの情報を登録して会員になることができる。登録された情報は D 社のデータベースサーバに格納される。

1 か月前から，D サイトの会員の一部から“D サイトに登録したメールアドレスに，複数の送信元メールアドレスから迷惑メールが届くようになって困っている”とのクレームが寄せられるようになった。このクレームに対応するため，D 社情報システム部の Y 課長は部下に D サイトの停止を命じ，個人情報の漏えいなどがないかを調査させた。その結果，D サイトの「マイページ」を表示する画面から，D 社の Web サーバを経由してデータベースサーバ上の情報を参照する処理に不備があり，SQL インジェクション攻撃が成功することがわかった。

〔SQL インジェクション攻撃〕

あるデータベースに表のテーブル X があり，列 B の値がブラウザから入力された文字列に一致する行の列 C の値だけを取り出す①の SQL 文があるとする。この SQL 文が実行されるとき，ブラウザから入力された文字列が“$val”の位置にそのまま埋め込まれるものとする。①の SQL 文中の“'”は文字列の直前と直後に付与する識別用文字であり，“;”は SQL 文の末尾を表す文字である。

表　テーブル X

A	B	C
1000	john	35
1001	mike	29
1002	tom	18

```
SELECT C FROM X WHERE B = '$val';                    …①
```

攻撃者がブラウザから「'; SELECT * FROM X; --」のような文字列を入力すると，①の SQL 文の“$val”の位置にこれらの文字列がそのまま埋め込まれ，②の SQL 文のようになる。②の SQL 文の文字列がデータベースサーバに送信されると，二つの SELECT 文が実行されることになる。

```
SELECT C FROM X WHERE B = ''; SELECT * FROM X; --';       …②
```

②の SQL 文の二つ目の SELECT 文「SELECT * FROM X;」が実行されるので，テーブル X の［　a　］がブラウザに表示される。なお，「--」は「以降の文字を注釈として無視する」という SQL 文の制御文字である。

Dサイトの「マイページ」を表示する画面の直前の画面では，利用者が「マイページ」のリンクをクリックすると，③のSQL文の“$id”の部分に，その利用者の利用者IDが埋め込まれたSQL文が作成され，Webサーバを介してデータベースサーバに送られる。

```
SELECT (略) FROM members WHERE userid = '$id' (略);        …③
```

攻撃者が，③のSQL文の“$id”の部分に「 b 」という文字列を何らかの方法で設定してWebサーバに送りつけると，攻撃者は会員の情報が登録されたテーブルmembersの内容を全て参照することができる。なお，このSQL文の実行によってエラーは発生しないものとする。

D社はセキュリティ事故が発生したと判断し，Dサイトの全ての会員に対して謝罪するとともに，Dサイトの告知用ページにおいて事故の経緯や被害状況の説明を行うことにした。また，Dサイトのプログラムを修正し，他にもセキュリティ上の問題点がないかを調査することにした。

設問　記述中の [a] ，[b] に入れる適切な字句の組合せを，解答群の中から選べ。

解答群

	a	b
ア	先頭の行だけ	; SELECT * FROM members;
イ	先頭の行だけ	'; SELECT * FROM members; --
ウ	全部の行	; SELECT * FROM members;
エ	全部の行	'; SELECT * FROM members; --

問18　E 社は従業員数 100 名の情報システム開発企業である。E 社では，自社の開発業務において扱う情報資産に対し，以下の図に示す方法で情報セキュリティリスクアセスメント（以下，リスクアセスメントという）を実施している。

E 社は，１か月前に受注した新規情報システム開発業務（以下，Z 開発業務という）について，リスクアセスメントを実施することになった。

(1) リスクアセスメントの準備
（省略）
(2) リスクの特定
(2-1) 情報資産の洗出し
開発業務において扱う情報資産を漏れなく列挙する。
(2-2) 情報資産の価値の数値化
(2-1)で洗い出した各情報資産の価値を数値化する。情報資産の価値とは，その情報資産が失われたり損なわれたり，外部に漏えいしたりしたときの影響の大きさのことである。E 社では，機密性の観点についてのみ，影響の大きさを 1～3 の値で評価している。
(2-3) 脅威と脆（ぜい）弱性の数値化
(2-1)で洗い出した各情報資産に関する脅威と脆弱性を数値化する。脅威とは，その情報資産が失われたりする現象が発生する可能性の大きさである。脆弱性とは，脅威が発生したときに被害が顕在化する度合いの大きさである。脅威及び脆弱性は，それぞれ 1～3 の値で評価している。
※情報資産の価値，脅威及び脆弱性については，影響，可能性，または度合いの大きさが「大きい」場合に 3,「中程度」の場合に 2,「小さい」,「少ない」または「ない」場合に 1 とする。
(3) リスクの分析と評価
(3-1) リスク値の算出
各情報資産のリスク値を次の式で求める。
リスク値　＝　情報資産の価値 × 脅威 × 脆弱性
※(3-1)のリスク値が 12 以上の情報資産に対しては追加のリスク対策を実施し，そうでない情報資産に対してはリスクを受容するものとする。
（以下，省略）

図　E 社のリスクアセスメントの手順（抜粋）

E 社が洗い出した Z 開発業務に関する情報資産のうち，顧客から提供されるテストデータ（以下，テストデータという）のリスク値を算出することにした。テストデータには顧客の扱う個人情報やその他の極秘情報が含まれており，テストデータが社外に漏れると顧客からの信頼を失うのみならず，顧客の個人情報が漏えいするため機密性の観点から見て大きな影響があると判明している。テストデータが社外に漏れる可能性として，顧客のテストデータをUSBメモリに保存して顧客のオフィスからE社に持ち帰る途中で紛失するというケースが想定される。顧客と E 社との間でインターネットを介してテストデータを送受信することはできず，USB メモリを介してのみテストデータの受け渡しができる状況である。テストデータの受け渡しは頻繁に発生すると見込まれているので，紛失の可能性は大きいと判断された。

この脅威に対して，E 社ではテストデータを USB メモリに格納する際に暗号化の措置を施している。そのため，脆弱性の観点からは，万が一 USB メモリを紛失してもテストデータの内容が外部に漏えいする可能性を低減できるので，情報漏えいの被害が顕在化する度合いは少ないと判断された。

以上から，テストデータのリスク値は［　a　］であり，追加のリスク対策の実施は［　b　］と判断された。

設問　記述中の［　a　］，［　b　］に入れる適切な字句の組合せを，解答群の中から選べ。

解答群

	a	b
ア	9	必要
イ	9	不要
ウ	12	必要
エ	12	不要

〔 メ モ 用 紙 〕

〔 メ モ 用 紙 〕