文部科学省後援

情報システム試験

令和３年度後期 情報検定

＜実施　令和4年2月13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（ＰＨＳ），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33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。筆記用具が破損等により使用不能となった場合は，手をあげて試験監督者に合図してください。

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。

＜問題の構成＞

必須問題　全員解答

問題１～問題４	2 ページ～16 ページ

選択問題　次の問題から１問選択し解答せよ。

（選択した問題は解答用紙「選択欄」に必ずマークすること）

※選択欄にマークがなく，解答のみマークした場合は採点を行いません。

・Ｃ言語の問題	18 ページ～22 ページ
・表計算の問題	23 ページ～28 ページ
・アセンブラの問題	29 ページ～33 ページ

必須問題

問題１　次のヒープに関する記述を読み，各設問に答えよ。

データ構造の一つに二分木構造があり，データ探索などに利用される。二分木構造の例を図１に示す。図１中の「○」をノード，ノード間を結ぶ線を枝という。枝で結ばれている２つのノードには上が親，下が子という親子関係が存在する。また，一番上位にあるノードを根，子を持たないノードを葉という。

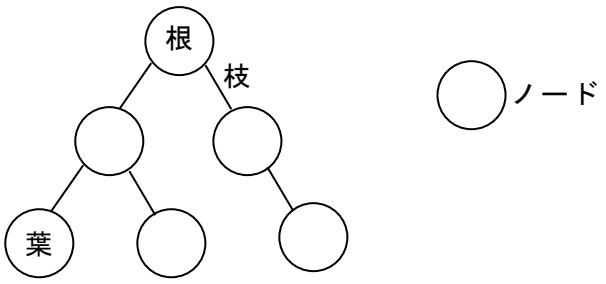


図１　二分木構造の例

＜設問１＞　次の木構造に関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。

二分木構造のうち葉以外のすべてのノードが２つの子ノードを持ち，また，根からすべての葉までの深さが等しい構造のことを完全二分木と呼ぶ。ここで，深さとは根からそのノードに至る経路の枝の数であり，図１の二分木の深さは２である。完全二分木の深さが６のとき，ノードの数は［　(1)　］となる。

また，二分木構造において，親の値が子の値より常に大きいか等しい，または小さいか等しいという制約を満たすものをヒープと呼ぶ。以降では，親の値が子の値より常に大きいか等しいものを扱う。ヒープの例を図２に示す。

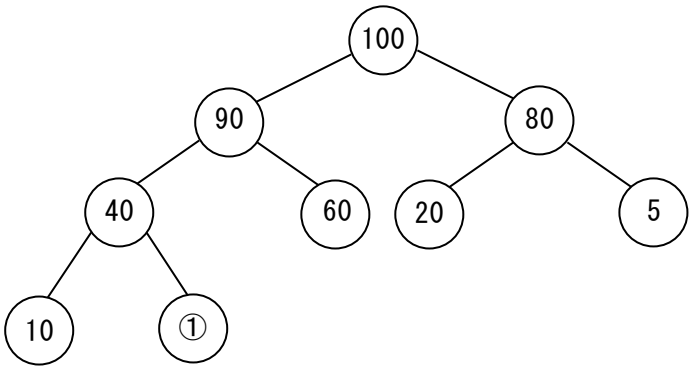


図２　ヒープの例

ここで，図２の①にヒープの制約を満たしたデータを格納する場合，［　(2)　］が適切となる。

次に図２のヒープを配列で表現する場合，下記のルールに基づいて格納する。

［ヒープのルール］

・ヒープ構造を一次元配列 heap に構築する。
・配列の添字は 0 から始まるものとする。
・根は heap[0]に格納する。
・親のノードの位置を i とすると，左の子が格納される位置は i×2＋1，右の子が格納される位置は i×2＋2 とする。

図２のヒープを配列 heap に格納した例の一部を図３に示す。

位置	0	1	2	…
heap	100	90	80	…

図３　一次元配列 heap の一部

・根である 100 は heap[0]に格納する。
・親のノード(100)の位置を 0 とすると，左の子(90)が格納される位置は 0×2＋1＝1 となり，heap[1]に格納する。右の子が格納される位置は 0×2＋2＝2 となり，heap[2]に格納する。

このとき，図２のヒープを一次元配列 heap で表すと［ (3) ］となる。

ここで，図２の①はそのまま①として表記している。

(1) の解答群

ア．63　　イ．64　　ウ．127　　エ．128

(2) の解答群

ア．30　　イ．50　　ウ．55　　エ．70

(3) の解答群

ア．

100	90	80	60	40	①	20	10	5

イ．

100	90	80	40	60	20	5	10	①

ウ．

100	90	80	40	60	10	①	20	5

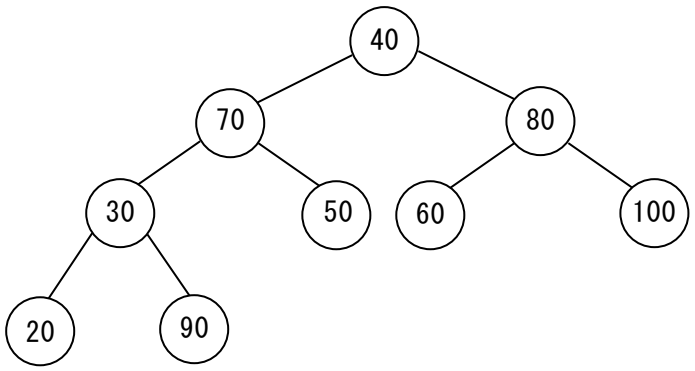
エ．

100	90	60	40	80	20	5	10	①

＜設問２＞　次のヒープの構築に関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。

ヒープは，「親と子要素の中で，最大値を親の位置に格納するようにデータを入れ替える」ことを繰り返し行うことで構築していく。このとき，入替えが発生した場合は，入れ替えた子の位置を新たな親の位置として構築操作を行い，子が存在しなくなるまで構築操作を繰り返す。

ここで，任意の値が格納されたヒープ構築前の二分木と配列を示す（図４）。



位置	0	1	2	3	4	5	6	7	8
heap	40	70	80	30	50	60	100	20	90

図４　ヒープ構築前の二分木と配列

ヒープ構築を開始する位置は，子がないノードから始めるので「(配列の要素数－2)÷2」で求めることができる。なお，除算の結果は小数点以下を切り捨てた整数値とする。このとき，図４の状態でヒープ構築の開始位置である末端の親は［(4)］である。

［(4)］と，その子要素の中で最大値である［(5)］を入れ替える。さらに入れ替えた［(5)］を新たな親として構築操作をするとき，子が存在しないため構築操作を終了する。

次に，heap[2]と，その子要素の中で最大値である heap[6]を入れ替える。さらに heap[6]を新たな親として構築操作をするとき，子が存在しないため構築操作を終了する。

これを進めると，heap[1]と，その子要素の中で最大値である［(6)］を入れ替える。さらに［(6)］を新たな親として構築操作をする。このとき，［(6)］とその子要素を比較すると［(6)］が最大値となるため，入替えは行わず，構築操作を終了する。

このような操作を繰り返し，ヒープ構造を作成する。

以上の操作で構築されたヒープは［(7)］となる。

(4) の解答群

ア．heap[0]　　イ．heap[1]　　ウ．heap[2]　　エ．heap[3]

(5) の解答群

ア．heap[1]　　イ．heap[2]　　ウ．heap[6]　　エ．heap[8]

(6) の解答群

ア．heap[3]　　イ．heap[4]　　ウ．heap[7]　　エ．heap[8]

(7) の解答群

ア．

100	90	80	70	50	60	40	20	30

イ．

100	90	80	60	50	30	10	20	70

ウ．

100	90	40	70	50	60	80	20	30

エ．

100	70	40	90	50	60	80	20	30

問題２　次のバブルソート法に関する記述を読み，各設問に答えよ。

［バブルソート法の説明］

1 次元配列 dat[0]～dat[n－1]に n 個のデータが格納されている。このデータを，バブルソート法により昇順に整列する。バブルソート法は，隣接する要素間で大小の判断を繰り返しながら整列するアルゴリズムである。繰返しの継続条件の違いにより方法１と方法２の二つの方法を示す。

［方法１の説明］

手順１：配列の添字 j の値を 0 から，1 ずつ増やしながら n-1 より小さい間，手順 2 を実行する。

手順２：配列の末尾から隣接する要素を順次比較し，最小値を dat[j]に求める。図１に n＝5 とした例を示す

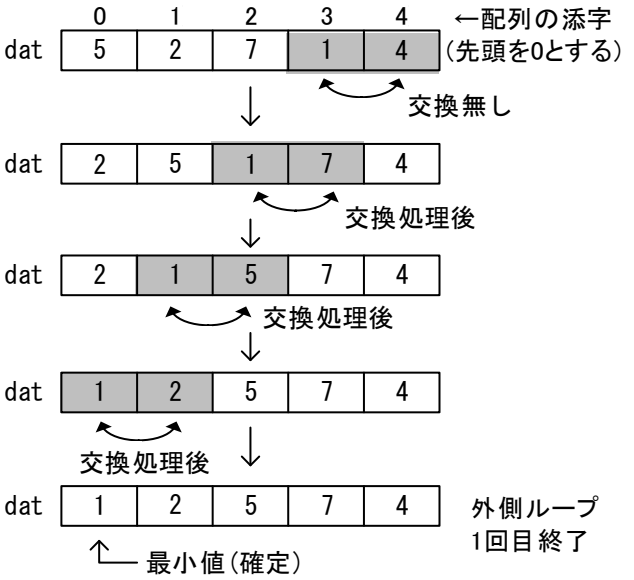


図１　n＝5 とした手順 2 の例

［方法２の説明］

隣接要素間の比較であるため，図１のように全要素の比較が終了した時点で交換が発生していなければ整列が終了することになる。そこでスイッチを使って交換の有無を判断する。スイッチの初期値はオン(sw＝1)とする。

手順１：スイッチをオフ(sw＝0)にして，方法１の手順 2 を実行する。ただし，j の値は方法１と同様に 0 から 1 ずつ増加させ，交換が発生したときはスイッチをオン(sw＝1)とする。

手順２：スイッチがオンの間，手順 1 を実行する。

＜設問１＞　次のバブルソート法に関する２つの流れ図中の［　　］に入れるべき適切な字句を解答群から選べ。

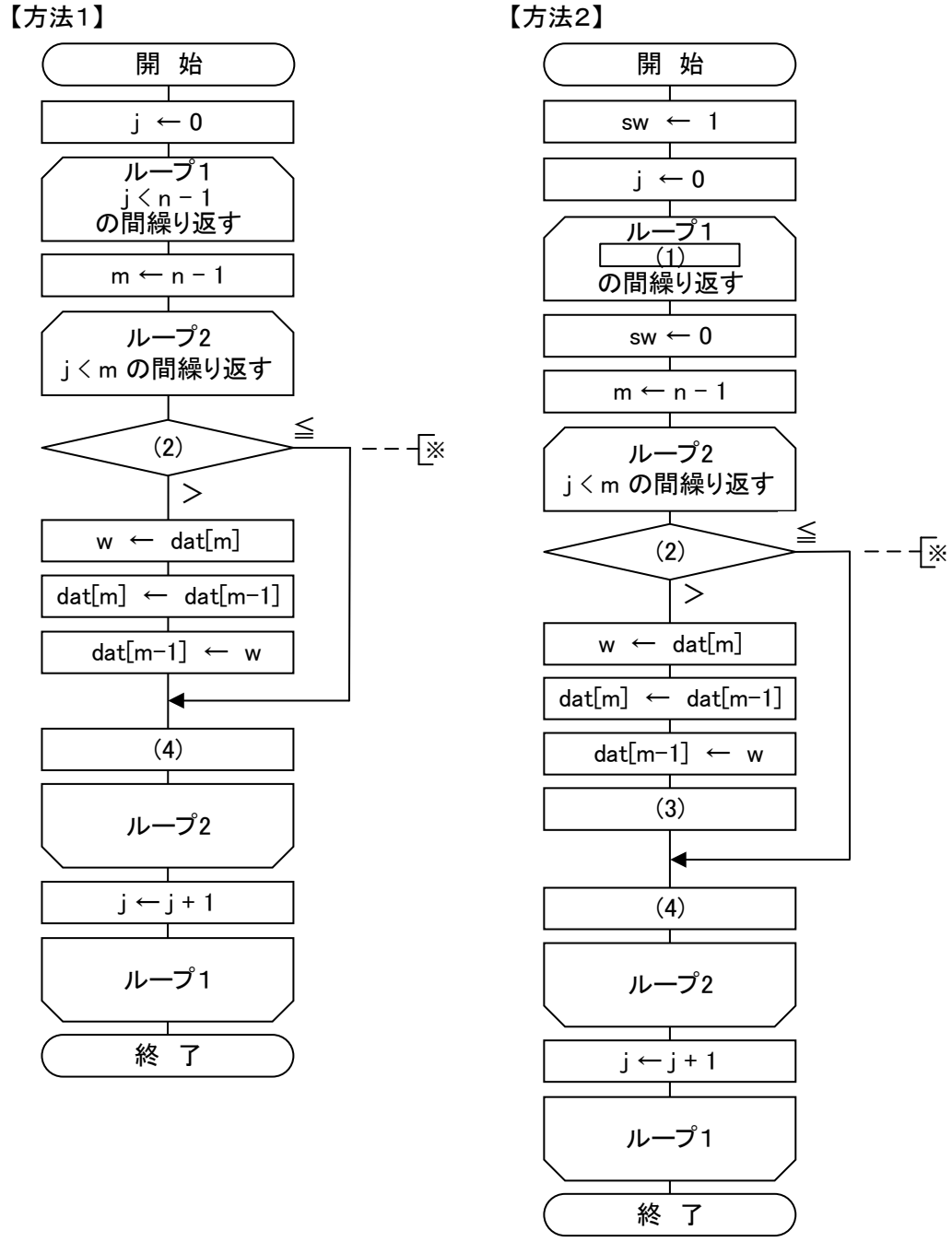


図２　二つの方法によるバブルソート法の流れ図

(1) の解答群

ア．sw＝0　　イ．sw＝1　　ウ．sw＝n

(2) の解答群

ア．dat[m-1]：dat[m]　　イ．dat[m]：dat[m-1]

ウ．dat[m]：dat[m+1]　　エ．dat[m+1]：dat[m]

(3)，(4) の解答群

ア．m ← 0　　イ．m ← 1　　ウ．m ← m－1

エ．m ← m＋1　　オ．sw ← 0　　カ．sw ← 1

＜設問２＞　図２の流れ図に関する適切な記述を (5) の解答群から選べ。

(5) の解答群

ア．図２の流れ図中※で示される比較回数は,配列 dat の内容が同じとき，方法２の方が方法１より必ず多くなる。

イ．図２の流れ図中※で示される方法１の比較回数は，配列 dat の内容にかかわらず一定である。

ウ．方法２では，処理終了後の sw の値は１である。

問題３　次のデータ探索に関する記述を読み，設問に答えよ。

多くのデータの中から目的のデータを探索する方法は，データ構造やデータの並びにより様々なものが考えられる。ここでは，配列の中から探索する方法を考える。

配列から探索する場合には，配列が整列済みかどうかにより２つの手法が考えられる。一つは配列が整列済みかどうかに関係なく処理できる線形探索であり，もう一つは配列が整列済みの場合に処理できる二分探索である。

なお,この問題では整数値のみを扱うものとし,配列は ARY[],配列の大きさは SIZE,探索するデータは X にあらかじめ格納されているものとする。なお，配列の添字は 0 から始まり，配列中に同じ値は存在しないものとする。

［線形探索について］

配列内のデータを１つずつ順番に目的のデータと比較する方法である。ここでは配列の先頭から末尾に向かって順番に探索する。

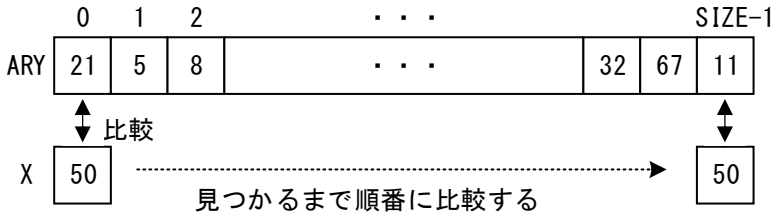


図１　線形探索

［線形探索の流れ図の説明］

ARY[]から X に格納されたデータを探索する。見つかった場合はその位置（配列の添字）を，見つからなかった場合は－1 を表示する。

＜設問１＞　次の線形探索の流れ図中の［　　］に入れるべき適切な字句を解答群から選べ。

(1) の解答群

ア．K ＞ 0　　イ．K ＜ SIZE － 1

ウ．K ＜ SIZE　　エ．K ≦ SIZE

(2) の解答群

ア．RESULT ← 0　　イ．RESULT ← K

ウ．RESULT ← K － 1　　エ．RESULT ← X

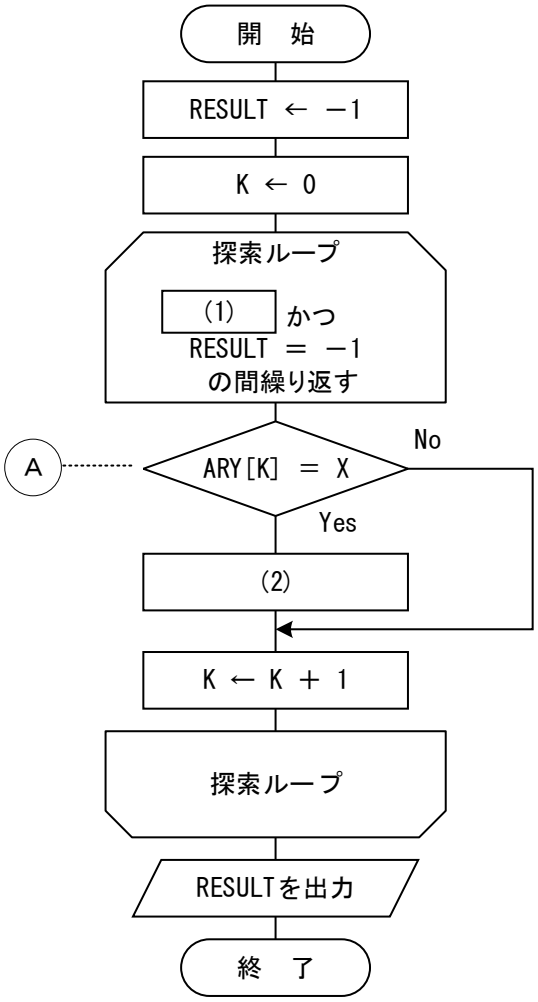


図２　線形探索の流れ図

＜設問２＞　次の線形探索の比較回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。

図２の流れ図中Ⓐが実行される回数を考える。

ARY[]の大きさをＮとすれば，最小比較回数は１回であり最大比較回数は［(3)］回である。また，平均比較回数は，おおよそ［(4)］回である。

(3)，(4) の解答群

ア．Ｎ ／ ２　　イ．Ｎ － １

ウ．Ｎ　　エ．２Ｎ

［二分探索について］

整列済みの配列に対して行われる探索方法で，配列の探索範囲を半減させながら処理するものである。ここでは，ARY[]は昇順に整列済みとする。

［二分探索の手順］

探索範囲の先頭要素の添字をL，末尾要素の添字をH とする。

以下の手順の２と３を繰返すが，その途中で L>H となった時は，目的のデータが探索できなかった場合である。よって，繰返しは L>H となるか探索できた場合に処理を終える。

１．L を 0，H を SIZE－1 とする。

２．L と H の中間を求める。ここでは変数 M に (L＋H)÷2 として求める。なお，小数点以下は切り捨てる。

３．ARY[M]と X を比較し，

(Case 1) ARY[M]＝X であれば見つかった。

(Case 2) ARY[M]＞X であれば，X が格納されている位置は M より小さい位置である可能性があるため，H へ M－1 を代入する。

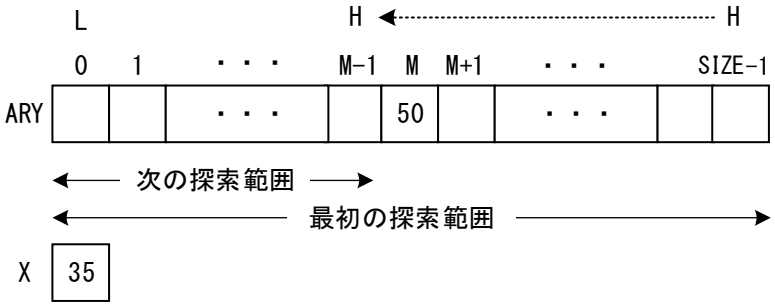


図３　ARY[M]＞X の場合の処理

(Case 3) ARY[M]＜X であれば，X が格納されている位置は M より大きい位置である可能性があるため，L へ M＋1 を代入する。

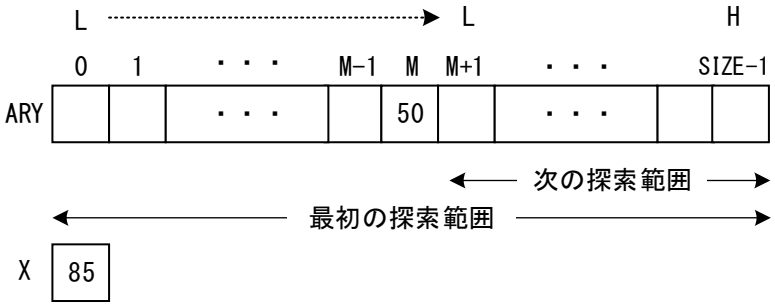


図４　ARY[M]＜X の場合の処理

＜設問３＞　次の二分探索の流れ図中の［　　］に入れるべき適切な字句を解答群から選べ。

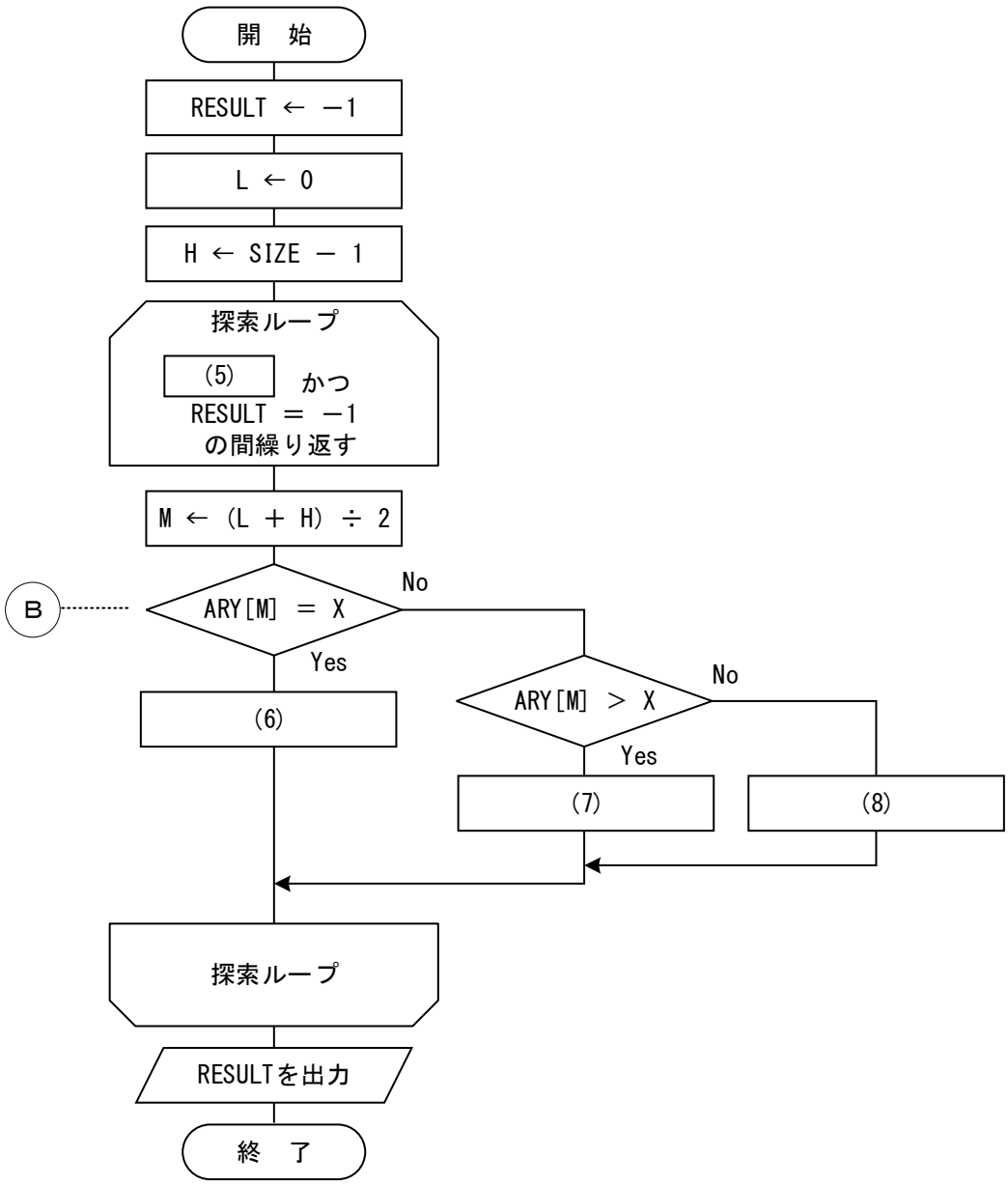


図５　二分探索の流れ図

(5) の解答群

ア．L＜H　　イ．L≦H　　ウ．L＞H　　エ．L＝H

(6) ～ (8) の解答群

ア．H ← L − 1　　イ．H ← M − 1　　ウ．L ← H ＋ 1

エ．L ← M ＋ 1　　オ．RESULT ← M − 1　　カ．RESULT ← M

＜設問４＞　次の二分探索の比較回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。

図５の流れ図中Ⓑが実行される回数を考える。

二分探索では一度の比較で次の探索範囲がほぼ半分になる。このことから，一般的に最大比較回数は，「$\log_2 N$ を超える最小の整数」と言われているが，2^0から順番に２のべき乗を計算してみて，その結果が配列の要素数を超えた時の指数の値を見れば知ることができる。例えば，配列の要素数が 10 の場合では、$2^0=1$，$2^1=2$，$2^2=4$，$2^3=8$，$2^4=16$ となり，指数が４のとき 10 を超えるので，最大比較回数は４回となる。

同様に考えて，ARY[]の要素数が 100 であれば最大比較回数は［　(9)　］回になる。

また，最大比較回数が N 回であった配列がある場合，この配列の大きさを２倍にした場合の最大比較回数は［　(10)　］回になる。

(9) の解答群

ア．6　　イ．7　　ウ．8　　エ．9

(10) の解答群

ア．N　　イ．N ＋ 1　　ウ．2N － 1　　エ．2N

問題４　次のプログラムの説明を読み，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］

受け取ったデータＸ(整数の 0～15)をビット列として配列 data_b に格納した後，ハミング符号を生成して返す関数 Make_Ham である。

ハミング符号とはパリティチェックを拡張した誤り検出符号で，データの誤り検出と１ビット誤りの訂正ができるものである。ここでは，データビットを４ビット，検査用ビットを３ビットとしたハミング符号を扱うものとする。

ここで扱うデータビットと検査用ビットは，図１のとおりである。なお，検査用ビットの c1～c3 は，次の式で求める。また，各配列の添字は 0 から始まる。

c1 ＝ (x1 + x2 + x3) mod 2

c2 ＝ (x1 + x3 + x4) mod 2

c3 ＝ (x2 + x3 + x4) mod 2　　※　a mod b は，a を b で割った余りである。

	添字	0	1	2	3
データビット	配列 data_b	x1	x2	x3	x4

	添字	0	1	2
検査用ビット	配列 check_b	c1	c2	c3

図１　データビットと検査用ビット

データビットの後ろに検査用ビットを結合し，図２のハミング符号を生成する。

	添字	0	1	2	3	4	5	6
ハミング符号	配列 Ham_b	x1	x2	x3	x4	c1	c2	c3

図２　ハミング符号

［擬似言語の記述形式の説明］

記述形式	説明
○	手続き，変数などの名前，型などを宣言する。
・変数 ← 式	変数に式の値を代入する。
/* 文 */	注釈を記述する。
▲ 条件式 ・処理１ ―――― ・処理２ ▼	選択処理を示す。 条件式が真の時は処理１を実行し， 偽の時は処理２を実行する。
■ 条件式 ・処理 ■	前判定繰り返し処理を示す。 条件式が真の間，処理を実行する。

［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 ▲
乗除演算	＊，／，％	
加減演算	＋，－	
関係演算	＞，＜，≧，≦，＝，≠	
論理積	and	
論理和	or	▼ 低

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(mod)を表す。

[プログラム]

○Make_Ham(整数型：X)
○整数型配列：data_b[4], check_b[3], ham_b[7]
○整数型：i

```
/* データから 1 ビットずつ取り出す */
・(1)
■ i >= 0
    ・(2)
    ・i ← i - 1
■

/* 検査ビット c1,c2,c3 の計算 */
・check_b[0] ← (data_b[0] + data_b[1] + data_b[2]) % 2
・check_b[1] ← (data_b[0] + data_b[2] + data_b[3]) % 2
・check_b[2] ← (data_b[1] + data_b[2] + data_b[3]) % 2

/* ハミング符号を作成 */
・(3)
■ i < 4
    ・(4)
    ・i ← i + 1
■
・(3)
■ i < 3
    ・(5)
    ・i ← i + 1
■
```

(1)，(3) の解答群

ア. i ← 0　　イ. i ← 1
ウ. i ← 2　　エ. i ← 3

(2) の解答群

ア. data_b[i] ← X % 2
　　X ← X / 2
ウ. X ← X % 2
　　data_b[i] ← X / 2

イ. data_b[i] ← X / 2
　　X ← X % 2
エ. X ← X / 2
　　data_b[i] ← X % 2

(4)，(5) の解答群

ア. ham_b[i] ← data_b[i]
ウ. ham_b[i] ← check_b[i]

イ. ham_b[3+i] ← data_b[i]
エ. ham_b[4+i] ← check_b[i]

＜ 選 択 問 題 ＞

選択問題は問題から１つ選択し解答せよ。
選択した問題は必ず，解答用紙「選択欄」にマークすること。
※選択欄にマークがなく，解答のみの場合は採点を行いません。

各構成は以下のとおり。

選択問題

選択問題　C言語の問題

次のデータの圧縮に関する記述を読み，各設問に答えよ。

［データの圧縮について］

圧縮は，データの保存や伝送時に容量を減らすために行われる。圧縮の方法にはいくつかあるが，ここではランレングス法を考える。なお，使用する文字は英数字のみとし，文字列は１文字ずつ配列に格納されるものとする。

［ランレングス法による圧縮について］

同じ値が連続するところに注目した圧縮方法で，連続するデータがある場合，データの値とその個数の情報に置き換えるものである。ただし，連続していないデータとの混在を実現するため，圧縮していることを示すための記号を付加して区別する。ここでは４文字以上連続する場合にのみ圧縮するものとし，その形式は図のとおりである。

対象文字	制御文字	個数

図　圧縮の形式

・制御文字は'$'を使う。
・個数は4～26を'D'～'Z'の文字で表す。連続する文字が26以上の場合は一度圧縮の形式を出力し，再び１から文字数を数える。例えば，'a'が30個連続する場合は，aZaD となる。

次の表は，ランレングス法で圧縮する前の文字列と圧縮後の文字列の例である。

表　ランレングス法で圧縮する前と後

圧縮前	圧縮後
abcd	abcd
hhkkkkttt	hhk$Dttt
xxxxxxxxpppp	xHpD

＜設問１＞　次の文字列圧縮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］

配列に格納された文字列をランレングス法で圧縮を行うcompress関数と，ランレングス法で圧縮された文字列を展開するunCompress関数である。compress関数は4～26の整数値を'D'～'Z'に変換するtoAlpha関数を，unCompress関数は'D'～'Z'を4～26の

整数値に変換するtoNum関数を呼び出す。

［関数の説明］

compress 関数

引　数：unComp（圧縮前の文字列：文字型配列），comp（圧縮後の文字列：文字型配列）

機　能：unComp に格納された文字列に対してランレングス法で圧縮を行った結果をcomp に格納する。

戻り値：なし。

toAlpha 関数

引　数：n（連続する文字数：整数値）

機　能：連続する文字数を英字に変換する。ただし，4 未満または 26 より大きい場合は'¥0'を返却する。

戻り値：4～26 を'D'～'Z'に変換した文字，または'¥0'。

［プログラム］

```
char toAlpha(int n) {
    if (n < 4 || n > 26) {
        return '¥0';
    } else {
        return 'A' + n - 1;
    }
}

void compress(char *unComp, char *comp) {
    int  i, k, n, p, length, sw;
    char ch;

    /* unCompとcompの添字を初期化 */
    p = 0;
    k = 0;
    /* unCompの文字数を取得 */
    length = strlen(unComp);
    while(k < length) {
        /* 同じ文字が連続する個数をnに求める */
        ch = unComp[k];
        n = 0;
        sw = 0;
        while(sw == 0) {
```

```
            if (k < length) {
                /* 文字が26文字以下で連続しているかを判断する */
                if ( (1) ) {
                    sw = 1;
                } else {
                    /* 26文字以下で連続している場合の処理 */
                    n++;
                    k++;
                }
            } else {
                sw = 1;
            }
        }
        /* 圧縮するかを判断 */
        if (n >= 4) {
            /* 圧縮する場合の処理 */
            comp[p] = ch;
            comp[p + 1] = '$';
            comp[p + 2] = toAlpha(n);
            (2) ;
        } else {
            /* 圧縮しない場合の処理 */
            for(i = 0; i < n; i++) {
                comp[p] = ch;
                (3) ;
            }
        }
    }
    comp[p] = '¥0';
}
```

(1) の解答群

ア. `ch != unComp[k] || n >= 26`　　イ. `ch != unComp[k] && n >= 26`

ウ. `ch == unComp[k] || n < 26`　　エ. `ch == unComp[k] && n < 26`

(2), (3) の解答群

ア. `p--`　　イ. `p++`

ウ. `p += 2`　　エ. `p += 3`

＜設問２＞　次のランレングス法で圧縮された文字列の展開に関するプログラム中の［　　］に入れるべき適切な字句を解答群から選べ。

［関数の説明］

unCompress 関数

引　数：comp（圧縮後の文字列：文字型配列），unComp（圧縮前の文字列：文字型配列）

機　能：comp に格納されたランレングス法で圧縮された文字列を展開して unComp に格納する。

戻り値：なし。

toNum 関数

引　数：ch（連続する数に対応した文字：文字）

機　能：連続する数を表す文字（'D'～'Z'）を整数値（4～26）に変換する。ただし，文字コードが'D'未満または'Z'より大きい場合は-1 を返却する。

戻り値：'D'～'Z'を 4～26 に変換した整数値，または-1。

［プログラム］

```
int toNum(char ch) {
    if (ch < 'D' || ch > 'Z') {
        return -1;
    } else {
        return ch - 'A' + 1;
    }
}

void unCompress(char *comp, char *unComp) {
    int i, k, n, p, length;
    char ch;

    /* compとunCompの添字を初期化 */
    p = 0;
    k = 0;
    /* compの文字数を取得 */
    length = strlen(comp);
    while(k < length) {
        /* compからunCompへ１文字転送 */
        ch = comp[k];
        [  (4)  ];
        k++;
```

```
            p++;
            if (k < length) {
                /* 圧縮されているかを判断して展開する */
                if (comp[k] == '$' && k < length - 1) {
                    n = toNum( (5) );
                    /* 圧縮された文字を展開する */
                    for(i = 1; i < n; i++) {
                        unComp[p] = ch;
                        (6);
                    }
                    (7);
                }
            }
        }
        unComp[p] = '¥0';
    }
```

(4) の解答群

ア. unComp[k] = ch　　イ. unComp[k] = p

ウ. unComp[p] = ch　　エ. unComp[p] = k

(5) の解答群

ア. comp[k - 1]　　イ. comp[k]

ウ. comp[k + 1]　　エ. comp[k + 2]

(6), (7) の解答群

ア. k++　　イ. k += 2

ウ. p++　　エ. p += 2

選択問題　表計算の問題

次の表計算ソフトの記述を読み，各設問に答えよ。

この問題で使用する表計算ソフトの仕様は下記のとおりである。

COUNTIFS 関数

範囲内のセルの中で複数の条件に一致するセルの数を返す。

書式：COUNTIFS(条件範囲１，検索条件１，条件範囲２，検索条件２，…)

MATCH 関数

検査範囲から検査値が存在するセルの相対的な位置を返す。位置は 1 から始まる相対的な値である。検査範囲は 1 行または 1 列である。検査の型は，検査値と等しい最初の値を検索する場合は 0，検査値以下の最大の値を検索する場合は 1，検査値以上の最小の値を検索する場合は-1 を指定する。

書式：MATCH(検査値，検査範囲，検査の型)

MAX 関数

指定した範囲内の最大値を返す。

書式：MAX(範囲)

MIN 関数

指定した範囲内の最小値を返す。

書式：MIN(範囲)

SMALL 関数

指定された範囲の中で，小さい方から指定した順位のデータを返す。

書式：SMALL(範囲，順位)

SUMIF 関数

指定した検索範囲の中で，条件に一致するセルの合計範囲に対応するセルの値の合計値を返す。

書式：SUMIF(検索範囲，条件，合計範囲)

VLOOKUP 関数

検索範囲から，検索値を探し，位置で指定した列の値を返す。位置は１から始まる相対的な値であり，検索範囲中に見つけた行の中で，左から何番目の列かを示す。検索方法は 0 または 1 を指定し，0 の場合は完全に一致する値を，1 の場合は検索値以下の最大値を探す。

書式：VLOOKUP(検索値，検索範囲，位置，検索方法)

式
＝に続いて計算式や関数などを入力する。

セル番地の絶対参照
セル番地に＄を付けることで，絶対番地(絶対参照)を表す。

別シートの参照
ワークシート名に「！」を付けてセル位置を指定することにより，別のワークシートを参照できる。
例：ワークシート名「集計」のセル A1 を参照する場合は，「集計!A1」と記述する。

J パソコンショップではパソコンを購入する際にあわせて購入するアクセサリの傾向について調査するため，アンケートを実施した。

お客様の年齢・性別・ご職業をお聞かせ下さい。

年齢	□20才未満　□20代　□30代　□40代　□50代　□60代　□70代以上
性別	□男性　□女性　□未回答
ご職業	□会社員　□公務員　□自営業　□主婦　□学生　□アルバイト　□その他

Q1. 購入したPCの種類は何ですか？

□デスクトップ　□ノートPC　□タブレット　□その他

Q2. 同時に購入したアクセサリは何ですか？

□ディスプレイ　□外付ハードディスク　□外付SSD　□ルータ　□その他　□購入していない

Q3. Q2でアクセサリを購入と回答の方，購入した理由は何ですか？

使いやすさ	□満足している	□やや満足している	□どちらともいえない	□あまり満足していない	□満足していない
価格の妥当性	□満足している	□やや満足している	□どちらともいえない	□あまり満足していない	□満足していない
機能性	□満足している	□やや満足している	□どちらともいえない	□あまり満足していない	□満足していない

Q4. Q2でアクセサリを購入と回答の方，購入した店舗はどこですか？

□同じ市内の店舗　□郊外の店舗　□他県の店舗

ご協力いただき，誠にありがとうございました。

図１　アンケート用紙

アンケート用紙を図１に示す。アンケートの結果は，図２の「アンケートデータ」ワークシートの 200 行目まで入力済みである。なお，お客様番号は，便宜上入力順に採番している。また，各項目の回答は，例えば年齢では，20 才未満は 1，20 代は 2，30 代は 3，・・・のように，左側から 1，2，3，・・・と数字で入力している。各項目と数字の対応は，図３の「項目表」ワークシートに記録している。

	A	B	C	D	E	F	G	H	I	J
1	XX年XX月　　アンケートデータ									
2	アンケート項目	年齢	性別	職業	Q1.	Q2.	Q3.アクセサリを購入した理由			Q4.
3	お客様番号				購入したPCの種類	同時購入アクセサリ	使いやすさ	価格の妥当性	機能性	アクセサリ購入場所
4	21090001	4	2	7	4	1	2	3	3	1
5	21090002	1	2	4	2	2	1	2	1	1
6	21090003	1	2	3	1	3	2	1	3	3
7	21090004	2	2	2	1	1	2	2	3	1
8	21090005	4	2	6	3	1	3	5	1	3
:	:	:	:	:	:	:	:	:	:	:
199	21090196	6	2	1	3	5	1	3	2	3
200	21090197	7	2	2	1	6	3	2	1	1

図２　「アンケートデータ」ワークシート

	A	B	C	D	E	F	G	H
1	年齢表			PCの種類			アンケートの評価	
2	1	20才未満		1	デスクトップ		1	満足している
3	2	20代		2	ノートPC		2	やや満足している
4	3	30代		3	タブレット		3	どちらともいえない
5	4	40代		4	その他		4	あまり満足していない
6	5	50代					5	満足していない
7	6	60代		アクセサリの種類				
8	7	70代以上		1	ディスプレイ		アクセサリの購入場所	
9				2	外付ハードディスク		1	PCを購入した店舗
10	性別			3	外付SSD		2	PCを購入した店舗とは別の店舗
11	1	男		4	ルータ		3	ネット通販
12	2	女		5	その他			
13	3	未回答		6	購入していない		アクセサリの購入地域	
14							1	同じ市内の店舗
15	職業			購入した理由			2	郊外の店舗
16	1	会社員		1	使いやすさ		3	他県の店舗
17	2	公務員		2	価格の妥当性			
18	3	自営業		3	機能性			
19	4	主婦						
20	5	学生						
21	6	アルバイト						
22	7	その他						

図３　「項目表」ワークシート

＜設問１＞　次の「PC・アクセサリ販売集計表」ワークシートの作成に関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。

	A	B	C	D	E	F	G	H	I
1		アクセサリの種類	1	2	3	4	5	6	
2	PCの種類		ディスプレイ	外付ハードディスク	外付SSD	ルータ	その他	購入していない	合計
3	1	デスクトップ	10	8	3	6	5	8	40
4	2	ノートPC	9	12	5	4	10	11	51
5	3	タブレット	8	6	9	14	9	7	53
6	4	その他	3	13	12	9	7	9	53
7		合計	30	39	29	33	31	35	197

図４　「PC・アクセサリ販売集計表」ワークシート

・ B 列の PC の種類は，セル A3～A6 に 1～4 を入力し，「項目表」ワークシートから数字で検索して表示する。セル B3 に次の式を入力し，セル B4～B6 まで複写した。
= [　(1)　]

・ ２行目のアクセサリの種類は，セル C1～H1 に 1～6 を入力し，「項目表」ワークシートから数字で検索して表示する。セル C2 に次の式を入力し，セル D2～H2 まで複写した。
= [　(2)　]

・ PC の種類によってどのアクセサリを購入しているかを集計するため，セル C3 に次の式を入力し，セル D3～H3，C4～H6 まで複写した。
= COUNTIFS([　(3)　], [　(4)　])

・ ７行目はアクセサリごとの販売数の合計を表示した。

・ I 列は PC ごとのアクセサリの販売数の合計と総合計を表示した。

同様に，年齢によってどのアクセサリを購入しているかを集計した「年齢・アクセサリ販売集計表」ワークシートなども作成した。

(1) の解答群

ア．VLOOKUP(A$3, 項目表!D$2:E$5, 2, 0)
イ．VLOOKUP(A$3, 項目表!D2:E5, 2, 0)
ウ．VLOOKUP(A3, 項目表!$D2:$E5, 2, 0)
エ．VLOOKUP(A3, 項目表!D$2:E$5, 2, 0)

(2) の解答群

ア. VLOOKUP($C$1, 項目表!$D8:$E13, 2, 0)

イ. VLOOKUP($C$1, 項目表!D8:E13, 2, 0)

ウ. VLOOKUP(C1, 項目表!$D8:$E13, 2, 0)

エ. VLOOKUP(C1, 項目表!D$8:E$13, 2, 0)

(3) の解答群

ア. アンケートデータ!$E$4:$E$200, $A3

イ. アンケートデータ!$E4:$E200, $A3

ウ. アンケートデータ!E$4:E$200, A$3

エ. アンケートデータ!E4:E200, A3

(4) の解答群

ア. アンケートデータ!$F$4:$F$200, $C1

イ. アンケートデータ!$F$4:$F$200, C$1

ウ. アンケートデータ!F$4:F$200, C1

エ. アンケートデータ!F4:F200, C1

＜設問２＞　次の「購入内訳表」ワークシートの作成に関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。

	A	B	C	D	E	F	G	H	I
1		購入理由	1	2	3		評価順		
2		アクセサリの種類	使いやすさ	価格の妥当性	機能性	合計	1番	2番	3番
3	1	ディスプレイ	76	84	66	226	機能性	使いやすさ	価格の妥当性
4	2	外付ハードディスク	99	125	93	317	機能性	使いやすさ	価格の妥当性
5	3	外付SSD	80	89	86	255	使いやすさ	機能性	価格の妥当性
6	4	ルータ	92	95	99	286	使いやすさ	価格の妥当性	機能性
7	5	その他	75	74	73	222	機能性	価格の妥当性	使いやすさ
8	6	購入していない	104	109	92	305			
9		合計	347	393	344	1,084			

図５　「購入内訳表」ワークシート

・ B 列のアクセサリの種類は，セル A3～A8 に 1～6 を入力し，「項目表」ワークシートから数字で検索して表示した。

・ ２行目の購入理由は，セル C1～E1 に 1～3 を入力し，「項目表」ワークシートから数字で検索して表示した。

・ アクセサリの種類によって購入理由の評価点を集計するため，セル C3 に次の式を入力し，セル D3～E3，C4～E8 まで複写した。

= SUMIF([　(5)　] , [　(6)　])

・ 9行目は購入理由ごとの評価点の合計を表示した。

・ F列はアクセサリごとの評価点の合計と総合計を表示した。

・ G列は各アクセサリの評価で1番目に高い評価を表示する。セルG3に次の式を入力し，セルG4～G7まで複写した。なお，アンケートデータでの満足度は1が一番高く，2，3，4と段々低くなり，5が一番低い。また，G～I列は同じ表示になる場合がある。

= VLOOKUP((7) , 項目表!D$16:E$18, 2, 0)

・ H列は各アクセサリの評価で2番目に高い評価を表示する。セルH3に次の式を入力し，セルH4～H7まで複写した。

= VLOOKUP((8) , 項目表!D$16:E$18, 2, 0)

・ I列は各アクセサリの評価で3番目に高い評価を表示する。セルI3に次の式を入力し，セルI4～I7まで複写した。

= VLOOKUP((9) , 項目表!D$16:E$18, 2, 0)

(5) の解答群

ア. アンケートデータ!$F$4:$F$200, $A3
イ. アンケートデータ!$F$4:$F$200, A$3
ウ. アンケートデータ!$F4:$F200, $A3
エ. アンケートデータ!F$4:F$200, A3

(6) の解答群

ア. アンケートデータ!$G$4:$G$200
イ. アンケートデータ!$G4:$G200
ウ. アンケートデータ!G$4:G$200
エ. アンケートデータ!G4:G200

(7) ～ (9) の解答群

ア. MATCH(MAX(C$3:E$3), C$3:E$3, 0)
イ. MATCH(MAX(C3:E3), C3:E3, 0)
ウ. MATCH(MIN(C$3:E$3), C$3:E$3, 0)
エ. MATCH(MIN(C3:E3), C3:E3, 0)
オ. MATCH(SMALL(C$3:E$3, 2), C$3:E$3, 0)
カ. MATCH(SMALL(C3:E3, 2), C3:E3, 0)

選択問題　アセンブラの問題

次のアセンブラ言語CASLⅡプログラムの説明を読み，各設問に答えよ。

［プログラムの説明］

A と B 二つの整数の最大公約数を求めるためのプログラムである。最大公約数は，それぞれの数値を素因数分解して求められる素数の中で，共通する素数の積である。次の三つの STEP を順に実行することで求める。ただし，A と B はともに 100 未満の整数であり，10 未満の素数の積で構成される。なお，前の STEP の実行結果となるデータは後の STEP では格納済みとし，設問は STEP1 と STEP3 部分とする。

［STEP1：プログラムERATの説明］

「エラトステネスのふるい」を用いて，10 未満の素数を SOSU 番地から始まる領域に，素数の個数を SCNT 番地に求めるプログラムである。「エラトステネスのふるい」は連続領域に 2 以降の整数を格納し，2 の倍数，3 の倍数，…と素数の倍数が格納されている領域に 0 を格納して消去していく方法である。プログラム ERAT では次の処理を行う。

① SOSU+2 番地から SOSU+11 番地に 2～10 の整数を格納する。

② SOSU+2 番地から SOSU+11 番地まで順にデータを取り出し，取り出したデータが 0 でなければそのデータの倍数が格納されている番地に 0 を格納する。ただし，①で整数を格納した領域内とする。

③ SOSU+2 番地から SOSU+11 番地の中で 0 以外の値を SOSU+0 番地から連続して格納する。

④ 素数の個数を SCNT 番地に格納する。

［STEP2：プログラムの説明］

A と B を素因数分解する。素因数分解とは図１のようにデータを素数の積に分解することである。

A…　36 ＝ 2 × 2 × 3 × 3
B…　42 ＝ 2 × 3 × 7

図１　A と B の素因数分解

プログラム ERAT が実行されると，図２のように 10 未満の素数は SOSU+0 番地から SOSU+3 番地に格納済みであり，A の素因数分解の結果を ASOIN+1 番地以降に，B の素因数分解の結果を BSOIN+1 番地以降に求める。また，A を構成する素数の個数を ASOIN+0 番地に，B を構成する素数の個数を BSOIN+0 番地に格納する。なお，10 未満の素数（2，3，5，7）の個数 4 は SCNT に格納済みである。

SOSU+0	2
+1	3
+2	5
+3	7

ASOIN+0	4
+1	2
+2	2
+3	3
+4	3

BSOIN+0	3
+1	2
+2	3
+3	7

図２　SOSU 番地のデータと STEP2 のプログラム実行後(A＝36,B＝42 の場合)

[STEP3:プログラムGCMの説明]

A と B の最大公約数を求める。STEP2 が実行されると，ASOIN 番地以降および BSOIN 番地以降には図２の値が格納される。この両方に共通する素数を GSOIN+1 番地以降に取り出し，これらの積を求め最大公約数とする。求めた最大公約数は GSOIN 番地に格納する。

＜設問１＞　次のプログラム ERAT 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ERAT] STEP1

行番号	ラベル	命令	オペランド	コメント
100	ERAT	START		
110		RPUSH		
120		LD	GR1,=2	; 整数の初期値 2 を設定
130	LOOP1	ST	GR1,SOSU,GR1	
140		LAD	GR1,1,GR1	
150		［　　(1)　　］		; 10 まで格納したか？
160		JMI	LOOP1	
170		LD	GR0,=0	
180		［　　(2)　　］		; SOSU 番地の指標初期値
190		LD	GR2,=0	
200	LOOP2	LAD	GR1,1,GR1	
210		CPA	GR1,=10	
220		JPL	NEXT1	
230		LD	GR3,SOSU,GR1	
240		CPA	GR3,=0	
250		［　　(3)　　］		
260		ST	GR3,SOSU,GR2	; 素数が確定し詰める
270		LAD	GR2,1,GR2	;
280		LD	GR4,GR1	

```
290     LOOP3   ADDA    GR4,GR1         ; 倍数の位置の指標
300             CPA     GR4,=10
310             JPL     LOOP2
320             (4)                     ; 倍数位置の消去
330             JUMP    LOOP3
340     NEXT1   ST      GR2,SCNT
350             RPOP
360             RET
370     SCNT    DS      1
380     SOSU    DS      11
390             END
```

(1) の解答群

ア. `CPA GR1,=9`　　イ. `CPA GR1,=10`

ウ. `CPA GR1,=11`　　エ. `CPA GR1,=12`

(2) の解答群

ア. `LD GR1,=0`　　イ. `LD GR1,=1`

ウ. `LD GR1,=2`　　エ. `LD GR1,=10`

(3) の解答群

ア. `JZE LOOP1`　　イ. `JZE LOOP2`

ウ. `JZE LOOP3`　　エ. `JZE NEXT1`

(4) の解答群

ア. `ST GR0,SOSU,GR1`　　イ. `ST GR0,SOSU,GR2`

ウ. `ST GR0,SOSU,GR3`　　エ. `ST GR0,SOSU,GR4`

＜設問２＞　次のプログラム GCM 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 GCM] STEP3

```
1000    GCM     START
1010            RPUSH
1020            LD      GR3,=0          ; GSOIN の指標
1030            LD      GR4,=0          ; 倍数の位置に格納する 0
1040            LD      GR1,=1          ; ASOIN の指標
1050    LOOP1   LD      GR2,=1          ; BSOIN の指標
1060            LD      GR0,ASOIN,GR1
```

```
1070    LOOP2   CPA     GR0,BSOIN,GR2
1080            [   (5)   ]         ;A と B に共通の素数か？
1090            LAD     GR3,1,GR3
1100            ST      GR0,GSOIN,GR3
1110            ST      GR4,BSOIN,GR2
1120    NEXT1   LAD     GR2,1,GR2       ;BSOIN の指標を進める
1130            CPA     GR2,BSOIN
1140            JMI     LOOP2
1150            [   (6)   ]         ;ASOIN の指標を進める
1160            CPA     GR1,ASOIN
1170            JMI     LOOP1
1180            LD      GR1,=1          ; 乗算の開始(加算の利用)
1190    LOOP3   ST      GR1,DAT
1200            LD      GR2,GSOIN,GR3   ; 加算回数を GR2 へ
1210    LOOP4   SUBA    GR2,=1
1220            JZE     NEXT2
1230            [   (7)   ]         ; 加算による乗算の実行
1240            JUMP    LOOP4
1250    NEXT2   SUBA    GR3,=1
1260            JPL     LOOP3
1270            ST      GR1,GSOIN
1280            RPOP
1290            RET
1300    ASOIN   DC      4
1310            DC      2
1320            DC      2
1330            DC      3
1340            DC      3
1350    BSOIN   DC      3
1360            DC      2
1370            DC      3
1380            DC      7
1390    DAT     DS      1
1400    GSOIN   DS      7
1410            END
```

(5) の解答群

ア. **JNZ LOOP1**　　イ. **JNZ LOOP2**

ウ. **JNZ LOOP3**　　エ. **JNZ NEXT1**

(6) の解答群

ア．**LAD　GR1,1,GR1**　　イ．**LAD　GR2,1,GR2**

ウ．**LAD　GR3,1,GR3**　　エ．**LAD　GR4,1,GR4**

(7) の解答群

ア．**ADDA　GR1,DAT**　　イ．**ADDA　GR2,DAT**

ウ．**ADDA　GR3,DAT**　　エ．**ADDA　GR4,DAT**