文部科学省後援

情報システム試験

令和７年度前期 情報検定

＜実施　令和7年9月14日（日）＞

プログラミングスキル

（説明時間　10：00～10：10）

（試験時間　10：10～11：40）

・試験問題は試験開始の合図があるまで開かないでください。

・解答用紙（マークシート）への必要事項の記入は，試験開始の合図と同時に行いますので，それまで伏せておいてください。

・試験開始の合図の後，次のページを開いてください。**＜受験上の注意＞**が記載されています。必ず目を通してから解答を始めてください。

・試験問題は，すべてマークシート方式です。正解と思われるものを１つ選び，解答欄の()をＨＢの黒鉛筆でぬりつぶしてください。２つ以上ぬりつぶすと，不正解になります。

・辞書，参考書類の使用および筆記用具の貸し借りは一切禁止です。

・電卓の使用が認められます。ただし，下記の機種については使用が認められません。

＜使用を認めない電卓＞

１．電池式（太陽電池を含む）以外の電卓

２．文字表示領域が複数行ある電卓（計算状態表示の一行は含まない）

３．プログラムを組み込む機能がある電卓

４．電卓が主たる機能ではないもの

＊パソコン（電子メール専用機等を含む），携帯電話，スマートフォン，タブレット，電子手帳，電子メモ，電子辞書，翻訳機能付き電卓，音声応答のある電卓，電卓付き腕時計，時計型ウェアラブル端末等

５．その他試験監督者が不適切と認めるもの

一般財団法人　職業教育・キャリア教育財団

＜受験上の注意＞

１．この試験問題は18ページあります。ページ数を確認してください。
乱丁等がある場合は，手をあげて試験監督者に合図してください。
※問題を読みやすくするために空白ページを設けている場合があります。

２．解答用紙（マークシート）に，受験者氏名・受験番号を記入し，受験番号下欄の数字をぬりつぶしてください。正しく記入されていない場合は，採点されませんので十分注意してください。

３．試験問題についての質問には，一切答えられません。自分で判断して解答してください。

４．試験中の筆記用具の貸し借りは一切禁止します。

５．試験を開始してから30分以内は途中退出できません。30分経過後退出する場合は，もう一度，受験番号・マーク・氏名が記載されているか確認して退出してください。なお，試験終了５分前の合図以降は退出できません。試験問題は各自お持ち帰りください。

６．試験後の合否結果（合否通知），および合格者への「合格証・認定証」はすべて，Ｗｅｂ認証で行います。
①情報検定（J検）Ｗｅｂサイト合否結果検索ページ及びモバイル合否検索サイト上で，ディジタル「合否通知」，ディジタル「合格証・認定証」が交付されます。
②団体宛には合否結果一覧ほか，試験結果資料一式を送付します。
③合否等の結果についての電話・手紙等でのお問い合わせには，一切応じられませんので，ご了承ください。

問題１　次のデータ構造に関する記述を読み，各設問に答えよ。

配列を用いたデータ管理で，ある配列の要素が他の配列の添字を示す参照配列がある。参照配列はデータの間接的な参照や動的なデータ操作において重要な役割を果たしている。

以下に３つの一次元配列である map1，map2，values がある。
一次元配列 map1 に図１のデータが格納されている。

0	1	2	3	4	5	6	7	8	9
7	3	0	6	1	4	8	2	5	9

図１　配列 map1

一次元配列 map2 に図２のデータが格納されている。

0	1	2	3	4	5	6	7	8	9
6	9	4	7	2	8	0	3	1	5

図２　配列 map2

一次元配列 values に図３のデータが格納されている。

0	1	2	3	4	5	6	7	8	9
100	200	300	400	500	600	700	800	900	1000

図３　配列 values

ここで，変数 i は map1 の添字を意味する。i の値が０のとき，map1[i]の値は７であり，map2 の添字を指し示す。map2[7]の値は３であり，values の添字を指し示す。よって，values[3]の値である 400 が導き出される。

＜設問１＞　次の参照配列に関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。

i の値が４のとき，map2[　(1)　]を経由し，values の値は　(2)　が導き出される。また，values の値である 700 を導き出したい場合，逆にたどっていくと map2[　(3)　]を経由し，i の値を　(4)　にすることで求めることができる。このような二段階の参照は中間にあるデータを変更するだけで最終的にアクセスするデータなどを簡単に切り替えられるメリットがある。

(1)，(3)，(4) の解答群

ア．0	イ．1	ウ．2	エ．3
オ．4	カ．5	キ．6	ク．7
ケ．8	コ．9		

(2) の解答群

ア．100　　イ．200　　ウ．300　　エ．400
オ．500　　カ．600　　キ．700　　ク．800
ケ．900　　コ．1000

＜設問２＞　次の参照配列のネストに関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。

参照配列はネスト(入れ子)にすることで，短く表現することができる。例えば，map1はmap1[i]と表現し，map2までの表現は［(5)］となる。次にvaluesまでの表現は［(6)］となる。

(5) の解答群

ア．map1[i]　　イ．map1[map2[i]]
ウ．map2[i]　　エ．map2[map1[i]]

(6) の解答群

ア．map1[map2[values[i]]]　　イ．map2[map1[values[i]]]
ウ．values[map1[map2[i]]]　　エ．values[map2[map1[i]]]

問題２　次の流れ図の説明を読み，流れ図中の［　　］に入れるべき適切な字句を解答群から選べ。

［流れ図の説明］

二つの自然数ａとｂについて，ａをｂで割ったときの余りをｒとすると，「ａとｂの最大公約数」は，「ｂとｒの最大公約数」に等しいという定理がある。

この定理を使って，二つの自然数の最大公約数を求める方法がユークリッドの互除法であり，その手順を次に示す。

［ユークリッドの互除法の手順］

① 二つの自然数ａとｂの大きい方をｘに，小さい方をｙに代入する。

② 「余りを求める手順」により求めたｘ÷ｙの余りをｒに代入する。

③ ｒ＝0ならばｙが最大公約数となり手順を終了する。
ｒ≠0ならばｙをｘに，ｒをｙに代入して②へ戻る。

［余りを求める手順］

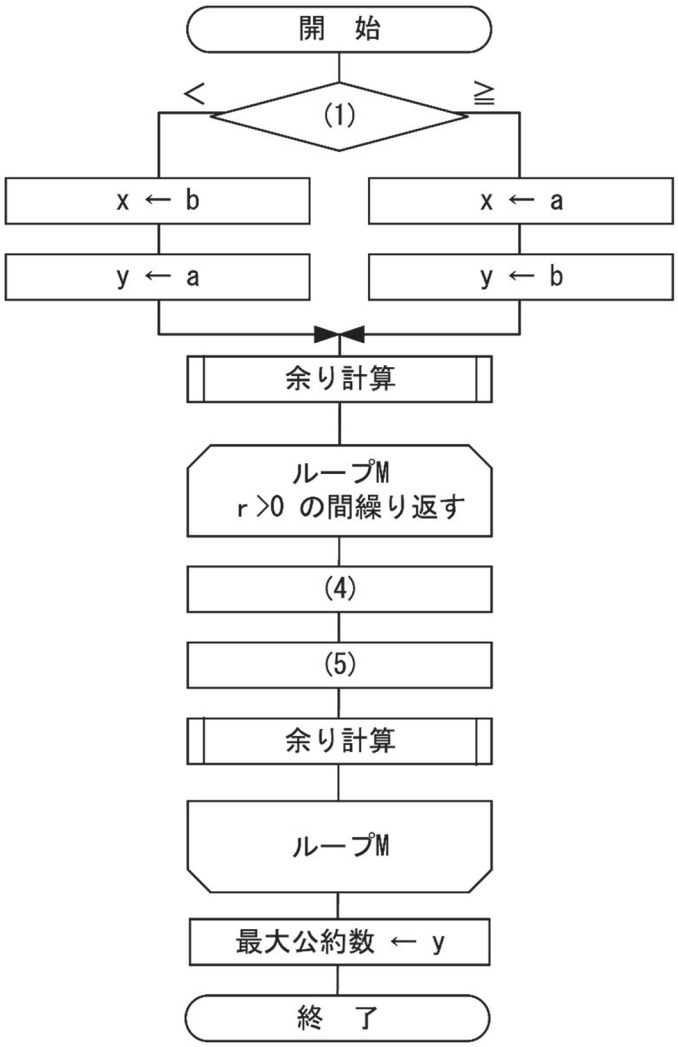
④ ｘをｒに代入する。

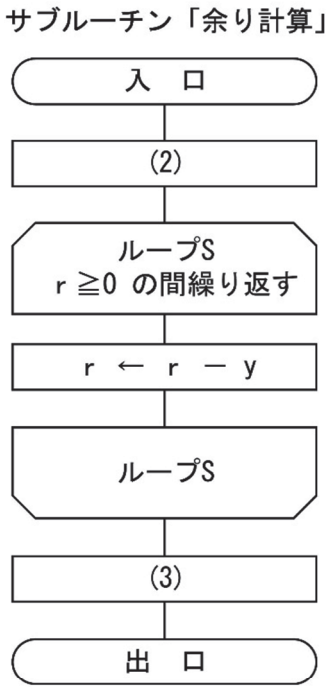
⑤ ｒ－ｙをｒに代入する。

⑥ ｒが負数になるまで⑤を繰り返す。

⑦ 引きすぎた分を戻すため，ｒ＋ｙをｒに代入する。この時のｒが余りである。

［流れ図］





(1) の解答群

ア．a ： b　　イ．b ： a

ウ．x ： y　　エ．y ： x

(2), (3) の解答群

ア．r ← x　　イ．r ← y

ウ．r ← r ＋ x　　エ．r ← r ＋ y

(4), (5) の解答群

ア．x ← r　　イ．x ← y

ウ．y ← r　　エ．y ← r ＋ x

問題３　次の相関関係に関する各設問に答えよ。なお，配列の添字は１から始まる。

＜設問１＞　次の相関係数に関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。

相関関係とは２つの変数の間に何らかの関係があることを意味するもので，一方の値が変化すると，もう一方の値も特定の傾向で変化する場合に，相関関係があるという。相関関係があるかどうかは，図１のような［　(1)　］を作成して視覚的に判断する方法と，計算によって判断する方法がある。

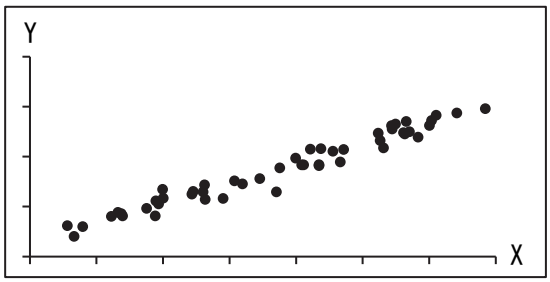


図１　相関関係のあるデータのグラフ

図１のプロットされた点のパターンから相関関係の種類や強さを把握できる。Ｘ軸が増えると，Ｙ軸も増える場合は正の相関があると判断でき，Ｘ軸が増えるとＹ軸が減る場合は負の相関があると判断できる。また，点の集まりが直線に近い場合は［　(2)　］と判断できる。

相関関係の強さの指標として使われるのが，相関係数である。相関係数（ｒ）は次の式で求めることができ－1 ～ ＋1 に収まる。

$$r = \frac{\frac{1}{n}\Sigma(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\frac{1}{n}\Sigma(x_i - \overline{x})^2}\ \sqrt{\frac{1}{n}\Sigma(y_i - \overline{y})^2}}$$

x_i は個々のデータを，$\overline{x}$ は平均を，Σは総和を表すものであり，分子の式は共分散，分母の式はxの標準偏差とyの標準偏差を掛けたものである。

ここで，５つの点(2，2)，(3，3)，(4，4)，(5，6)，(6，5) が与えられた時の相関係数を求める。図１と同様のグラフを描くと図２のとおりであり，直線に近いことがわかる。

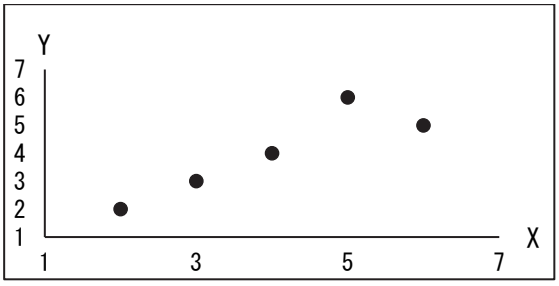


図２　５つの点のグラフ

xおよびyの平均はそれぞれ４である。xとyの偏差（個々の値から平均を引いた値）は次のようになる。

xの偏差：[−2, −1, 0, 1, 2]

yの偏差：[−2, −1, 0, 2, 1]

共分散は，同じ要素位置にある偏差（−2, −2），（−1, −1），（0, 0），（1, 2），（2, 1）同士を掛けた値の平均なので［ (3) ］である。

xの標準偏差はxの偏差の２乗の平均の平方根，yの標準偏差はyの偏差の２乗の平均の平方根なので，どちらも$\sqrt{2}$ である。よって，相関係数は［ (4) ］となる。

この値が［ (5) ］に近いほど相関は弱いと判断される。

(1)，(2) の解答群

ア．散布図　　イ．相関が無い

ウ．強い相関がある　　エ．ヒストグラム

オ．マップ　　カ．弱い相関がある

(3)，(4) の解答群

ア．0.7　　イ．0.8　　ウ．0.9　　エ．1.0

オ．1.2　　カ．1.4　　キ．1.6　　ク．1.8

(5) の解答群

ア．−1　　イ．0　　ウ．1　　エ．絶対値が 1

［相関係数を出力する流れ図］

相関係数を求める流れ図を次に示す。流れ図中の関数 cov は共分散を求める関数であり，関数 dev は標準偏差を求める関数である。

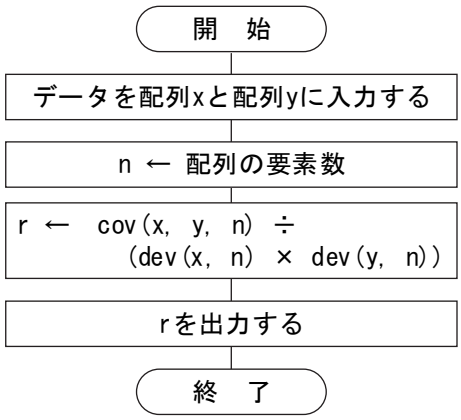


図３　相関係数を求める流れ図

＜設問２＞　次の平均値を返す流れ図中の［　　］に入れるべき適切な字句を解答群から選べ。

関数 cov と関数 dev では，平均値を求める必要がある。次の流れ図は，引数で受け取った配列に格納された値の平均値を返す関数 avg の流れ図である。関数 avg は配列データを配列 ary に，その要素数を変数 n に受け取る。

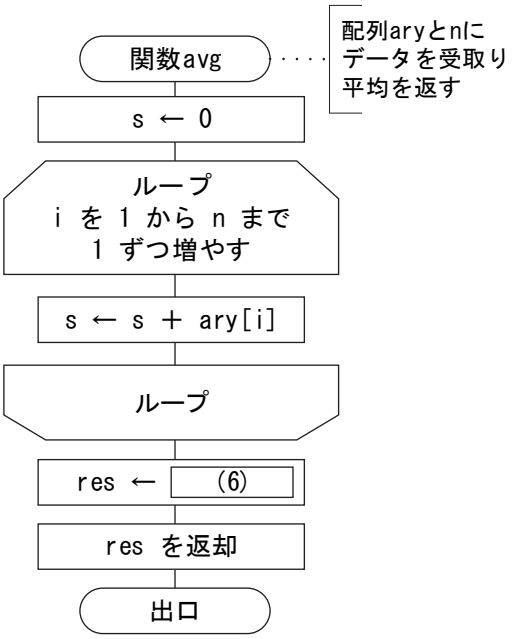


図４　関数 avg の流れ図

(6) の解答群

ア．s　　イ．s の平方根

ウ．s ÷ n　　エ．(s ÷ n)の平方根

＜設問３＞　次の共分散を返す流れ図中の［　　］に入れるべき適切な字句を解答群から選べ。

引数で受け取った配列に格納された値の共分散 $\frac{1}{n}\sum(x_i - \overline{x})(y_i - \overline{y})$ を計算する関数 cov の流れ図である。関数 cov は２つの配列データを配列 x および y に，その要素数を変数 n に受け取る。なお，この流れ図では図４の関数 avg を使用している。

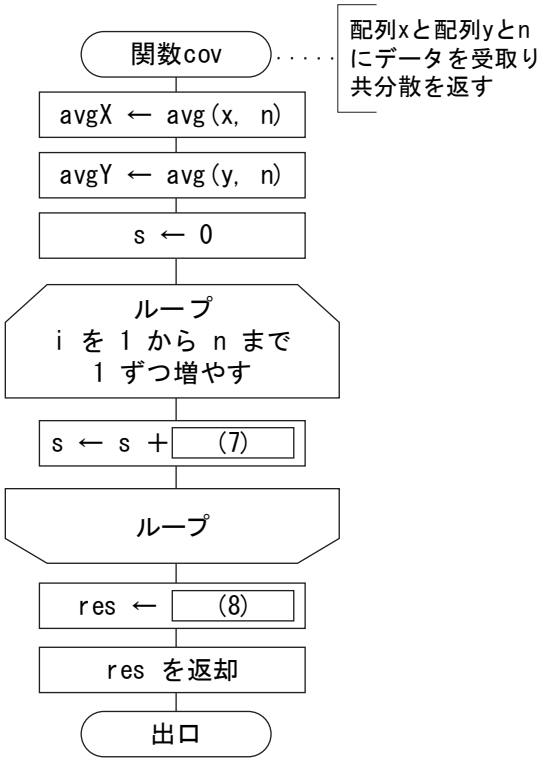


図５　関数 cov の流れ図

(7) の解答群

ア．　x[i] － avgX　　イ．y[i] － avgY

ウ．(ary[i] － a) ÷ n　　エ．(x[i] － avgX) × (y[i] － avgY)

(8) の解答群

ア．s　　イ．s の平方根

ウ．s ÷ n　　エ．(s ÷ n)の平方根

＜設問４＞　次の標準偏差を表示する流れ図中の［　　］に入れるべき適切な字句を解答群から選べ。

引数で受け取った配列に格納された値の標準偏差 $\sqrt{\frac{1}{n}\sum(ary - \overline{ary})^2}$ を計算する関数 dev の流れ図である。関数 dev はデータを配列 ary に，その要素数を変数 n に受け取る。なお，この流れ図中では図４の関数 avg を使用している。

［流れ図］

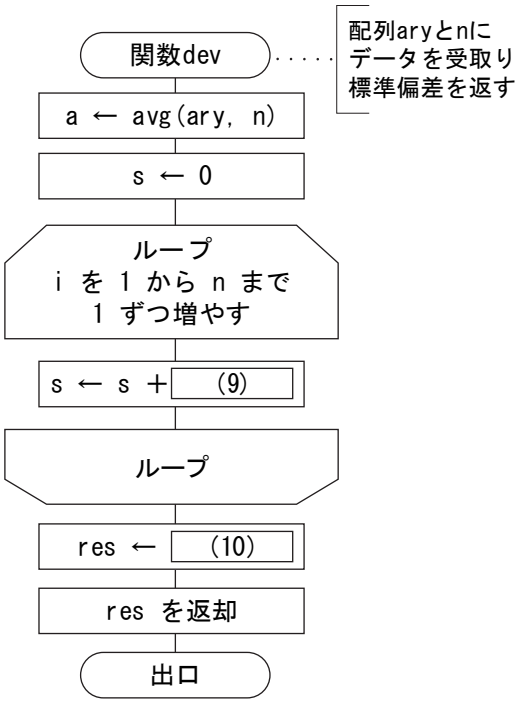


図６　関数 dev の流れ図

(9) の解答群

ア．ary[i] － a　　イ．ary[i] × ary[i]

ウ．(ary[i] － a) ÷ n　　エ．(ary[i] － a) × (ary[i] － a)

(10) の解答群

ア．s　　イ．s の平方根

ウ．s ÷ n　　エ．(s ÷ n)の平方根

問題４，問題５で使用する擬似言語の仕様は以下のとおりである。

［擬似言語の記述形式の説明］

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*	注釈を記述する。
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。

問題４　次のプログラムの説明を読み，各設問に答えよ。

コンピュータでは，データを昇順や降順に整列してから使用することが多い。整列されているデータは，重複を簡単に検出し取り除くことにより，一貫性と整合性を保ちやすく，効率が向上する。そのため，データを整列するにはいろいろなアルゴリズムがある。

＜設問１＞　次のバケットソートに関する説明を読み，プログラム中の［　　］に入れるべき適切な字句を解答群から選べ。

［プログラムの説明］

配列 DAT[i](i＝0,1,･･･,N-1)に 0～MAX-1 の範囲の整数値がランダムに格納されている。この配列 DAT の内容を，次の手順により昇順に整列する。なお，整数値は，ある値が複数存在することや，一つも存在しないこともある。

手順１：配列 DAT の内容を順にチェックし，配列 CNT[k](k＝0,1,･･･,MAX-1)の整数値を添字とする位置に，その整数値の出現回数をカウントする。例えば，配列 DAT 中に 2 が 3 個存在する場合には，CNT[2]＝3 となる。

手順２：配列 CNT の内容を順にチェックし，CNT[k]が 0 でないとき，配列 DAT に k を CNT[k]個連続して格納する。例えば，配列 CNT[0]＝2，CNT[1]＝0，CNT[2]＝3，CNT[3]＝2 の場合，配列 DAT は次のように昇順に整列された値が格納される。

	0	1	2	3	4	5	6	…
DAT	0	0	2	2	2	3	3	…

図１　手順２実行後の配列 DAT の内容

［プログラム］

```
○Bucket_sort (整数型: DAT[], 整数型: N, 整数型: MAX )
整数型: CNT[MAX], i, j, k
/* 配列 CNT の全要素を 0 クリアする */
for (k を [ (1) ] の間 1 ずつ増やす)
    CNT[k] ← 0
endfor
/* 配列 DAT の各整数値の個数を求める */
for (i を 0 から i < N の間 1 ずつ増やす)
    [ (2) ]
endfor



/* 配列 DAT に整数値を格納する */
j ← 0
for (k を 0 から k < MAX の間 1 ずつ増やす)
    if (CNT[k] ≠ 0)
        for (i を [ (3) ] の間 1 ずつ増やす)
            [ (4) ]
            j ← j + 1
        endfor
    endif
endfor
```

(1), (3) の解答群

ア. 0 から k ＜ MAX　　イ. 0 から k ≦ MAX

ウ. 1 から k ≦ MAX　　エ. 0 から i ＜ CNT[k]

オ. 0 から i ≦ CNT[k]　　カ. 1 から i ＜ CNT[k]

(2), (4) の解答群

ア. CNT[i] ← CNT[i] ＋ 1

イ. CNT[DAT[i]] ← CNT[DAT[i]] ＋ 1

ウ. DAT[i] ← DAT[i] ＋ 1

エ. DAT[i] ← i

オ. DAT[i] ← j

カ. DAT[j] ← k

＜設問２＞　次のバケットソートの欠点に関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。

バケットソートは，データ数に対して最大値(MAX)が大きいほど配列 CNT の要素数が左右され，未使用領域も増加するという欠点がある。例えば，0～99 の範囲の整数値が 30 個あるとき，配列 CNT の要素数は［　(5)　］個必要になる。

(5) の解答群

ア．29　　イ．30　　ウ．99　　エ．100

問題５　次の基本選択法のアルゴリズムに関する記述を読み，設問に答えよ。

1 次元配列 data[0] ～ data[n-1] に n 個のデータが格納されている。このデータを基本選択法により整列する。基本選択法とは，最小値（または最大値）を繰り返し見つけて昇順（または降順）に並び替える整列アルゴリズムである。なお，本問題における配列の要素番号は 0 から始める。

＜設問１＞　次の基本選択法に関する説明およびプログラム中の［　　］に入れるべき適切な字句を解答群から選べ。

[基本選択法による昇順への整列]

配列の要素を順に探索し，探索範囲内での最小値を求めつつ，探索範囲を狭めていくことで行う。例えば，data=[50，20，30，10，40]という配列の場合，図 1 のように最小値を先頭の添字から順に確定することで昇順への整列が完了する。

最初の探索範囲 data[0]～data[4]では，data[0]に探索範囲内の最小値である 10 を確定し，次の探索範囲 data[1]～data[4]では，data[1]に探索範囲内の最小値である 20 を確定する。以降，同様の手順を繰り返す。なお，data[4]は，data[3]が確定した時点で要素が確定する。

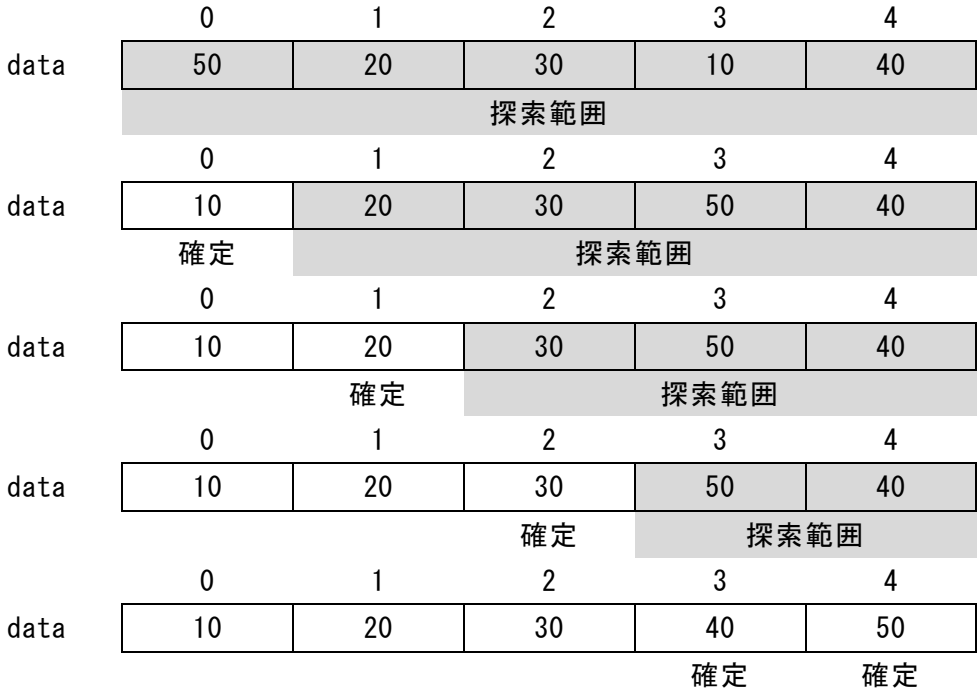


図１　配列要素の探索を行い，配列の値を確定する手順

[最小値の求め方]

本問題における最小値は，全ての要素を比較することにより求める。探索範囲内で最小値が見つかった場合，その先頭要素との交換を行う。探索範囲内の探索が終了した時点でその先頭の要素に最小値が確定する。

例えば，配列 data の各要素が[50, 20, 30, 10, 40]のとき，data[0]～data[4]の探索を完了し data[0]に 10 を確定する場合，図 2 のように data[0]より小さい値がある要素と逐次交換する。

同様の手順で，配列 data の各要素が data[0]の確定後の[10, 50, 30, 20, 40]であるとき， data[1]～data[4]の範囲の探索を完了し data[1]に 20 を確定する場合，data[0]～data[4]の状態は (1) となる。

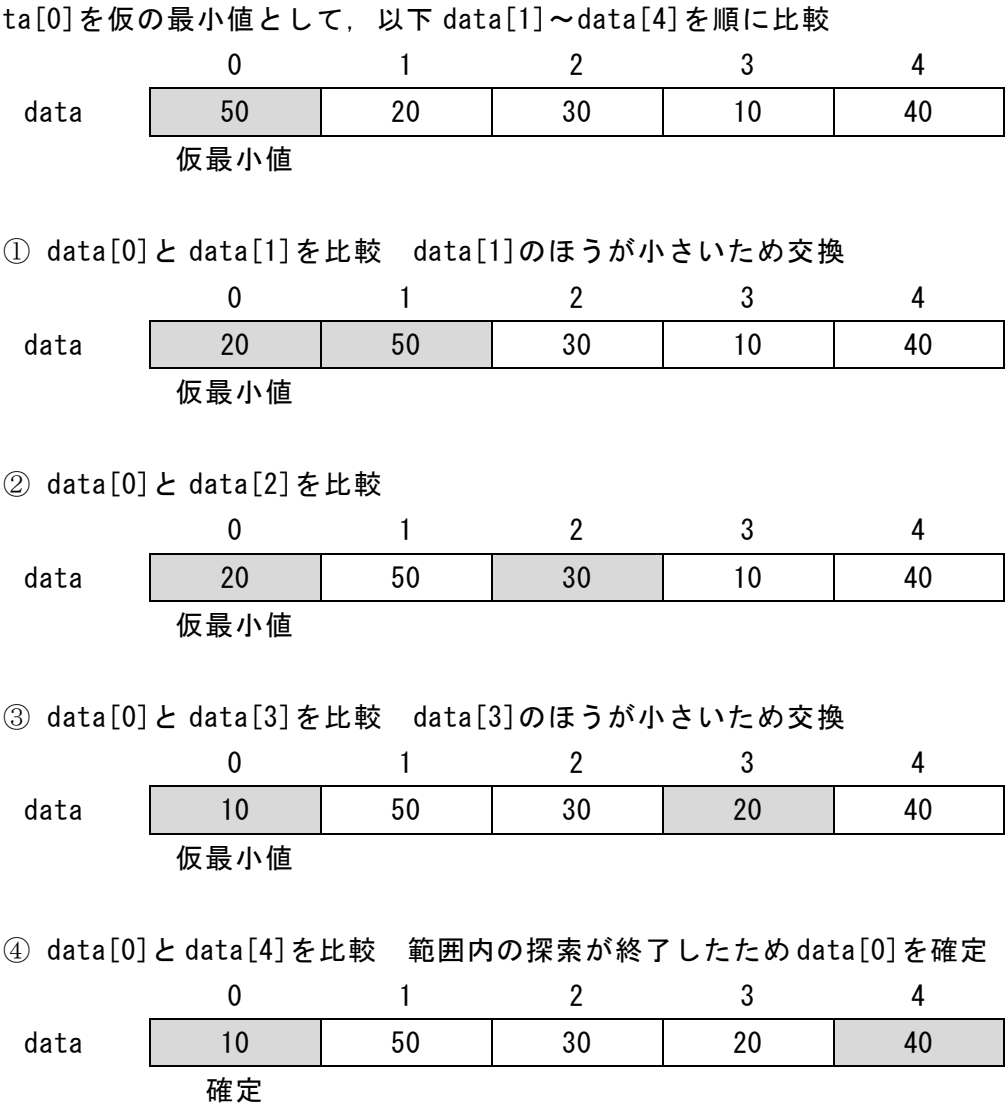


図２　最小値の求め方

(1) の解答群

ア．[10, 20, 30, 50, 40]　　イ．[10, 20, 40, 30, 50]

ウ．[10, 20, 40, 50, 30]　　エ．[10, 20, 50, 30, 40]

[プログラムの説明]

配列 data の要素を順に探索し，基本選択法により昇順に整列する ascendingSort メソッドを持つ SelectionSort クラスである。ascendingSort メソッドは SelectionSort クラス内の main メソッドから呼び出され実行される。以下，ascendingSort メソッドの引数とその並び，および戻り値を示す。

[ascendingSort メソッドの説明]

main メソッドより引数として受け取った配列 data に対し，基本選択法により昇順に整列を行う。整列結果を戻り値として main メソッドに渡し，main メソッドで結果の表示を行う。

表　ascendingSort メソッドの引数と戻り値

引数/戻り値	データ型	意味
引　数	整数型の配列 data	整列対象のデータ
	整数型　　len	データの要素数
戻り値	整数型の配列 data	整列後のデータ

[プログラム]

```
○整数型の配列：ascendingSort( 整数型の配列：data, 整数型：len)
  整数型：i, j, swap
  i   ← 0
  j   ← 0
  swap   ← 0
  while( (2) )
      j    ← i + 1
      while( (3) )
          if( (4) )
              swap       ← data[i]
              data[i]    ← data[j]
              data[j]    ← swap
          endif
          j   ← j + 1
      endwhile
      i    ← i + 1
  endwhile
  return data
```

(2), (3) の解答群

ア. **i ≦ len - 2**　　イ. **i ≦ len - 1**　　ウ. **i ≦ len**

エ. **j ≦ len - 2**　　オ. **j ≦ len - 1**　　カ. **j ≦ len**

(4) の解答群

ア. **data[i - 1] > data[j - 1]**　　イ. **data[i - 1] > data[j]**

ウ. **data[i] > data[j - 1]**　　エ. **data[i] > data[j]**

＜設問２＞　次の降順に整列を行う基本選択法に関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。なお，(4)は設問１と同様である。

［基本選択法による降順への整列と最大値の求め方］

基本選択法による降順への整列は，昇順による整列と同様に行うが，探索範囲内の先頭から確定させていくのは最小値ではなく最大値である。

例えば，配列 data の各要素が[40, 20, 10, 50, 30]のとき，data[0]～data[4]の探索を完了し data[0]に 50 を確定させた場合の data の各要素は[50, 20, 10 , 40, 30]である。同様に data[1]～data[4]の範囲の探索を完了し data[1]に 40 を確定させた場合，その時点の data[0]～data[4]の状態は［　(5)　］となる。

(5) の解答群

ア. [50, 40, 10, 20, 30]　　イ. [50, 40, 20, 10, 30]

ウ. [50, 40, 20, 30, 10]　　エ. [50, 40, 30, 10, 20]

［設問１のプログラムを降順に修正］

昇順・降順いずれの整列も，先頭の要素から順に探索を行い，探索範囲を狭める点で共通している。このことから，設問１におけるプログラムを昇順から降順に変更する場合，ascendingSort メソッドにおいて［　(4)　］を［　(6)　］に変更することで可能となる。

(6) の解答群

ア. **data[i - 1] < data[j - 1]**　　イ. **data[i - 1] < data[j]**

ウ. **data[i] < data[j - 1]**　　エ. **data[i] < data[j]**

＜メモ欄＞

＜メモ欄＞

＜メモ欄＞