基本情報技術者試験　科目 B
サンプル問題

試験時間	100分
問題番号	問 1 ～問20
選択方法	全問必須

擬似言語の記述形式（基本情報技術者試験用）

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。

〔擬似言語の記述形式〕

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***	
変数名 ← ***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	選択処理を示す。 ***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	前判定繰返し処理を示す。 ***条件式***が真の間，***処理***を繰返し実行する。 ***処理***は，0 以上の文の集まりである。
do 　***処理*** while（***条件式***）	後判定繰返し処理を示す。 ***処理***を実行し，***条件式***が真の間，***処理***を繰返し実行する。 ***処理***は，0 以上の文の集まりである。
for（***制御記述***） 　***処理*** endfor	繰返し処理を示す。 ***制御記述***の内容に基づいて，***処理***を繰返し実行する。 ***処理***は，0 以上の文の集まりである。

〔演算子と優先順位〕

演算子の種類		演算子	優先度
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。

〔論理型の定数〕

true, false

〔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，内側の“{”と“}”に囲まれた部分は，1 行分の内容を表す。

〔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，その変数は未定義になる。

問1　次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。

プログラムを実行すると，“［　　］”と出力される。

〔プログラム〕

```
整数型: x ← 1
整数型: y ← 2
整数型: z ← 3
x ← y
y ← z
z ← x
yの値 と zの値 をこの順にコンマ区切りで出力する
```

解答群

ア　1, 2	イ　1, 3	ウ　2, 1
エ　2, 3	オ　3, 1	カ　3, 2

問２　次のプログラム中の［　a　］～［　c　］に入れる正しい答えの組合せを，解答群の中から選べ。

関数 fizzBuzz は，引数で与えられた値が，3 で割り切れて 5 で割り切れない場合は“3 で割り切れる”を，5 で割り切れて 3 で割り切れない場合は“5 で割り切れる”を，3 と 5 で割り切れる場合は“3 と 5 で割り切れる”を返す。それ以外の場合は“3 でも 5 でも割り切れない”を返す。

〔プログラム〕

```
○文字列型: fizzBuzz(整数型: num)
  文字列型: result
  if (num が［　a　］で割り切れる)
    result ← ”［　a　］で割り切れる”
  elseif (num が［　b　］で割り切れる)
    result ← ”［　b　］で割り切れる”
  elseif (num が［　c　］で割り切れる)
    result ← ”［　c　］で割り切れる”
  else
    result ← ”3 でも 5 でも割り切れない”
  endif
  return result
```

解答群

	a	b	c
ア	3	3 と 5	5
イ	3	5	3 と 5
ウ	3 と 5	3	5
エ	5	3	3 と 5
オ	5	3 と 5	3

問３　次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。ここで，配列の要素番号は 1 から始まる。

関数 makeNewArray は，要素数 2 以上の整数型の配列を引数にとり，整数型の配列を返す関数である。関数 makeNewArray を makeNewArray({3, 2, 1, 6, 5, 4})として呼び出したとき，戻り値の配列の要素番号 5 の値は［　　］となる。

〔プログラム〕

```
○整数型の配列: makeNewArray(整数型の配列: in)
  整数型の配列: out ← {}  // 要素数0の配列
  整数型: i, tail
  outの末尾 に in[1]の値 を追加する
  for（i を 2 から inの要素数 まで 1 ずつ増やす）
    tail ← out[outの要素数]
    outの末尾 に（tail ＋ in[i]）の結果を追加する
  endfor
  return out
```

解答群

ア　5　　イ　6　　ウ　9　　エ　11　　オ　12

カ　17　　キ　21

問４　次のプログラム中の［　a　］～［　c　］に入れる正しい答えの組合せを，解答群の中から選べ。

関数 gcd は，引数で与えられた二つの正の整数 num1 と num2 の最大公約数を，次の(1)～(3)の性質を利用して求める。

(1)　num1 と num2 が等しいとき，num1 と num2 の最大公約数は num1 である。

(2)　num1 が num2 より大きいとき，num1 と num2 の最大公約数は，(num1 － num2) と num2 の最大公約数と等しい。

(3)　num2 が num1 より大きいとき，num1 と num2 の最大公約数は，(num2 － num1) と num1 の最大公約数と等しい。

〔プログラム〕

```
○整数型: gcd(整数型: num1, 整数型: num2)
  整数型: x ← num1
  整数型: y ← num2
  [ a ]
  if ( [ b ] )
    x ← x － y
  else
    y ← y － x
  endif
  [ c ]
return x
```

解答群

	a	b	c
ア	if (x ≠ y)	x ＜ y	endif
イ	if (x ≠ y)	x ＞ y	endif
ウ	while (x ≠ y)	x ＜ y	endwhile
エ	while (x ≠ y)	x ＞ y	endwhile

問５　次のプログラム中の［　　］に入れる正しい答えを，解答群の中から選べ。

関数 calc は，正の実数 x と y を受け取り，$\sqrt{x^2+y^2}$ の計算結果を返す。関数 calc が使う関数 pow は，第 1 引数として正の実数 a を，第 2 引数として実数 b を受け取り，a の b 乗の値を実数型で返す。

〔プログラム〕

```
○実数型: calc(実数型: x, 実数型: y)
  return ［　　］
```

解答群

ア　(pow(x, 2) ＋ pow(y, 2)) ÷ pow(2, 0.5)

イ　(pow(x, 2) ＋ pow(y, 2)) ÷ pow(x, y)

ウ　pow(2, pow(x, 0.5)) ＋ pow(2, pow(y, 0.5))

エ　pow(pow(pow(2, x), y), 0.5)

オ　pow(pow(x, 2) ＋ pow(y, 2), 0.5)

カ　pow(x, 2) × pow(y, 2) ÷ pow(x, y)

キ　pow(x, y) ÷ pow(2, 0.5)

問６　次のプログラム中の［　　］に入れる正しい答えを，解答群の中から選べ。

関数 rev は 8 ビット型の引数 byte を受け取り，ビットの並びを逆にした値を返す。例えば，関数 rev を rev(01001011) として呼び出すと，戻り値は 11010010 となる。

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，演算子 >> は論理右シフト，演算子 << は論理左シフトを表す。例えば，value >> n は value の値を n ビットだけ右に論理シフトし，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〕

```
○8 ビット型: rev(8 ビット型: byte)
  8 ビット型: rbyte ← byte
  8 ビット型: r ← 00000000
  整数型: i
  for (i を 1 から 8 まで 1 ずつ増やす)
    [　　]
  endfor
  return r
```

解答群

ア
```
r ← (r << 1) ∨ (rbyte ∧ 00000001)
rbyte ← rbyte >> 1
```

イ
```
r ← (r << 7) ∨ (rbyte ∧ 00000001)
rbyte ← rbyte >> 7
```

ウ
```
r ← (rbyte << 1) ∨ (rbyte >> 7)
rbyte ← r
```

エ
```
r ← (rbyte >> 1) ∨ (rbyte << 7)
rbyte ← r
```

問７　次のプログラム中の［　　］に入れる正しい答えを，解答群の中から選べ。

関数 factorial は非負の整数 n を引数にとり，その階乗を返す関数である。非負の整数 n の階乗は n が 0 のときに 1 になり，それ以外の場合は 1 から n までの整数を全て掛け合わせた数となる。

〔プログラム〕

```
○整数型: factorial(整数型: n)
  if (n ＝ 0)
    return 1
  endif
  return ［　　］
```

解答群

ア　(n － 1) × factorial(n)　　イ　factorial(n － 1)

ウ　n　　エ　n × (n － 1)

オ　n × factorial(1)　　カ　n × factorial(n － 1)

〔 メ モ 用 紙 〕

問８　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。

優先度付きキューを操作するプログラムである。優先度付きキューとは扱う要素に優先度を付けたキューであり，要素を取り出す際には優先度の高いものから順番に取り出される。クラス PrioQueue は優先度付きキューを表すクラスである。クラス PrioQueue の説明を図に示す。ここで，優先度は整数型の値 1，2，3 のいずれかであり，小さい値ほど優先度が高いものとする。

手続 prioSched を呼び出したとき，出力は［　　］の順となる。

コンストラクタ	説明
PrioQueue()	空の優先度付きキューを生成する。

メソッド	戻り値	説明
enqueue(文字列型: s, 整数型: prio)	なし	優先度付きキューに，文字列 s を要素として，優先度 prio で追加する。
dequeue()	文字列型	優先度付きキューからキュー内で最も優先度の高い要素を取り出して返す。最も優先度の高い要素が複数あるときは，そのうちの最初に追加された要素を一つ取り出して返す。
size()	整数型	優先度付きキューに格納されている要素の個数を返す。

図　クラス PrioQueue の説明

〔プログラム〕

```
○prioSched()
  PrioQueue: prioQueue ← PrioQueue()
  prioQueue.enqueue("A", 1)
  prioQueue.enqueue("B", 2)
  prioQueue.enqueue("C", 2)
  prioQueue.enqueue("D", 3)
  prioQueue.dequeue()  /* 戻り値は使用しない */
  prioQueue.dequeue()  /* 戻り値は使用しない */
  prioQueue.enqueue("D", 3)
  prioQueue.enqueue("B", 2)
  prioQueue.dequeue()  /* 戻り値は使用しない */
  prioQueue.dequeue()  /* 戻り値は使用しない */
  prioQueue.enqueue("C", 2)
  prioQueue.enqueue("A", 1)
  while (prioQueue.size() が 0 と等しくない)
    prioQueue.dequeue() の戻り値を出力
  endwhile
```

解答群

ア　“A”, “B”, “C”, “D”

イ　“A”, “B”, “D”, “D”

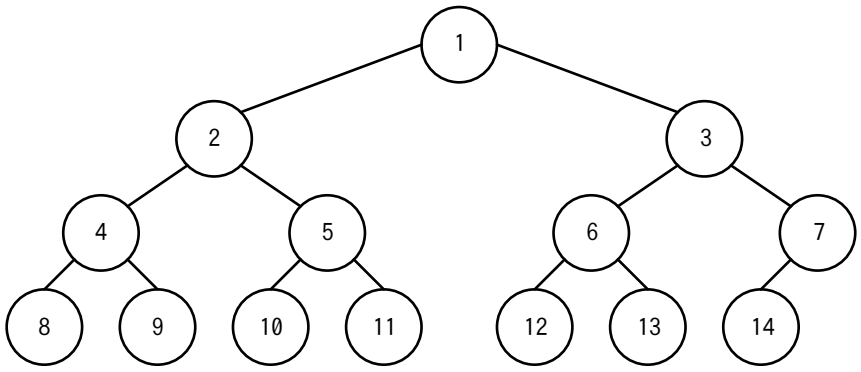
ウ　“A”, “C”, “C”, “D”

エ　“A”, “C”, “D”, “D”

問９　次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。ここで，配列の要素番号は 1 から始まる。

手続 order は，図の 2 分木の，引数で指定した節を根とする部分木をたどりながら，全ての節番号を出力する。大域の配列 tree が図の 2 分木を表している。配列 tree の要素は，対応する節の子の節番号を，左の子，右の子の順に格納した配列である。例えば，配列 tree の要素番号 1 の要素は，節番号 1 の子の節番号から成る配列であり，左の子の節番号 2，右の子の節番号 3 を配列 {2, 3} として格納する。

手続 order を order(1)として呼び出すと，［　　］の順に出力される。



注記 1　○の中の値は節番号である。
注記 2　子の節が一つの場合は，左の子の節とする。

図　プログラムが扱う 2 分木

〔プログラム〕

```
大域: 整数型配列の配列: tree ← {{2, 3}, {4, 5}, {6, 7}, {8, 9},
                                {10, 11}, {12, 13}, {14}, {}, {}, {},
                                {}, {}, {}, {}}  // {}は要素数0の配列

〇order(整数型: n)
  if (tree[n]の要素数 が 2 と等しい)
    order(tree[n][1])
    nを出力
    order(tree[n][2])
  elseif (tree[n]の要素数 が 1 と等しい)
    order(tree[n][1])
    nを出力
  else
    nを出力
  endif
```

解答群

ア　1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14

イ　1, 2, 4, 8, 9, 5, 10, 11, 3, 6, 12, 13, 7, 14

ウ　8, 4, 9, 2, 10, 5, 11, 1, 12, 6, 13, 3, 14, 7

エ　8, 9, 4, 10, 11, 5, 2, 12, 13, 6, 14, 7, 3, 1

問10　次のプログラム中の［　　］に入れる正しい答えを，解答群の中から選べ。

手続 delNode は，単方向リストから，引数 pos で指定された位置の要素を削除する手続である。引数 pos は，リストの要素数以下の正の整数とする。リストの先頭の位置を 1 とする。

クラス ListElement は，単方向リストの要素を表す。クラス ListElement のメンバ変数の説明を表に示す。ListElement 型の変数はクラス ListElement のインスタンスの参照を格納するものとする。大域変数 listHead には，リストの先頭要素の参照があらかじめ格納されている。

表　クラス ListElement のメンバ変数の説明

メンバ変数	型	説明
val	文字型	要素の値
next	ListElement	次の要素の参照 次の要素がないときの状態は未定義

〔プログラム〕

```
大域: ListElement: listHead  // リストの先頭要素が格納されている

○delNode(整数型: pos)  /* posは，リストの要素数以下の正の整数 */
  ListElement: prev
  整数型: i
  if (pos が 1 と等しい)
    listHead ← listHead.next
  else
    prev ← listHead
    /* posが2と等しいときは繰返し処理を実行しない */
    for (i を 2 から pos − 1 まで 1 ずつ増やす)
      prev ← prev.next
    endfor
    prev.next ← ［　　］
  endif
```

解答群

ア　listHead	イ　listHead.next
ウ　listHead.next.next	エ　prev
オ　prev.next	カ　prev.next.next

問11　次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。ここで，配列の要素番号は 1 から始まる。

関数 binSort を binSort(［　　］)として呼び出すと，戻り値の配列には未定義の要素は含まれておらず，値は昇順に並んでいる。

〔プログラム〕

```
○整数型の配列: binSort(整数型の配列: data)
  整数型: n ← dataの要素数
  整数型の配列: bins ← {n個の未定義の値}
  整数型: i

  for (i を 1 から n まで 1 ずつ増やす)
    bins[data[i]] ← data[i]
  endfor

  return bins
```

解答群

ア　{2, 6, 3, 1, 4, 5}　　イ　{3, 1, 4, 4, 5, 2}

ウ　{4, 2, 1, 5, 6, 2}　　エ　{5, 3, 4, 3, 2, 6}

〔 メ モ 用 紙 〕

問12　次のプログラム中の［　　］に入れる正しい答えを，解答群の中から選べ。ここで，配列の要素番号は 1 から始まる。

関数 simRatio は，引数として与えられた要素数 1 以上の二つの文字型の配列 s1 と s2 を比較し，要素数が等しい場合は，配列の並びがどの程度似ているかの指標として，（要素番号が同じ要素の文字同士が一致する要素の組みの個数 ÷ s1の要素数）を実数型で返す。例えば，配列の全ての要素が一致する場合の戻り値は 1，いずれの要素も一致しない場合の戻り値は 0 である。

なお，二つの配列の要素数が等しくない場合は，－1 を返す。

関数 simRatio に与える s1，s2 及び戻り値の例を表に示す。プログラムでは，配列の領域外を参照してはならないものとする。

表　関数 simRatio に与える s1，s2 及び戻り値の例

s1	s2	戻り値
{"a", "p", "p", "l", "e"}	{"a", "p", "p", "l", "e"}	1
{"a", "p", "p", "l", "e"}	{"a", "p", "r", "i", "l"}	0.4
{"a", "p", "p", "l", "e"}	{"m", "e", "l", "o", "n"}	0
{"a", "p", "p", "l", "e"}	{"p", "e", "n"}	－1

〔プログラム〕

```
○実数型: simRatio(文字型の配列: s1, 文字型の配列: s2)
  整数型: i, cnt ← 0
  if (s1の要素数 ≠ s2の要素数)
    return －1
  endif
  for (i を 1 から s1の要素数 まで 1 ずつ増やす)
    if (　　)
      cnt ← cnt ＋ 1
    endif
  endfor
  return cnt ÷ s1の要素数　/* 実数として計算する */
```

解答群

ア　s1[i] ≠ s2[cnt]　　イ　s1[i] ≠ s2[i]

ウ　s1[i] ＝ s2[cnt]　　エ　s1[i] ＝ s2[i]

問13　次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。ここで，配列の要素番号は 1 から始まる。

関数 search は，引数 data で指定された配列に，引数 target で指定された値が含まれていればその要素番号を返し，含まれていなければ －1 を返す。data は昇順に整列されており，値に重複はない。

関数 search には不具合がある。例えば，data の［　　］場合は，無限ループになる。

〔プログラム〕

```
○整数型: search(整数型の配列: data, 整数型: target)
  整数型: low, high, middle

  low ← 1
  high ← dataの要素数

  while (low ≦ high)
    middle ← (low ＋ high) ÷ 2 の商
    if (data[middle] ＜ target)
      low ← middle
    elseif (data[middle] ＞ target)
      high ← middle
    else
      return middle
    endif
  endwhile

  return －1
```

解答群

ア　要素数が 1 で，target がその要素の値と等しい

イ　要素数が 2 で，target が data の先頭要素の値と等しい

ウ　要素数が 2 で，target が data の末尾要素の値と等しい

エ　要素に－1 が含まれている

問14　次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。ここで，配列の要素番号は 1 から始まる。

要素数が 1 以上で，昇順に整列済みの配列を基に，配列を特徴づける五つの値を返すプログラムである。

関数 summarize を summarize({0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1})として呼び出すと，戻り値は［　　］である。

〔プログラム〕

```
○実数型: findRank(実数型の配列: sortedData, 実数型: p)
  整数型: i
  i ← (p × (sortedDataの要素数 − 1)) の小数点以下を切り上げた値
  return sortedData[i + 1]

○実数型の配列: summarize(実数型の配列: sortedData)
  実数型の配列: rankData ← {}  /* 要素数0の配列 */
  実数型の配列: p ← {0, 0.25, 0.5, 0.75, 1}
  整数型: i
  for (i を 1 から pの要素数 まで 1 ずつ増やす)
    rankDataの末尾 に findRank(sortedData, p[i])の戻り値 を追加する
  endfor
  return rankData
```

解答群

ア　{0.1, 0.3, 0.5, 0.7, 1}

イ　{0.1, 0.3, 0.5, 0.8, 1}

ウ　{0.1, 0.3, 0.6, 0.7, 1}

エ　{0.1, 0.3, 0.6, 0.8, 1}

オ　{0.1, 0.4, 0.5, 0.7, 1}

カ　{0.1, 0.4, 0.5, 0.8, 1}

キ　{0.1, 0.4, 0.6, 0.7, 1}

ク　{0.1, 0.4, 0.6, 0.8, 1}

問15　次の記述中の［　a　］と［　b　］に入れる正しい答えの組合せを，解答群の中から選べ。

三目並べにおいて自分が勝利する可能性が最も高い手を決定する。次の手順で，ゲームの状態遷移を木構造として表現し，根以外の各節の評価値を求める。その結果，根の子の中で最も評価値が高い手を，最も勝利する可能性が高い手とする。自分が選択した手を○で表し，相手が選択した手を×で表す。

〔手順〕

(1)　現在の盤面の状態を根とし，勝敗がつくか，引き分けとなるまでの考えられる全ての手を木構造で表現する。

(2)　葉の状態を次のように評価する。

①　自分が勝ちの場合は 10

②　自分が負けの場合は－10

③　引き分けの場合は 0

(3)　葉以外の節の評価値は，その節の全ての子の評価値を基に決定する。

①　自分の手番の節である場合，子の評価値で最大の評価値を節の評価値とする。

②　相手の手番の節である場合，子の評価値で最小の評価値を節の評価値とする。

ゲームが図の最上部にある根の状態のとき，自分が選択できる手は三つある。そのうち A が指す子の評価値は［　a　］であり，B が指す子の評価値は［　b　］である。

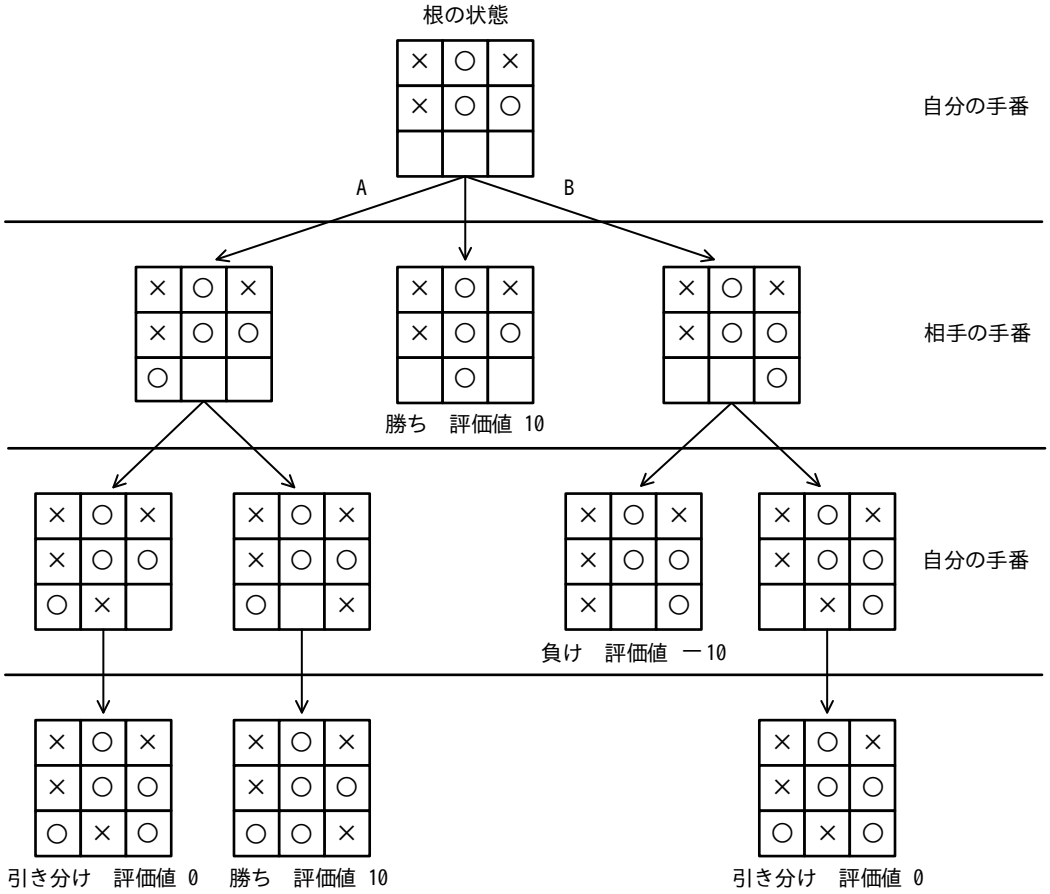


図　三目並べの状態遷移

解答群

	a	b
ア	0	－10
イ	0	0
ウ	10	－10
エ	10	0

問16　次のプログラム中の［　　］に入れる正しい答えを，解答群の中から選べ。二つの［　　］には，同じ答えが入る。ここで，配列の要素番号は 1 から始まる。

Unicode の符号位置を，UTF-8 の符号に変換するプログラムである。本問で数値の後ろに“(16)”と記載した場合は，その数値が 16 進数であることを表す。

Unicode の各文字には，符号位置と呼ばれる整数値が与えられている。UTF-8 は，Unicode の文字を符号化する方式の一つであり，符号位置が 800(16) 以上 FFFF(16) 以下の文字は，次のように 3 バイトの値に符号化する。

3 バイトの長さのビットパターンを 1110xxxx 10xxxxxx 10xxxxxx とする。ビットパターンの下線の付いた“x”の箇所に，符号位置を 2 進数で表した値を右詰めで格納し，余った“x”の箇所に，0 を格納する。この 3 バイトの値が UTF-8 の符号である。

例えば，ひらがなの“あ”の符号位置である 3042(16) を 2 進数で表すと 11000001000010 である。これを，上に示したビットパターンの“x”の箇所に右詰めで格納すると，1110xx11 10000001 10000010 となる。余った二つの“x”の箇所に 0 を格納すると，“あ”の UTF-8 の符号 11100011 10000001 10000010 が得られる。

関数 encode は，引数で渡された Unicode の符号位置を UTF-8 の符号に変換し，先頭から順に 1 バイトずつ要素に格納した整数型の配列を返す。encode には，引数として，800(16) 以上 FFFF(16) 以下の整数値だけが渡されるものとする。

〔プログラム〕

```
○整数型の配列: encode(整数型: codePoint)
  /* utf8Bytesの初期値は，ビットパターンの“x”を全て0に置き換え，
     8桁ごとに区切って，それぞれを2進数とみなしたときの値 */
  整数型の配列: utf8Bytes ← {224, 128, 128}
  整数型: cp ← codePoint
  整数型: i
  for (i を utf8Bytesの要素数 から 1 まで 1 ずつ減らす)
    utf8Bytes[i] ← utf8Bytes[i] ＋ (cp ÷ ［　　］ の余り)
    cp ← cp ÷ ［　　］ の商
  endfor
  return utf8Bytes
```

解答群

ア ((4 − i) × 2)	イ (2 の (4 − i)乗)	ウ (2 の i 乗)
エ (i × 2)	オ 2	カ 6
キ 16	ク 64	ケ 256

問17　製造業のA社では，ECサイト（以下，A社のECサイトをAサイトという）を使用し，個人向けの製品販売を行っている。A サイトは，A 社の製品やサービスが検索可能で，ログイン機能を有しており，あらかじめ A サイトに利用登録した個人（以下，会員という）の氏名やメールアドレスといった情報（以下，会員情報という）を管理している。Aサイトは，B社のPaaSで稼働しており，PaaS上のDBMSとアプリケーションサーバを利用している。

A社は，Aサイトの開発，運用をC社に委託している。A社とC社との間の委託契約では，Web アプリケーションプログラムの脆弱性対策は，C 社が実施するとしている。

最近，A社の同業他社が運営しているWebサイトで脆弱性が悪用され，個人情報が漏えいするという事件が発生した。そこで A 社は，セキュリティ診断サービスを行っているD社に，Aサイトの脆弱性診断を依頼した。脆弱性診断の結果，対策が必要なセキュリティ上の脆弱性が複数指摘された。図1にD社からの指摘事項を示す。

項番 1　A サイトで利用しているアプリケーションサーバの OS に既知の脆弱性があり，脆弱性を悪用した攻撃を受けるおそれがある。
項番 2　A サイトにクロスサイトスクリプティングの脆弱性があり，会員情報を不正に取得されるおそれがある。
項番 3　A サイトで利用している DBMS に既知の脆弱性があり，脆弱性を悪用した攻撃を受けるおそれがある。

図1　D社からの指摘事項

設問　図 1 中の各項番それぞれに対処する組織の適切な組合せを，解答群の中から選べ。

解答群

	項番 1	項番 2	項番 3
ア	A 社	A 社	A 社
イ	A 社	A 社	C 社
ウ	A 社	B 社	B 社
エ	B 社	B 社	B 社
オ	B 社	B 社	C 社
カ	B 社	C 社	B 社
キ	B 社	C 社	C 社
ク	C 社	B 社	B 社
ケ	C 社	B 社	C 社
コ	C 社	C 社	B 社

問18　A 社は IT 開発を行っている従業員 1,000 名の企業である。総務部 50 名，営業部 50 名で，ほかは開発部に所属している。開発部員の 9 割は客先に常駐している。現在，A 社における PC の利用状況は図 1 のとおりである。

1　A 社の PC
・総務部員，営業部員及び A 社オフィスに勤務する開発部員には，会社が用意した PC（以下，A 社 PC という）を一人 1 台ずつ貸与している。
・客先常駐開発部員には，A 社 PC を貸与していないが，代わりに客先常駐開発部員が A 社オフィスに出社したときに利用するための共用 PC を用意している。
2　客先常駐開発部員の業務システム利用
・客先常駐開発部員が休暇申請，経費精算などで業務システムを利用するためには共用 PC を使う必要がある。
3　A 社の VPN 利用
・A 社には，VPN サーバが設置されており，営業部員が出張時に A 社 PC からインターネット経由で社内ネットワークに VPN 接続し，業務システムを利用できるようになっている。規則で，VPN 接続には A 社 PC を利用すると定められている。

図 1　A 社における PC の利用状況

A 社では，客先常駐開発部員が業務システムを使うためだけに A 社オフィスに出社するのは非効率的であると考え，客先常駐開発部員に対して個人所有 PC の業務利用（BYOD）と VPN 接続の許可を検討することにした。

設問　客先常駐開発部員に，個人所有 PC からの VPN 接続を許可した場合に，増加する又は新たに生じると考えられるリスクを二つ挙げた組合せは，次のうちどれか。解答群のうち，最も適切なものを選べ。

（一） VPN 接続が増加し，可用性が損なわれるリスク

（二） 客先常駐開発部員が A 社 PC を紛失するリスク

（三） 客先常駐開発部員がフィッシングメールの URL をクリックして個人所有 PC がマルウェアに感染するリスク

（四） 総務部員が個人所有 PC を VPN 接続するリスク

（五） マルウェアに感染した個人所有 PC が社内ネットワークに VPN 接続され，マルウェアが社内ネットワークに拡散するリスク

解答群

ア （一），（二）	イ （一），（三）	ウ （一），（四）
エ （一），（五）	オ （二），（三）	カ （二），（四）
キ （二），（五）	ク （三），（四）	ケ （三），（五）
コ （四），（五）		

問19　A 社は従業員 200 名の通信販売業者である。一般消費者向けに生活雑貨，ギフト商品などの販売を手掛けている。取扱商品の一つである商品 Z は，Z 販売課が担当している。

〔Z 販売課の業務〕

現在，Z 販売課の要員は，商品 Z についての受注管理業務及び問合せ対応業務を行っている。商品 Z についての受注管理業務の手順を図 1 に示す。

商品 Z の顧客からの注文は電子メールで届く。 (1) 入力 販売担当者は，届いた注文（変更，キャンセルを含む）の内容を受注管理システム [1]（以下，J システムという）に入力し，販売責任者 [2] に承認を依頼する。 (2) 承認 販売責任者は，注文の内容と J システムへの入力結果を突き合わせて確認し，問題がなければ承認する。問題があれば差し戻す。

注 [1]　A 社情報システム部が運用している。利用者は，販売責任者，販売担当者などである。
注 [2]　Z 販売課の課長 1 名だけである。

図 1　受注管理業務の手順

〔J システムの操作権限〕

Z 販売課では，J システムについて，次の利用方針を定めている。

［方針 1］　ある利用者が入力した情報は，別の利用者が承認する。

［方針 2］　販売責任者は，Z 販売課の全業務の情報を閲覧できる。

J システムでは，業務上必要な操作権限を利用者に与える機能が実装されている。

この度，商品 Z の受注管理業務が受注増によって増えていることから，B 社に一部を委託することにした（以下，商品 Z の受注管理業務の入力作業を行う B 社従業員を商品 Z の B 社販売担当者といい，商品 Z の B 社販売担当者の入力結果を閲覧して，不備があれば A 社に口頭で差戻しを依頼する B 社従業員を商品 Z の B 社販売責任者という）。

委託に当たって，Z 販売課は情報システム部に J システムに関する次の要求事項を伝えた。

［要求 1］　B 社が入力した場合は，A 社が承認する。

［要求 2］　A 社の販売担当者が入力した場合は，現状どおりに A 社の販売責任者が承認する。

上記を踏まえ，情報システム部は今後の各利用者に付与される操作権限を表 1 にまとめ，Z 販売課の情報セキュリティリーダーである C さんに確認をしてもらった。

表 1　操作権限案

付与される操作権限 / 利用者	J システム		
	閲覧	入力	承認
（省略）	○		○
Z 販売課の販売担当者	（省略）	（省略）	（省略）
a1	○		
a2	○	○	

注記　○は，操作権限が付与されることを示す。

設問　表 1 中の［ a1 ］，［ a2 ］に入れる字句の適切な組合せを，a に関する解答群の中から選べ。

a に関する解答群

	a1	a2
ア	Z 販売課の販売責任者	商品 Z の B 社販売責任者
イ	Z 販売課の販売責任者	商品 Z の B 社販売担当者
ウ	商品 Z の B 社販売責任者	Z 販売課の販売責任者
エ	商品 Z の B 社販売責任者	商品 Z の B 社販売担当者
オ	商品 Z の B 社販売担当者	商品 Z の B 社販売責任者

問20　A 社は栄養補助食品を扱う従業員 500 名の企業である。A 社のサーバ及びファイアウォール（以下，FW という）を含む情報システムの運用は情報システム部が担当している。

ある日，内部監査部の監査があり，FW の運用状況について情報システム部の B 部長が図 1 のとおり説明したところ，表 1 に示す指摘を受けた。

・FW を含め，情報システムの運用は，情報システム部の運用チームに所属する 6 名の運用担当者が担当している。
・FW の運用には，FW ルールの編集，操作ログの確認，並びに編集後の FW ルールの確認及び操作の承認（以下，編集後の FW ルールの確認及び操作の承認を操作承認という）の三つがある。
・FW ルールの編集は事前に作成された操作指示書に従って行う。
・FW の機能には，FW ルールの編集，操作ログの確認，及び操作承認の三つがある。
・FW ルールの変更には，FW ルールの編集と操作承認の両方が必要である。操作承認の前に操作ログの確認を行う。
・FW の利用者 ID は各運用担当者に個別に発行されており，利用者 ID の共用はしていない。
・FW では，機能を利用する権限を運用担当者の利用者 ID ごとに付与できる。
・現在は，6 名の運用担当者とも全権限を付与されており，運用担当者は FW のルールの編集後，編集を行った運用担当者が操作に誤りがないことを確認し，操作承認をしている。
・FW へのログインにはパスワードを利用している。パスワードは 8 文字の英数字である。
・FW の運用では，運用担当者の利用者 ID ごとに，ネットワークを経由せずコンソールでログインできるかどうか，ネットワークを経由してリモートからログインできるかどうかを設定できる。
・FW は，ネットワークを経由せずコンソールでログインした場合でも，ネットワークを経由してリモートからログインした場合でも，同一の機能を利用できる。
・FW はサーバルームに設置されており，サーバルームにはほかに数種類のサーバも設置されている。
・運用担当者だけがサーバルームへの入退室を許可されている。

図 1　FW の運用状況

表 1　内部監査部からの指摘

指摘	指摘内容
指摘 1	FW の運用の作業の中で，職務が適切に分離されていない。
指摘 2	（省略）
指摘 3	（省略）
指摘 4	（省略）

B 部長は表 1 の指摘に対する改善策を検討することにした。

設問　表 1 中の指摘 1 について，FW ルールの誤った変更を防ぐための改善策はどれか。解答群のうち，最も適切なものを選べ。

解答群

ア　Endpoint Detection and Response（EDR）をコンソールに導入し，監視を強化する。

イ　FW での運用担当者のログインにはパスワード認証の代わりに多要素認証を導入する。

ウ　FW のアクセス制御機能を使って，運用担当者をコンソールからログインできる者，リモートからログインできる者に分ける。

エ　FW の運用担当者を 1 人に限定する。

オ　運用担当者の一部を操作ログの確認だけをする者とし，それらの者には操作ログの確認権限だけを付与する。

カ　運用担当者を，FW ルールの編集を行う者，操作ログを確認し，操作承認をする者に分け，それぞれに必要最小限の権限を付与する。

キ　作業を行う運用担当者を，曜日ごとに割り当てる。

〔 メ モ 用 紙 〕

〔 メ モ 用 紙 〕

〔 メ モ 用 紙 〕

〔 メ モ 用 紙 〕

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。