【改訂版】2023 年度第 2 回から出題

サンプル問題

情報処理技術者能力認定試験

１　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　■　(イ)　(ウ)　(エ)

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

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。

関数 price は，引数で指定されたコンサートの席（S 席，A 席，B 席，自由席）から，その席のチケット料金を返す。各席の料金を表１に示す。

表１　各席の料金

席	料金
S 席	12,000 円
A 席	8,000 円
B 席	6,000 円
自由席	3,500 円

引数には，"S"，"A"，"B"，その他の英字が指定され，その他の英字が指定された場合は自由席となる。

関数 price を引数 "A"，"D"の順で 2 回実行すると，“［　　］”の順で出力される。

〔プログラム〕

```
○整数型: price(文字型: grade)
  整数型: value
  if (grade が "S" と等しい)
    value ← 12000
  elseif (grade が "A" と等しい)
    value ← 8000
  elseif (grade が "B" と等しい)
    value ← 6000
  else
    value ← 3500
  endif
  return value
```

解答群

ア　8000，3500
イ　8000，6000
ウ　12000，3500
エ　12000，6000

問２　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。

関数 sumRange は，引数で渡された整数型の配列 a の要素に対して，同じく引数で渡された先頭要素番号 s から末尾要素番号 e まで（s ≦ e）の各要素の合計値を返す。

〔プログラム〕

```
○整数型: sumRange(整数型の配列: a, 整数型: s, 整数型: e)
  整数型: sum, i
  sum ← 0
  for ([　　])
    sum ← sum ＋ a[i]
  endfor
  return sum
```

解答群

ア　i を 0 から e － s まで 1 ずつ増やす
イ　i を 0 から s まで 1 ずつ増やす
ウ　i を s から e まで 1 ずつ増やす
エ　i を s から e － 1 まで 1 ずつ増やす

問３　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。

手続 copyData は，引数で与えられた文字型の配列 ary の連続した要素を，別の位置に転写する。手続 copyData の引数を表１に示す。

表１　手続 copyData の引数

データ型	引数	説明
文字型の配列	ary	要素を転写する対象の配列
整数型	sp	転写元の先頭要素番号
整数型	dp	転写先の先頭要素番号
整数型	len	転写要素数

なお，転写元の先頭要素番号，転写先の先頭要素番号及び転写要素数について，転写する要素が配列の末尾要素よりも後ろになるような指定はないものとする。例えば，配列要素数が N の場合，sp ＋ len 及び dp ＋ len は N ＋ 1 以下である。ここで，転写元が転写先と等しい場合は，何も処理せずに終了する。また，手続 copyData における引数 ary の各要素の値は，手続内で値が変わると呼出し元もその値が反映されるものとする。

〔プログラム〕

```
○copyData(文字型の配列: ary, 整数型: sp, 整数型: dp, 整数型: len)
  整数型: ofs ← 1
  if (sp が dp と等しくない)
    if (sp が dp より小さい)
      ofs ← −1
      [　　]
    endif
    while (len が 0 より大きい)
      ary[dp] ← ary[sp]
      sp ← sp ＋ ofs
      dp ← dp ＋ ofs
      len ← len − 1
    endwhile
  endif
```

解答群

ア	sp ← sp ＋ len dp ← dp ＋ len
イ	sp ← sp ＋ len － 1 dp ← dp ＋ len － 1
ウ	sp ← sp － 1 dp ← dp － 1
エ	sp ← sp － len dp ← dp － len

問４　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。

関数 rotateBit は，16 ビット型の引数 bits と，整数型の引数 cnt を受け取り，bits を cnt ビット分だけ右回転した値を返す。例えば，図１のように関数 rotateBit を「rotateBit(0111110010100101，5)」として呼び出すと，戻り値は0010101111100101 となる。

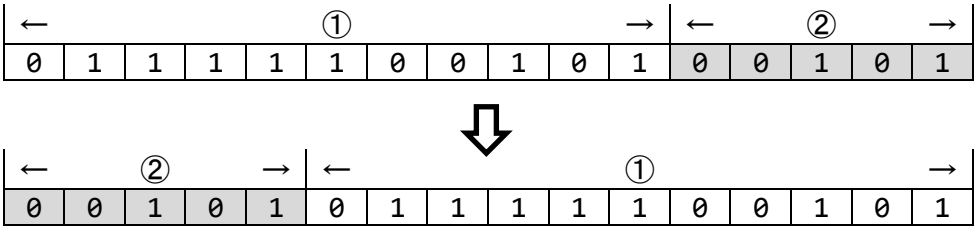


図１　5 ビット分右回転した結果（例）

なお，演算子∧はビット単位の論理積，演算子∨はビット単位の論理和，≫は論理右シフトすることを表す。例えば，「value ≫ n」は value の値を n ビットだけ右に論理シフトする。

〔プログラム〕

```
〇16 ビット型: rotateBit(16 ビット型: bits，整数型: cnt)
  16 ビット型: ck
  while (cnt が 0 より大きい)
    ck ← bits ∧ 0000000000000001
    bits ← bits ≫ 1
    if (ck が 0000000000000001 と等しい)
      bits ← bits ∨ 1000000000000000                    …（α）
    endif
    cnt ← cnt − 1
  endwhile
  return bits
```

「rotateBit(1100001111001001，8)」として呼び出すと，（α）の処理は［　　］回実行される。

解答群

ア　2　　イ　3　　ウ　4　　エ　5

問５　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 decToHex は，引数で与えられた整数値（10 進数表現で 0～65535）を 16 進数の文字列に変換して返す。例えば，「decToHex(0)」として呼び出すと "0" を，「decToHex(12)」として呼び出すと "C" を，「decToHex(512)」として呼び出すと "200" を，「decToHex(65535)」として呼び出すと "FFFF" を返す。関数 decToHex から呼び出される関数 getHexChar は，引数で与えられた整数値(10 進数表現で 0～15)を "0" ～ "9" ，"A" ～ "F" の 16 進数文字列（1 文字）に変換して返す。

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〕

```
〇文字列型: decToHex(整数型: value)
  文字列型: hex ← ""
  整数型: r
  while (value が 0 と等しくない)
    r ← [ a ]
    hex ← getHexChar(r) と hex をこの順に連結した文字列
    value ← [ b ]
  endwhile
  if (hex が "" と等しい)
    hex ← "0"
  endif
  return hex
```

解答群

	a	b
ア	value ÷ 16	value mod 10
イ	value ÷ 16	value mod 16
ウ	value mod 16	value ÷ 10
エ	value mod 16	value ÷ 16

問６　次のプログラム中の［　a　］，［　b　］に入れる適切な字句を，解答群の中から選べ。

関数 parse は，整数型の state（状態番号）及び文字型の ch を引数で受け取り，以下の状態遷移表に従って遷移後の状態番号を返す。なお，表１の␣は空白文字を表す。

表１　状態遷移表

		ch		
		数字 ("0"～"9")	英大文字 ("A"～"Z")	空白文字 ("␣")
state （状態番号）	0	1	2	0
	1	1	−1	0
	2	−1	2	0

例えば，「parse(0，"A")」と呼び出すと，戻り値は 2 となる。また「parse(2，"7")」と呼び出すと，戻り値は −1 となる。なお，引数で渡される state は 0～2 の値であり，ch は，数字，英大文字，空白文字である。

〔プログラム〕

```
〇整数型: parse(整数型: state, 文字型: ch)
  if (ch が [   a   ] と等しい)
    state ← 0
  else
    if (state が 0 と等しい)
      if (ch が 数字 と等しい)
        state ← 1
      else
        state ← 2
      endif
    elseif (state が 1 と等しい)
      if (ch が [   b   ] と等しい)
        state ← −1
      endif
    else
      if (ch が 数字 と等しい)
        state ← −1
      endif
    endif
  endif
  return state
```

解答群

	a	b
ア	英大文字	数字
イ	英大文字	空白文字
ウ	空白文字	数字
エ	空白文字	英大文字

問7　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。

手続 group は，引数で与えられた整数型の配列 array（要素数 1 以上）の先頭要素を基準値とし，基準値未満の値と基準値以上の値にグループ分けし，配列 array に格納する。例えば，group を引数 {63，90，8，51，75，27，68} で呼び出すと，先頭要素 63（基準値）は array の要素番号 4 に格納され，63 未満のグループと 63 以上のグループに分かれる。結果，配列 array の内容は，{27，51，8，63，75，90，68} となる。このとき，プログラム中の（α）は［　　］回実行される。なお，手続 group における引数 array の各要素の値は，手続内で値が変わると呼出し元もその値が反映されるものとする。

〔プログラム〕

```
〇group(整数型の配列: array)
  整数型: x，i，j，step
  x ← array[1]
  i ← 1
  j ← array の要素数
  step ← 1
  while (i が j より小さい)
    if (step が 1 と等しい)
      if (array[j] が x より小さい)
        array[i] ← array[j]                          …（α）
        i ← i ＋ 1
        step ← 2
      else
        j ← j － 1
      endif
    else
      if (array[i] が x 以上)
        array[j] ← array[i]
        j ← j － 1
        step ← 1
      else
        i ← i ＋ 1
      endif
    endif
  endwhile
  array[j] ← x
```

解答群

ア　1　　イ　2　　ウ　3　　エ　4

〔 メ モ 用 紙 〕

問８　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。

単方向リストを操作するクラス ListManager である。ListManager は，リスト要素を保持するクラス ListItem のインスタンスを利用して単方向リスト構造を管理する。ListItem の説明を以下の図１に示す。

メンバ変数	型	説明
id	整数型	保持するデータ
next	ListItem	次要素のインスタンス

コンストラクタ	説明
ListItem(整数型: id)	id をデータとして保持し，next を空状態に初期化する。

メソッド	戻り値	説明
setNext(ListItem 型: p)	なし	メンバ変数 next に，引数で指定した次要素のインスタンス p を格納する。
getNext()	ListItem	次要素のインスタンスを取得する。
getId()	整数型	保持しているデータを取得する。

注記　「空状態」とは，空の状態を表す値が格納されていることを指す。また，next の次要素がない場合，next の値は空状態である。

図１　クラス ListItem の説明

ListManager は，メンバ変数として，リストの先頭要素のインスタンスを保持する ListItem 型の head をもつ。リストがデータを一つも保持していないとき head は空状態となる。ListManager の説明を以下の図２に示す。

メンバ変数	型	説明
head	ListItem	リストの先頭要素のインスタンス

メソッド	戻り値	説明
remove(整数型: id)	なし	id と一致する要素をリストから削除する。

図２　クラス ListManager の説明

次に示すプログラムは，ListManager クラスのメソッド remove で，引数で指定された id と一致する要素を探し出し，最初に見つけた一致する要素をリストから削除する。一致する要素がない場合は，リストの状態は変化しない。

〔プログラム〕

```
○remove(整数型: id)
  ListItem: target, prev
  prev ← 空状態の値
  target ← head
  while (target が空状態と等しくない)
    if (target.getId() が id と等しい)
      if (prev が 空状態と等しい)
        head ← head.getNext()
      else
        [          ]
      endif
      target ← 空状態の値
    else
      prev ← target
      target ← target.getNext()
    endif
  endwhile
```

解答群

ア　prev.setNext(head)

イ　prev.setNext(target.getNext())

ウ　target.setNext(head)

エ　target.setNext(prev.getNext())

問９　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。ここで，配列の要素番号は１から始まる。

関数 binarySearch は，2 分探索法により，あらかじめ昇順に値が並んでいる整数型の配列 array の中から，目的の値を探し出し，その値の配列要素内の要素番号を返す。binarySearch は，次の手順で目的の値を探し出す。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。

① 引数で与えられた探索領域の先頭要素番号 h が末尾要素番号 t より大きい場合，目的の値 v が見つからなかったとして 0 を返す。
② 探索領域の中央の要素番号を m とする。
③ 配列内の要素番号 m の値と目的の値 v とを比較し，一致すればその要素番号を返す。
④ 配列内の要素番号 m の値が目的の値 v より小さければ，探索領域の先頭要素番号を m ＋ 1，末尾要素番号を t として，関数 binarySearch を再帰的に呼び出し，その戻り値を返す。
⑤ 配列内の要素番号 m の値が目的の値 v より大きければ，探索領域の先頭要素番号を h，末尾要素番号を m － 1 として，関数 binarySearch を再帰的に呼び出し，その戻り値を返す。

〔プログラム〕

```
○整数型: binarySearch(整数型の配列: array, 整数型: h,
                       整数型: t, 整数型: v)
  整数型: m
  if (h が t より大きい)
    return 0
  endif
  m ← (h ＋ t) ÷ 2
  if (array[m] が v と等しい)
    return m
  elseif (array[m] が v より小さい)
    return binarySearch(array, [ a ])
  else
    return binarySearch(array, [ b ])
  endif
```

解答群

	a	b
ア	h, m − 1, v	m ＋ 1, t, v
イ	h ＋ 1, m, v	m, t − 1, v
ウ	m, t − 1, v	h ＋ 1, m, v
エ	m ＋ 1, t, v	h, m − 1, v

問10　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。

クラス HashSet はハッシュテーブルを表し，そのインスタンスは，コンストラクタで渡された分割値 div を要素数とする配列（各配列要素には，複数の文字列データを格納できる）により，データを管理する。HashSet のインスタンスは，与えられた文字列データ（英大文字のみ）からそのハッシュ値(0 ～ div － 1)を求め，そのハッシュ値を要素番号とする配列要素に文字列データを登録する。同じハッシュ値をもつデータが既に配列要素として存在していた場合は，現在配列要素で管理している文字列データはそのままに，新たな文字列データも配列要素として追加する。HashSet の説明を図１に示す。

メンバ変数	型	説明
div	整数型	分割値

コンストラクタ	説明
HashSet(整数型: div)	分割値を指定して，クラス HashSet のインスタンスを生成する。

メソッド	戻り値	説明
add(文字列型: str)	なし	ハッシュテーブルに文字列データを登録する。
has(文字列型: str)	論理型	ハッシュテーブルに文字列が存在する場合は true を，存在しない場合は false を返す。
hashValue(文字列型: str)	整数型	文字列データのハッシュ値を返す。 ※クラス HashSet のメソッドのみから呼び出せるアクセス権である。
charToInt(文字型: ch)	整数型	文字 ch（英大文字）の値を整数値に変換する。変換される値は，"A"～"Z"がそれぞれ 1～26 となる。 ※クラス HashSet のメソッドのみから呼び出せるアクセス権である。

図１　クラス HashSet の説明

以下のプログラムに，メソッド hashValue の内容を示す。

〔プログラム〕

```
〇整数型: hashValue(文字列型: str)
  整数型: i, value
  value ← 0
  for (i を 1 から str の文字数まで 1 ずつ増やす)
    value ← value ＋ charToInt(str の i 番目の文字)
  endfor
  return value mod div
```

次のプログラムを実行した結果，(1)～(4)の戻り値をコンマで区切って出力すると，［ a ］となる。(3)の「hs.has("GABCDEF")」処理において，メソッド hashValue が引数"GABCDEF"で呼び出され，そのときのメソッド hashValue の戻り値は，［ b ］となる。

〔プログラム〕

```
HashSet: hs ← HashSet(13)を呼び出してインスタンスを生成
hs.add("ABCDEFG")
hs.add("OPQ")
hs.add("GABCDEF")
hs.has("OPQ")       …(1)
hs.has("OQQ")       …(2)
hs.has("GABCDEF")   …(3)
hs.has("FGABCDE")   …(4)
```

解答群

	a	b
ア	true, false, true, false	2
イ	true, false, true, false	3
ウ	true, true, false, true	2
エ	true, true, false, true	3

問 11　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。

手続 summary は，テキスト形式の入力ファイル名 infile と，テキスト形式の出力ファイル名 outfile を引数で受け取る。infile は n 行（n ≧ 1）のレコードで構成され，1 行には，都道府県コード（2 桁）と発送方法（1 桁）が，コンマ（","）区切りで記述されている。都道府県コードは "01" ～ "47" の数字で，発送方法は表 1 に示す。

表 1　発送方法

発送方法	内容
"1"	普通郵便
"2"	簡易書留
"3"	本人限定郵便

infile のレコードは，都道府県コードの昇順に並んでおり，同一都道府県コード内では，発送方法の昇順に並んでいる。

手続 summary では，都道府県コードと発送方法の組合せごとに，通数（入力のレコード数）を算出して，outfile に，都道府県コード，発送方法，通数の順にコンマ区切りで出力する。なお，ファイルの入出力に関してエラーは発生しないものとする。

以下の図 1 に，infile を入力したときに出力される outfile の例を示す。

```
01,1
01,1
01,2
01,3
01,3
01,3
02,3
02,3
04,1
04,1
04,1
04,1
```

```
01,1,2
01,2,1
01,3,3
02,3,2
04,1,4
```

図 1　infile（左）と outfile（右）

〔プログラム〕

```
〇summary(文字列型: infile, 文字列型: outfile)
  文字列型: line, cc, ch, sc, sh
  整数型: cnt ← 0
  論理型: isFirst ← true
  infile をファイル名としてファイルを読み込みオープンする
  outfile をファイル名としてファイルを書き込みオープンする
  while (infile にレコードが残っている)
    line ← infile から 1 レコード読み込み
    cc ← line の 1 項目      //  都道府県コード
    ch ← line の 2 項目      //  発送方法
    if (isFirst が true)
      isFirst ← false
    else
      if ( [ a ] )
        outfile に sc, sh, cnt をこの順にコンマ区切りで出力する
        cnt ← [ b ]
      endif
    endif
    sc ← cc
    sh ← ch
    cnt ← cnt + 1
  endwhile
  outfile に sc, sh, cnt をこの順にコンマ区切りで出力する
  outfile をファイル名としたファイルをクローズする
  infile をファイル名としたファイルをクローズする
```

解答群

	a	b
ア	(cc が sc と等しい) or (ch が sh と等しい)	0
イ	(cc が sc と等しい) or (ch が sh と等しい)	1
ウ	(cc が sc と等しくない) or (ch が sh と等しくない)	0
エ	(cc が sc と等しくない) or (ch が sh と等しくない)	1

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 areaTriangle は，三角形の３辺の長さ a，b，c を引数で受け取り，その三角形の面積を，小数第２位を四捨五入して，小数第１位までの値として返す。ここで，三角形の面積は，次の公式を用いて計算する。

$$s = \frac{a+b+c}{2}$$

$$\text{面積} = \sqrt{s \times (s-a) \times (s-b) \times (s-c)}$$

なお，関数 areaTriangle は，関数 sqrt を使用して平方根（実数値）を求める。また，小数第１位を四捨五入した実数値を求める関数 round も使用する。例えば，round(12.45)の戻り値は 12.0，round(12.54)の戻り値は 13.0 となる。

〔プログラム〕

```
〇実数型: areaTriangle(実数型: a, 実数型: b, 実数型: c)
  実数型: s
  s ← (a + b + c) ÷ 2
  return ［　　］
```

解答群

ア　round(s × sqrt((s − a) × (s − b) × (s − c)) ÷ 10) × 10
イ　round(s × sqrt((s − a) × (s − b) × (s − c)) × 10) ÷ 10
ウ　round(sqrt(s × (s − a) × (s − b) × (s − c)) ÷ 10) × 10
エ　round(sqrt(s × (s − a) × (s − b) × (s − c)) × 10) ÷ 10

〔 メ モ 用 紙 〕

問 13　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。ここで，配列の要素番号は 1 から始まる。

関数 comp は，文字列 str（1 文字以上）とクラス CompData の配列 cd を引数で受け取り，str の先頭から文字を走査し，同一文字が連続する数（1 以上）を求め，その情報を配列 cd に格納する。また，配列 cd に格納した情報の数を返す。ここで，配列 cd の要素数は十分にある。

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

表　クラス CompData のメンバ変数の説明

メンバ変数	型	説明
ch	文字型	文字
rep	整数型	連続する数

〔プログラム〕

```
○整数型: comp(文字列型: str, CompData 型の配列: cd)
  整数型: cnt, i, j
  文字型: sv
  j ← 1
  sv ← str[1]
  cnt ← 1
  for (i を 2 から str の文字数まで 1 ずつ増やす)
    if (sv が str[i] と等しい)
      cnt ← cnt ＋ 1
    else
      cd[j].ch ← [ a ]
      cd[j].rep ← cnt
      j ← j ＋ 1
      sv ← str[i]
      cnt ← [ b ]
    endif
  endfor
  cd[j].ch ← [ a ]
  cd[j].rep ← cnt
  return j
```

解答群

	a	b
ア	str[i]	1
イ	str[i]	cnt ＋ i
ウ	sv	1
エ	sv	cnt ＋ i

問 14　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。ここで，配列の要素番号は 1 から始まる。

東西と南北に碁盤の目のように整備された道路がある。この道路に遠隔操作のロボットを置き，F（前進），R（右 90 度回転），L（左 90 度回転）の三つのコマンドによりロボットを操作する。F コマンドでは，１区画分前進する。なお，矢印の先端がロボットの向きを表す。なお，道路が行き止まりで前進できないときの F コマンドは無効とし，ロボットの状態は変化しない。F，R，L の三つのコマンドによるロボットの移動を，それぞれ以下の図１～図３に示す。

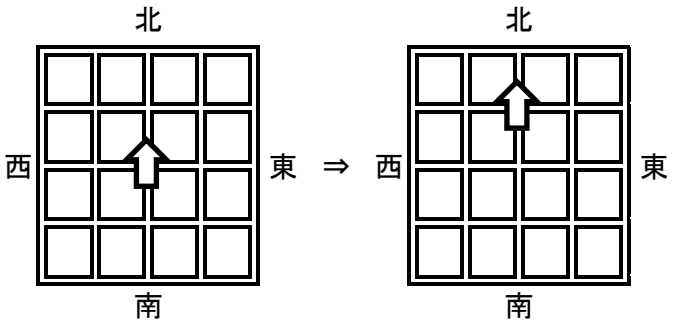


図１ F コマンドによるロボットの移動

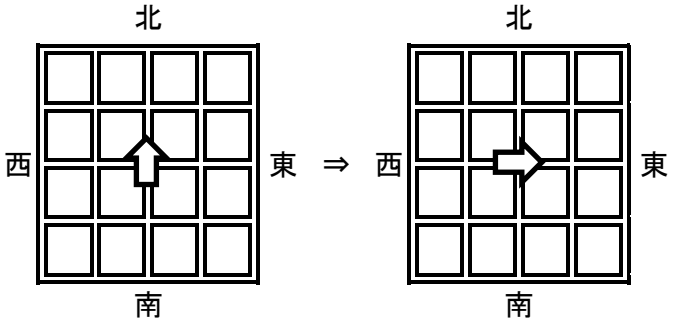


図２ R コマンドによるロボットの回転

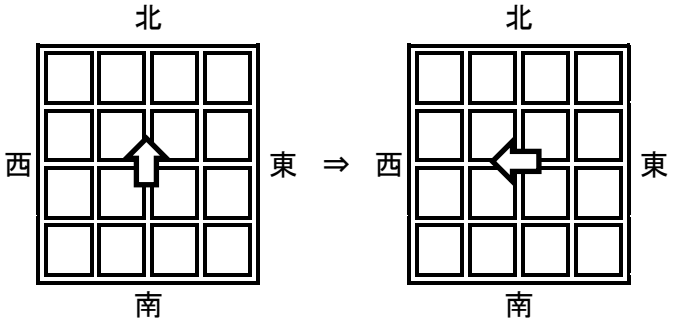


図３ L コマンドによるロボットの回転

ロボットの向きが文字型の "N"（北向き），"E"（東向き），"S"（南向き），"W"（西向き）で表されるとき，関数 turnRight は，引数に現在のロボットの向きを受け取り，R コマンドにより回転した後のロボットの向きを返す。同様に，関数 turnLeft は，引数に現在のロボットの向きを受け取り，L コマンドにより回転した後のロボットの向きを返す。

〔プログラム〕

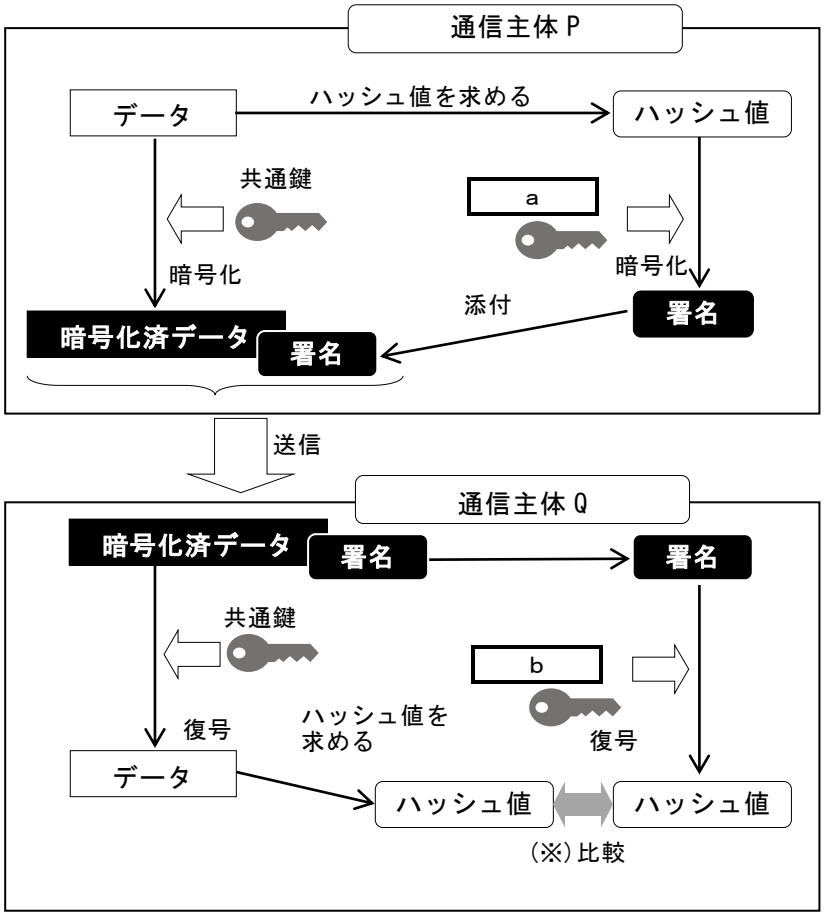
```
○文字型: turnRight(文字型: d)
  文字型の配列: dir ← {"N", "E", "S", "W", "N"}
  文字型: tr ← " "
  整数型: i ← 1
  while ((i が dir の要素数以下) and (tr が " " と等しい))
    if (dir[i] が d と等しい)
      tr ← [   a   ]
    endif
    i ← i ＋ 1
  endwhile
  return tr

○文字型: turnLeft(文字型: d)
  整数型: i
  for ([   b   ])
    d ← turnRight(d)
  endfor
  return d
```

解答群

	a	b
ア	dir[i － 1]	i を 1 から 2 まで 1 ずつ増やす
イ	dir[i － 1]	i を 1 から 3 まで 1 ずつ増やす
ウ	dir[i ＋ 1]	i を 1 から 2 まで 1 ずつ増やす
エ	dir[i ＋ 1]	i を 1 から 3 まで 1 ずつ増やす

問 15　インターネット上で情報を安全に送受信するためには，データを暗号化する必要性、また，通信先がなりすましでないことを確認したり，自身がなりすましでないことを通信先に対して証明したりする必要がある。ある通信主体 P と通信主体 Q との間で通信を行う前に，Q が P を認証する（なりすましでないことを確認する）ために，デジタル署名が用いられる。その仕組みを以下の図 1 に示す。



注記　共通鍵は，通信主体 P，Q の間で事前に共有した鍵である。

図 1　デジタル署名の概要

図 1 の(※)で示す比較において二つのハッシュ値が一致していれば，Q は P がなりすましでないことを確認できる。

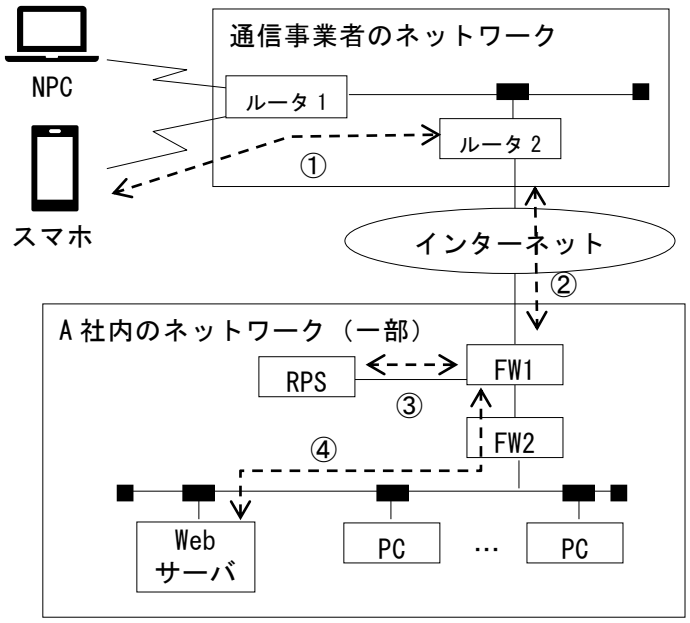
設問　図 1 中の［　a　］，［　b　］に入れる適切な字句の組合せを，解答群の中から選べ。

解答群

	a	b
ア	P の公開鍵	P の公開鍵
イ	P の公開鍵	P の秘密鍵
ウ	P の秘密鍵	P の公開鍵
エ	P の秘密鍵	P の秘密鍵

〔　メ　モ　用　紙　〕

問16　製造業を営むA社は，インターネットを利用した営業支援システム（以下，システムという）を運用している。社外にいるA社の営業部員は，所持している社用のノートPC（以下，NPCという）や社用のスマートフォン（以下，スマホという）からA社のWebサーバにアクセスし，Webサーバ上で稼働しているシステムを利用する。システムの構成を以下の図1に示す。



FW：ファイアウォール
RPS：リバースプロキシサーバ

図1　システムの構成

RPSは社外からシステムに対して行われるアクセスをWebサーバの代わりに受け付け，社外からのアクセスをWebサーバに中継する。社外のNPCやスマホ上のWebブラウザからはRPSがWebサーバとして見える。

当初，図1中の①～④の全ての経路においてHTTPが用いられる予定であった。しかし，A社の情報セキュリティ部門から，システムにログインする際の利用者IDとパスワードなどがインターネット上で盗聴されるリスクがあり，このままの運用は危険と判断された。

そこで，RPSにデジタル証明書と秘密鍵を格納し，Webブラウザから RPSまでの通信にはHTTP over TLS（以下，HTTPSという）を使用することにした。RPSはデジタル証明書をWebブラウザに示すことで自身の真正性を証明する。また，デジタル証明書に格納された公開鍵と，RPSが秘匿する秘密鍵の二つの鍵を用いて，暗号化に用いる共通鍵をWebブラウザと共有し，HTTPSにおいて暗号化された通信を行う。RPSは，Webブラウザから HTTPSで受け取った情報を復号し，通信プロトコルをHTTPSからHTTPに変換して，Webサーバに情報を転送する。

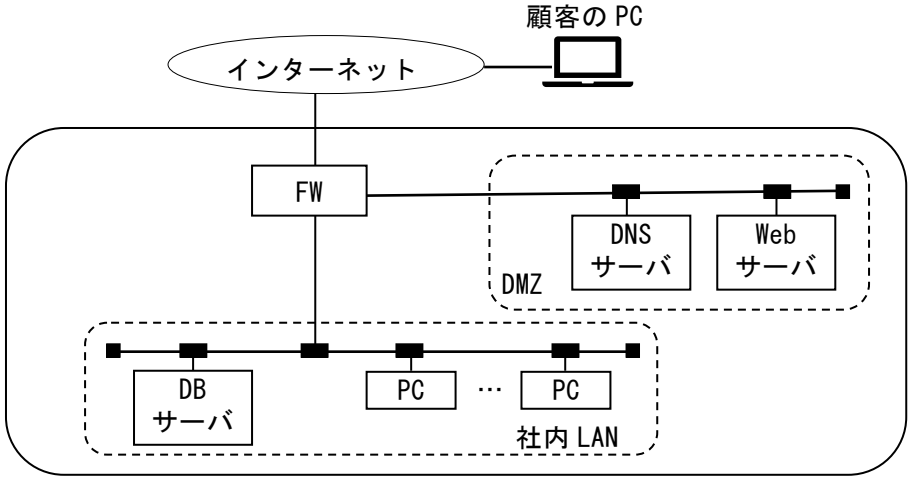
設問　社外の NPC やスマホ上の Web ブラウザからシステムにアクセスするとき，図 1 中の①～④の各通信経路を，(a)HTTPS だけが通信に使われるもの，(b)HTTP だけが通信に使われるもの，(c)HTTPS と HTTP の両方が通信に使われるものの三つのいずれかに分類した結果として適切な組合せを，解答群の中から選べ。

解答群

	①	②	③	④
ア	(a)	(a)	(b)	(b)
イ	(a)	(a)	(c)	(b)
ウ	(a)	(b)	(b)	(b)
エ	(a)	(b)	(c)	(b)

問 17　通信販売を行っている B 社は，Web サイト（以下，B サイトという）にて商品情報を顧客に公開している。顧客が PC を操作して Web ブラウザから B サイトの商品一覧ページの URL にアクセスすると，DMZ 上の Web サーバは社内 LAN 上のデータベースサーバ（以下，DB サーバという）に対して，現在 B 社が扱っている各商品の名称，単価，在庫の有無などの情報を問い合わせる。DB サーバから返された情報を，Web サーバは整形して Web ページを生成し，Web ブラウザに送る。なお，B サイトの全てのページは HTTP over TLS（以下，HTTPS という）でアクセスされる。

B 社のネットワーク構成を以下の図 1 に示す。なお，インターネット上の DNS サーバが B 社のドメイン名に対応する IP アドレスを問い合わせるときは，DMZ 上の DNS サーバに対して DNS でアクセスする。



FW：ファイアウォール

図 1　B 社のネットワーク構成

図 1 の通信経路で使用されるプロトコル及び宛先ポート番号を表 1 に示す。また，図 1 のファイアウォールにおけるフィルタリング設定を表 2 に示す。

表 1　通信経路で使用されるプロトコル及び宛先ポート番号

通信経路	プロトコル	宛先ポート番号
顧客の PC から Web サーバ	HTTPS	443
インターネット上の DNS サーバから B 社の DNS サーバ	DNS	53
Web サーバから DB サーバ	DB 専用	1666

表２　ファイアウォールにおけるフィルタリング設定

項番	条件			動作
	送信元	宛先	宛先ポート番号	
1	任意	Web サーバ	443	許可
2	任意	DNS サーバ	53	許可
3	任意	DB サーバ	1666	許可
4	任意	任意	任意	拒否

注記　項番が小さい行から順に照合し，最初に条件が一致したものが適用される。

設問　表２のフィルタリング設定をB社の情報セキュリティ責任者が検証したところ,設定に誤りがあることが分かった。この状況で起こり得るセキュリティリスクとして最も適切なものを，解答群の中から選べ。

解答群

ア　暗号化されていないHTTPプロトコルでWebサーバにアクセスされてしまう。
イ　顧客のPCから直接B社のDBサーバにアクセスされてしまう。
ウ　顧客のPCから直接B社の社内LAN上のPCにアクセスされてしまう。
エ　フィルタリング設定の内容が勝手に変更されてしまう。

問18　システム開発を事業とするC社は，自社の情報セキュリティにおけるリスクを数値化して評価し，管理することになった。そのために，情報資産のリスク評価を行うためのC社独自の基準（以下，C基準という）を設定することにした。

C基準では，機密性，完全性及び可用性のそれぞれについて，情報資産のリスクの値（以下，リスク値という）を次の式で求める。

リスク値 ＝ 情報資産の価値 × 脅威 × 脆（ぜい）弱性

〔情報資産の価値を評価するための基準〕

C基準では，機密性，完全性及び可用性のそれぞれから，情報資産の価値を評価する。そのための評価基準と情報資産の価値を表す値を，表１～３のとおりに設定している。

表１　機密性の評価基準と値

評価基準	値
C社以外の者にも開示することができる。	1
C社の社内，もしくは特定の部門にしか開示できない。	2
必要最小限の関係者以外には開示できない。	3

表２　完全性の評価基準と値

評価基準	値
完全性が損なわれても業務への影響はないか，または非常に小さい。	1
完全性が損なわれても業務への影響は小さいか，中程度にとどまる。	2
完全性が損なわれると，業務への影響が大きい。	3

表３　可用性の評価基準と値

評価基準	値
情報資産が利用できない時間（以下，利用停止時間という）が年間で10時間を超えても問題はない。	1
利用停止時間が年間で10時間まで容認できる。	2
利用停止時間が年間で1時間まで容認できる。	3

〔脅威と脆弱性を判断するための基準〕

C基準では，脅威と脆弱性を判断するための基準と，脅威や脆弱性の大きさを表す値を，それぞれ表４，５のとおりに設定した。

表４　脅威の判断基準と値

判断基準	値
事故が発生する可能性が低い。	1
事故が発生する可能性が中程度である。	2
事故が発生する可能性が高い。	3

表５　脆弱性の判断基準と値

判断基準	値
適切な対策が複数なされている。	1
適切な対策が一つだけなされている。	2
適切な対策が全くなされていない。	3

C社では，データベースサーバ（以下，DBサーバという）上で管理されている調達先情報のデータベース（以下，調達先DBという）について，機密性，完全性及び可用性のそれぞれから見た情報資産の価値を評価するために，調達先に関する業務状況などを社内の各部門から聴取した。

(1) 調達先 DB に記録されている情報を用いて，郵便物の宛名ラベルを印字するなどしている。ただし，調達先 DB が利用できない場合，管理台帳や調達先の Web サイトから住所などを参照できるので，利用停止時間は何時間になっても業務に影響はない。
(2) システム開発業務において調達先 DB の情報を使用することは少ないため，誤りがあったとしても業務への影響は非常に小さい。
(3) 調達先の中には，この調達先と取引していることを同業他社や別の調達先に知られたくない業者も存在するので，調達先 DB の内容を社外には公開できない。

〔脅威と脆弱性の現状と受容可能なリスク〕

C 社は，DB サーバに関する脅威とそれに対する C 社の現在の対策状況を調査した。その結果を表６に示す。

表６　DB サーバに関する脅威と C 社の現在の対策状況

脅威		現在の対策状況
種類	値	
不正アクセス	3	・ファイアウォールによる外部からのアクセス遮断 ・DB サーバのプログラム改修による外部からの攻撃抑止
なりすまし	2	・パスワードの管理の徹底
ウイルス感染	3	・特になし（ウイルス対策ソフトを導入していない）

C 基準では，機密性，完全性及び可用性のいずれについてもリスク値が 15 以下であれば，そのリスクは受容可能とみなして追加の対策は行わず，そうでなければ追加の対策を行うとしている。表６に示す３種類の脅威が，機密性，完全性及び可用性のいずれについても影響し被害を及ぼすと仮定すると，調達先 DB の完全性に関して，“不正アクセス”のリスク値は［　a　］に，“なりすまし”のリスク値は［　b　］になり，どちらも受容可能とわかる。調達先 DB の機密性に関しては，“ウイルス感染”のリスク値が 18 になるので追加のリスク対策が必要になる。

設問　記述中の［　a　］，［　b　］に入れるリスク値の適切な組合せを，解答群の中から選べ。

解答群

	a	b
ア	3	4
イ	3	12
ウ	6	4
エ	6	12

試験問題は著作権法上の保護を受けています。
試験問題の一部または全部について，サーティファイから文書による許諾を得ずに，いかなる方法においても私的使用の範囲を超えて，無断で複写，複製することを禁じます。
無断複製，転載は損害賠償，著作権法の罰則の対象になることがあります。
©CERTIFY Inc.2023