文部科学省後援

情報システム試験

令和２年度前期 情報検定

＜実施　令和2年9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32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．試験後にお知らせする合否結果（合否通知），および合格者に交付する「合格証・認定証」はすべて，Ｗｅｂページ（ＰＣ，モバイル）での認証によるディジタル「合否通知」，ディジタル「合格証・認定証」に移行しました。
①団体宛にはこれまでと同様に合否結果一覧ほか，試験結果資料一式を送付します。
②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～23 ページ
・表計算の問題	24 ページ～28 ページ
・アセンブラの問題	30 ページ～32 ページ

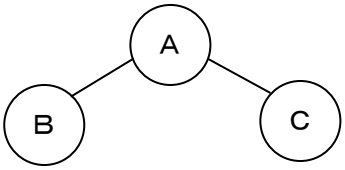
必須問題

問題１　次の２分木に関する記述を読み，各設問に答えよ。

２分木を走査する考え方に深さ優先探索がある。深さ優先探索は，ノードを縦（深さ）方向に走査し，行き止まりになったら,後戻りして別のノードを縦方向に走査する。走査する順序により，先行順，中間順，後行順次の３つに分けられる。

＜設問１＞　次の２分木に関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。

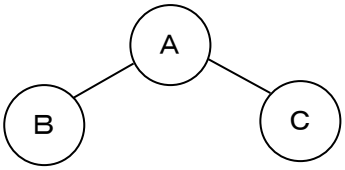
１．先行順(前順，行きがけ順とも言う)
ノード → 左部分木 → 右部分木の順に走査する。



Ａ→Ｂ→Ｃの順に走査

図１　先行順の走査

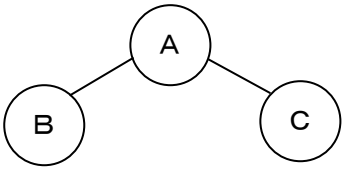
２．中間順(間順，通りがけ順とも言う)
左部分木 → ノード → 右部分木の順に走査する。



Ｂ→Ａ→Ｃの順に走査

図２　中間順の走査

３．後行順(後順，帰りがけ順とも言う)
左部分木 → 右部分木 → ノードの順に走査する。



Ｂ→Ｃ→Ａの順に走査

図３　後行順の走査

図４の２分木を走査する場合を考える。３つの走査方法で参照される文字の並びは，先行順では［ (1) ］，中間順では［ (2) ］，後行順では［ (3) ］となる。

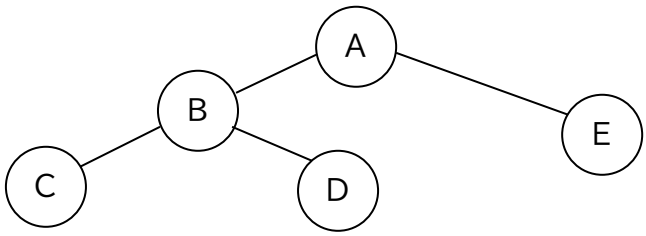


図４　２分木

(1) ～ (3) の解答群

ア．ＡＢＣＤＥ　　イ．ＣＢＤＡＥ　　ウ．ＣＤＢＥＡ　　エ．ＥＡＢＣＤ

＜設問２＞　次の逆ポーランド記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。

数式を逆ポーランド記法で表現するには演算子の優先順位をもとに２分木を作成し，後行順にたどる。演算子および括弧の優先順位は下表のとおりとする。

表　演算子および括弧の優先順位

演算子及び括弧	優先順位
括弧	高
×，÷	↕
＋，－	低

例えば，式「ａ＋ｂ×ｃ－ｄ」は次のように逆ポーランド記法で表記できる。

①　式中の優先度の低い演算子を中心に２分割する。優先度が同じ演算子がある場合は最右端の演算子を中心に２分割する。

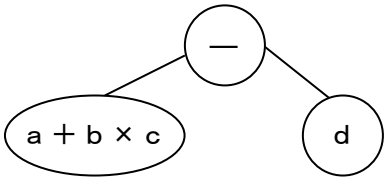


図５　２分木

② 分割されたノードをさらに分割する。

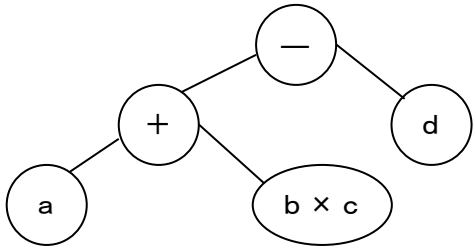


図６　２分木

③ さらに分割を繰り返す。

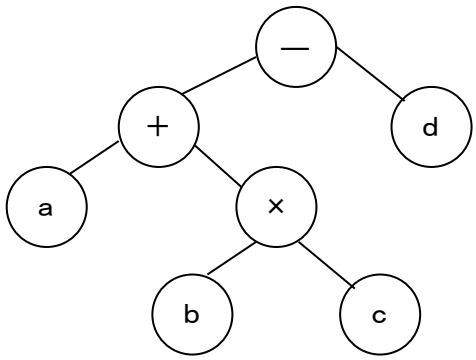


図７　２分木

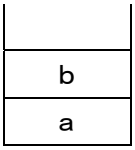
④ ２分木を後行順でたどると［　(4)　］となり，逆ポーランド記法で表記できる。

逆ポーランド記法で表現された式をコンピュータで計算するときはスタックを用いる。例えば，逆ポーランド記法で「ａ ｂ ÷ ｃ ｄ ＋ ×」と表記できる式を計算する手順は次のようになる。

式の左端から順に項を読み込み次の操作を行う。

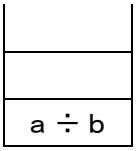
① 変数ならスタックに入れる。

スタック



② 演算子ならスタックから２つ取り出し，下の要素から上の要素を演算し，結果をスタックに入れる。

スタック



③　上記の操作を式中の項がなくなるまで繰り返す。

逆ポーランド記法で表記された式「ａ ｂ ÷ ｃ ｄ ＋ ×」にａ＝６，ｂ＝２，ｃ＝５，ｄ＝２を代入すると計算結果は［　(5)　］となる。

(4) の解答群

ア．ａ ｂ ｃ × ＋ ｄ －　　イ．ｂ ｃ ａ ＋ × ｄ －

ウ．ｃ ｂ ａ × ＋ ｄ －　　エ．ｄ － ａ ｂ ｃ × ＋

(5) の解答群

ア．11　　イ．18　　ウ．21　　エ．45

問題２　次のデータ探索に関する記述を読み，各設問に答えよ。

配列に格納されたデータから目的とするデータを見つけ出す探索法には，線形探索，二分探索などがある。

＜設問１＞　次の線形探索の説明を読み，記述中の［　　］に入れるべき適切な字句を解答群から選べ。

［線形探索の説明］

図１に示すように，DAT[0]から DAT[9]まで添字を１ずつ加算しながら探索し，目的とするデータを見つけた場合は配列の添字を ANS に格納して終了する。DAT[9]まで比較しても見つからない場合は－1 を ANS に格納する。なお，データは DAT[0]～DAT[9]にランダムに格納済であり，目的とするデータもＸに格納済である。

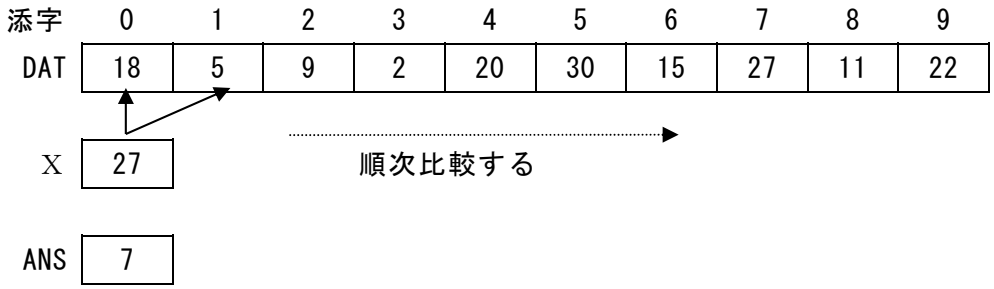


図１　線形探索の例

Ｘと配列要素との比較はそれぞれの要素に格納されているデータにより異なるが，最小で［　(1)　］回，最大で［　(2)　］回行われる。また，要素数をＮとした場合配列中のデータが同じ確率で探索されるとすると平均比較回数は一般的には［　(3)　］回となる。

(1)，(2) の解答群

ア．１　　イ．３　　ウ．５　　エ．10

(3) の解答群

ア．Ｎ　　イ．　Ｎ÷２　　ウ．Ｎ÷４

＜設問２＞　次の二分探索の説明を読み，記述中の［　　］に入れるべき適切な字句を解答群から選べ。

［二分探索法の説明］

二分探索法は，整列済みの一次元配列に対して行われる探索手法であり，次の①～④の手順で行う。なお，配列の大きさはN，データは昇順に DAT[0]～DAT[Ｎ－1]，目的とするデータはＸにそれぞれ格納済である。

① 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。なお，初期値は，Ｌ＝０，Ｈ＝Ｎ－１である。

② 探索範囲の中央要素となる DAT[ｋ]と比較する。ただし，ｋ＝(Ｌ＋Ｈ)÷２とし，小数点以下は切り捨てる。

③-ａ DAT[ｋ]＝Ｘなら，見つかったので ANS に添え字を格納し終了する。

③-ｂ DAT[ｋ]＜Ｘなら，Ｌ＝ｋ＋１とし，DAT[ｋ]より大きい値が格納されている範囲を次の探索範囲とする。

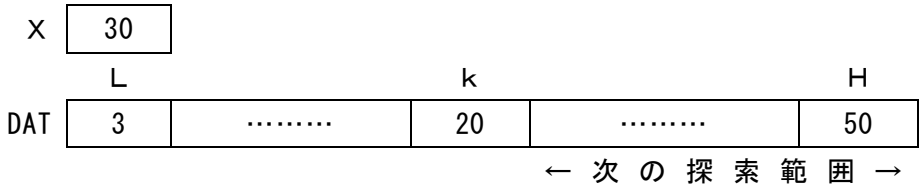


図２　比較例１

③-ｃ DAT[ｋ]＞Ｘなら，Ｈ＝ｋ－１とし，DAT[ｋ]より小さい値が格納されている範囲を次の探索範囲とする。

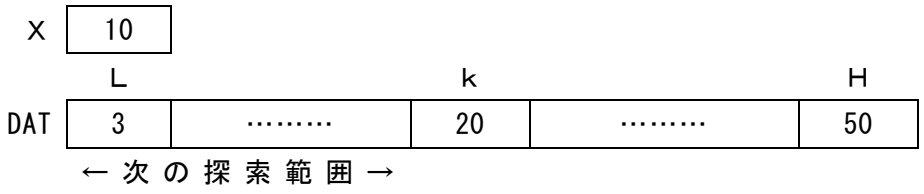


図３　比較例２

④ データが見つかるかＬ＞Ｈとなるまで，②～③を繰り返す。Ｌ＞Ｈの場合は，目的とするデータＸと同じ値が配列 DAT に存在しないことが確定するので ANS に－1 を格納し終了する。

Ｘと配列要素との比較はそれぞれに格納されているデータにより異なるが，一般的に二分探索の最大比較回数は，$2^m > N$を満たす最小のｍとなり，Ｎ＝10 の場合は［　(4)　］回である。また，平均比較回数は［　(5)　］となる。

(4) の解答群

ア．２　　イ．４　　ウ．８

(5) の解答群

ア．最大比較回数－1　　イ．最大比較回数と同じ　　ウ．最大比較回数＋1

＜設問３＞　次の説明を読み，記述中の［　　］に入れるべき適切な字句を解答群から選べ。

［データ格納の説明］

10 個のデータを次の手順で DAT[0]～DAT[9]に格納する。

① データを格納する添字 k を(データの値) mod 10 で求める。mod は余りを表し，ここではデータの値を 10 で割った余りである。

②-a　DAT[k]が空であればそのまま格納する。

②-b　DAT[k]にデータが格納済みの場合は，k に１を加えて②-a に戻るが，k が 0～9 の間に求まるようにするには，加算後の k に対して［(6)］の計算をする。

(6) の解答群

ア．k－10　　イ．k÷10　　ウ．k mod 10

問題３　次の再帰呼び出しに関する記述を読み，各設問に答えよ。

関数において自身を呼び出すことを再帰呼び出しと呼ぶ。再帰呼び出しを利用することで処理を単純に記述できる場合がある。

なお，再帰呼び出しを利用する場合は必ず処理を終える構造にする必要がある。例えば図１の流れ図では，関数Ｆはｘが１以下であれば関数Ｆを呼び出さずに処理を終えている。また，関数を呼び出す時にスタック領域にその時点のデータを記憶するため，再起呼び出しを多用するとスタック領域があふれてしまうので注意する必要がある。

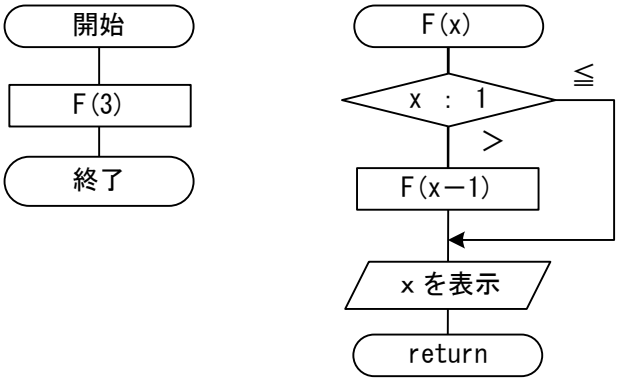


図１　関数Ｆを再帰呼び出しする

＜設問１＞　次の再帰呼び出しに関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。

図１の流れ図を検証する。関数Ｆはデータを変数ｘで受け取るが，ｘが１より大きい時は関数Ｆを呼び出す。関数Ｆの動作を追跡すると次のようになる。

① 関数Ｆが最初に呼び出された時は引数に３が与えられているため，変数ｘには３が格納される。よって，ｘ－１を引数として関数Ｆを呼び出す。

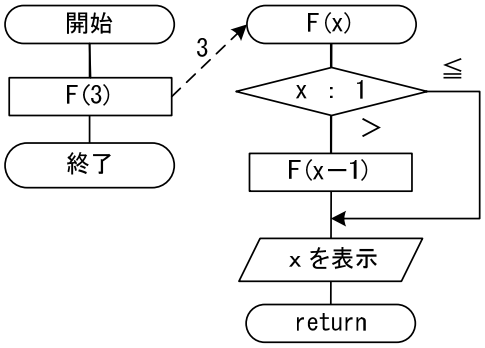


図２　最初の関数Ｆの呼び出し

② ２回目の関数Ｆの呼び出しでは引数に２が与えられているため，変数ｘには２が格納される。よって，ｘ－１を引数として関数Ｆを呼び出す。

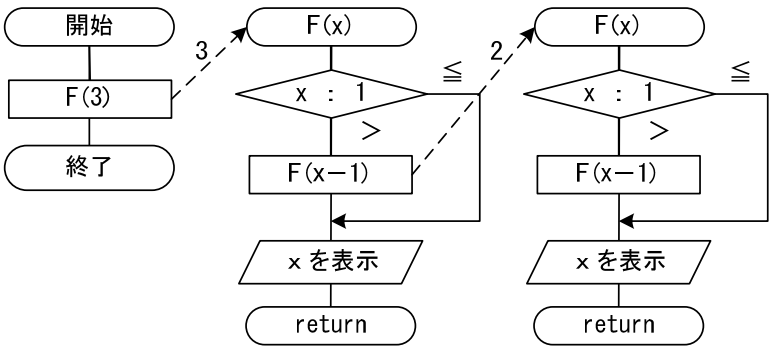


図３　２回目の関数Ｆ呼び出し

③ ３回目の関数Ｆの呼び出しでは引数に１が与えられているため，変数ｘには１が格納される。よってｘを表示して復帰する。この時表示される値は［　(1)　］である。

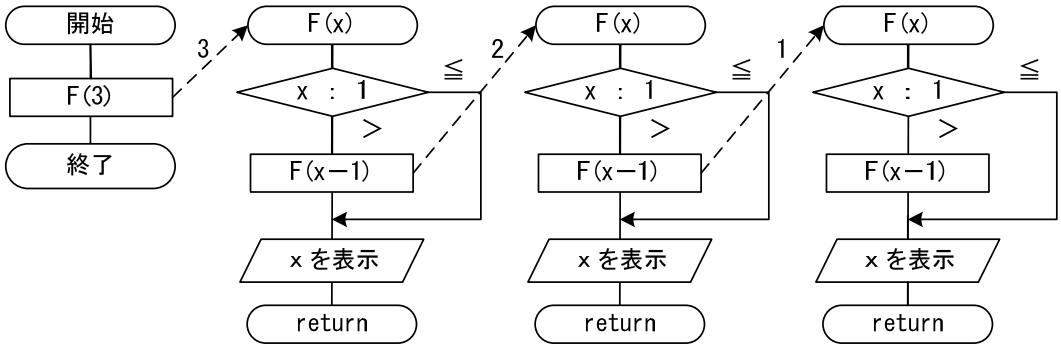


図４　３回目の関数Ｆ呼び出し

④ ３回目の関数Ｆの呼び出しから復帰する点は，３回目に関数Ｆを呼び出した次の処理であるから，ｘを表示して復帰する。この時表示される値は［　(2)　］である。

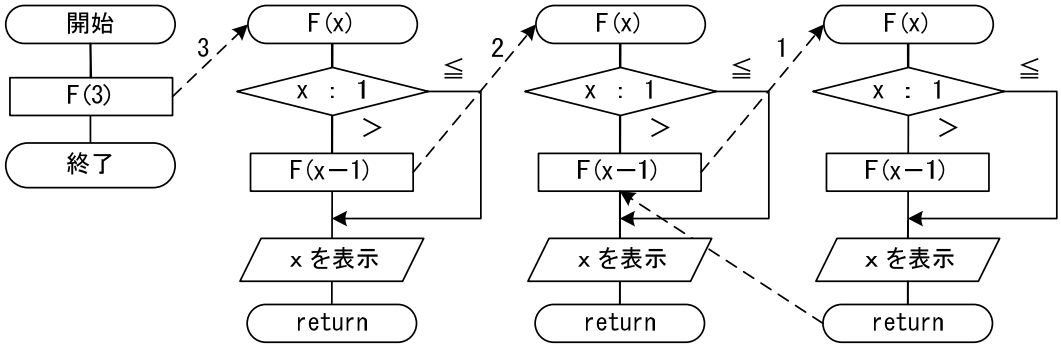


図５　３回目の呼び出しからの復帰

⑤ ２回目の関数Ｆの呼び出しから復帰する点は，２回目に関数Ｆを呼び出した次の処理であるから，ｘを表示して復帰する。この時表示される値は［　(3)　］である。

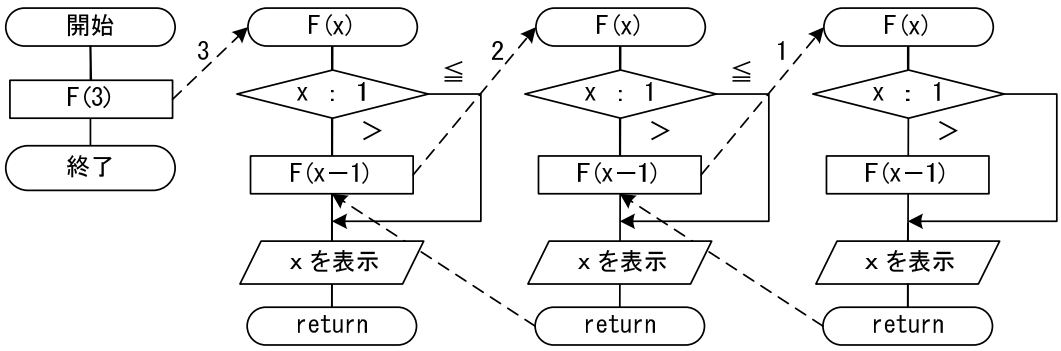


図６　２回目の呼び出しからの復帰

このようにして最初に関数Ｆを呼び出した処理の次に復帰して終了する。

(1) ～ (3) の解答群

ア．０　　イ．１　　ウ．２　　エ．３

＜設問２＞　次の漸化式に関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。

自然数を用いた数列を漸化式で表現したものは再帰呼び出しで処理することができる。例えば，１からｎまでの整数値の和は 1＋2＋3＋…＋ｎ であるが，漸化式で次のように表現できる。

$$\begin{cases} n = 1 & \cdots \quad F_1 = 1 \\ n > 1 & \cdots \quad F_n = F_{n-1} + n \end{cases}$$

ただし，ｎ≧１

この漸化式を流れ図で表現したものが図７である。

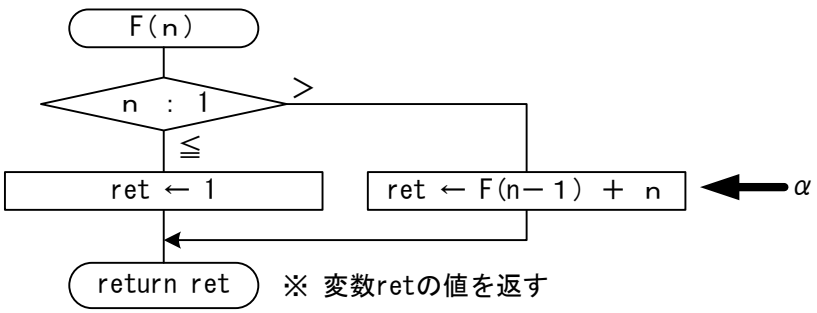


図７　１からｎまでの和を求める処理

図７のα部分を実行する時，まず関数Ｆの呼び出しが行われ，ｎとの加算が行われるのは関数Ｆの呼び出しから復帰した時点になる。ここで，図７の流れ図を「F(3)」として呼び出した場合，流れ図中α部分の加算が最初に実行される時の「F(n－1)」の返却値は［　(4)　］でありｎの値は［　(5)　］である。

(4)，(5) の解答群

ア．０　　イ．１　　ウ．２　　エ．３

＜設問３＞　次のフィボナッチ数列に関する記述および流れ図中の［　　］に入れるべき適切な字句を解答群から選べ。

フィボナッチ数列は次の漸化式で表すことができる。

$$\begin{cases} F_0 = 0 \\ F_1 = 1 \\ F_{n+2} = F_n + F_{n+1} \end{cases}$$

ただし，$n \geqq 0$

これにより，フィボナッチ数列の第５項（F_5）は［　(6)　］であり，第７項（F_7）は［　(7)　］であることがわかる。

図８は，フィボナッチ数列の第ｎ項を返却する流れ図である。

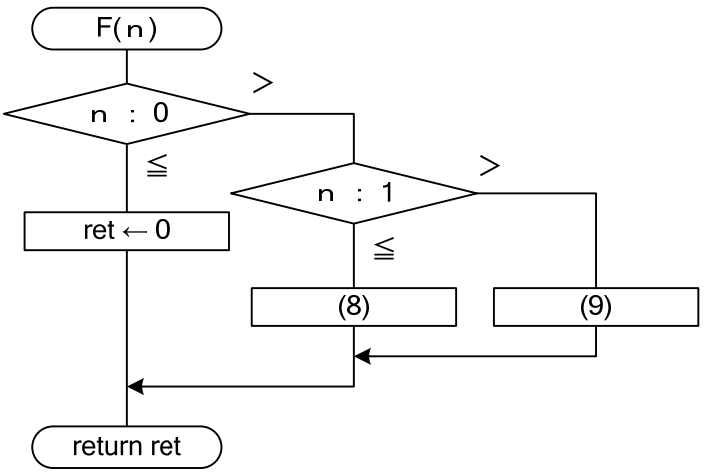


図８　フィボナッチ数列の第ｎ項を求める処理

(6)，(7) の解答群

ア．5　　イ．8　　ウ．13　　エ．21

(8) ～ (9) の解答群

ア. ret ← 0
イ. ret ← 1
ウ. ret ← F(0)
エ. ret ← F(1)
オ. ret ← F(n)
カ. ret ← F(n＋1) ＋ F(n＋2)
キ. ret ← F(n－1) ＋ F(n－2)

問題４　次の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］

１次元配列 text(長さ len_t)に文字列が格納されている。この文字列の先頭から配列 s_text（長さ len_s）に格納されている文字列の出現回数を調べて返す関数 Str_count である。図１の文字列では返却値が 2 となる。なお，各配列の添字は 0 から始まる。

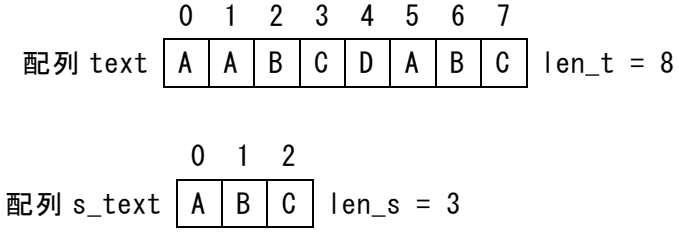


図１　文字列

［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。

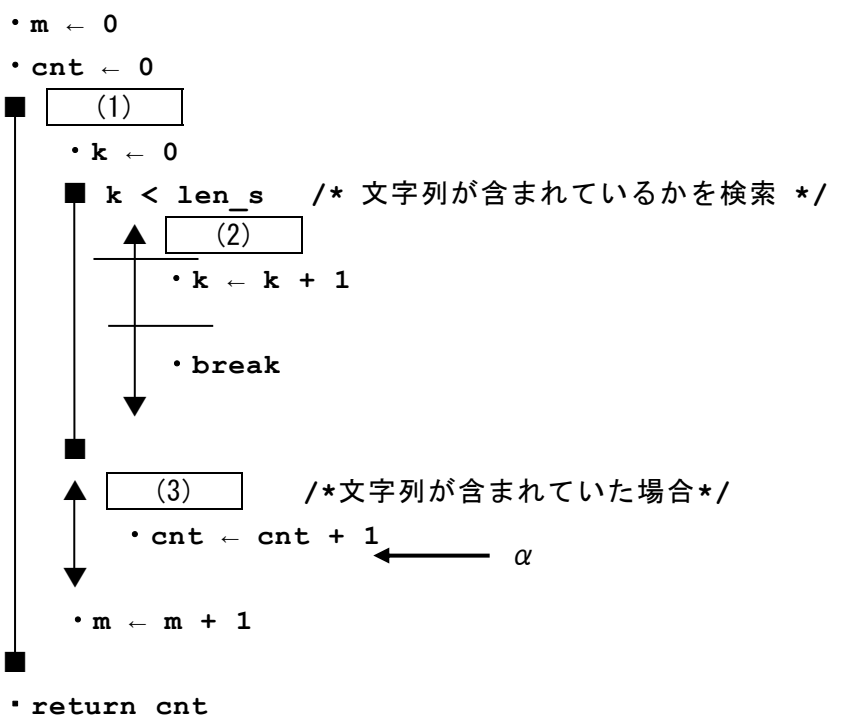
［演算子と優先順位］

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。％演算子は剰余算を表す。

［プログラム］

○Str_count(文字型配列：text[], s_text[], 整数型：len_t, len_s)
○整数型：k, m, cnt



＜設問１＞　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。

(1) の解答群

ア．m <= len_t - len_s　　イ．m <= len_t - len_s + 1
ウ．m > len_t - len_s　　エ．m > len_t - len_s + 1

(2) の解答群

ア．text[k] = s_text[k]　　イ．text[m] = s_text[k]
ウ．text[m+k] = s_text[k]　　エ．text[m+k] = s_text[m+k]

(3) の解答群

ア．k >= len_s　　イ．k > len_s
ウ．k < len_s　　エ．k <= len_s

＜設問２＞　次のプログラムの修正に関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。

関数 Str_count は，図 2 の文字列では返却値が 2 となるところ［ (4) ］となってしまう。それを解決するためには，探索文字列が見つかった場合に変数 m を文字数(len_s)に合わせて増加させると良い。そこでプログラム中のα部分に［ (5) ］を追加する。

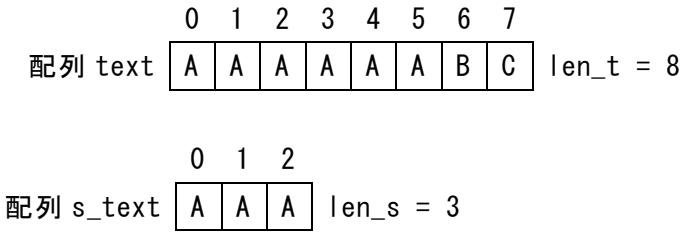


図２　文字列

(4) の解答群

ア．3　　イ．4　　ウ．5　　エ．6

(5) の解答群

ア．**m ← m + len_s + 1**　　イ．**m ← m + len_s**

ウ．**m ← m + len_s - 1**　　エ．**m ← len_s - 1**

＜ 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

次のハミング符号に関する記述を読み，各設問に答えよ。

［ハミング符号について］

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(7,4)ハミング符号（以下，ハミング符号）を扱うものとする。これは，2 ビットまでの誤り検出と 1 ビットの誤り訂正ができる。

なお，ここで扱うデータビットと検査用ビットの配置，およびビット番号は，図 1 のとおりである。

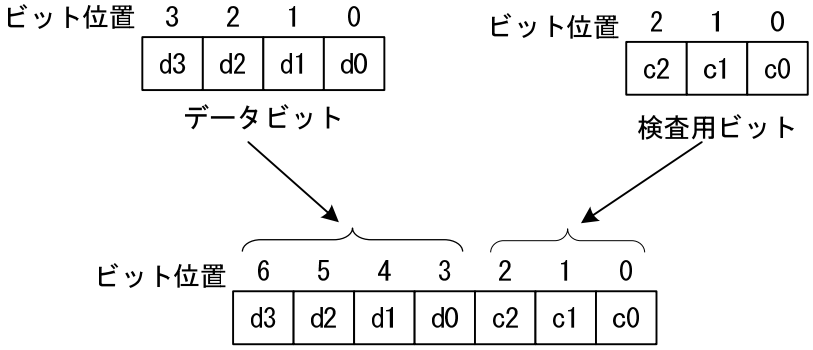


図 1　(7,4)ハミング符号の形式

［検査用符号の計算方法について］

検査用ビット（図 1 の c0～c2）を次のように計算する。

c0 … d0，d1，d2 の 3 ビットによる偶数パリティ

c1 … d0，d1，d3 の 3 ビットによる偶数パリティ

c2 … d1，d2，d3 の 3 ビットによる偶数パリティ

［誤り検出について］

ハミング符号から次の計算を行う。

s0 … d0，d1，d2，c0 の 4 ビットによる偶数パリティ

s1 … d0，d1，d3，c1 の 4 ビットによる偶数パリティ

s2 … d1，d2，d3，c2 の 4 ビットによる偶数パリティ

s0～s2 を図 2 のようなビット列で表した値（シンドロームと呼ぶ）が 0 以外であれば誤りが発生したと判断する。

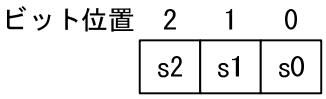


図 2　シンドローム

［誤りの訂正について］

シンドロームの値から誤りが発生したビットを次の表のように判断する。該当位置のビットを反転することで誤りを訂正する。

表　シンドロームと誤りのビット

シンドローム			誤りが発生したビット
s2	s1	s0	
0	0	1	c0
0	1	0	c1
0	1	1	d0
1	0	0	c2
1	0	1	d2
1	1	0	d3
1	1	1	d1

＜設問１＞　次のハミング符号に関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。なお，誤りが発生するビット数は１以下とする。

ハミング符号のビット列が 0010100 の場合を検証する。図１に合わせてビットの値を振り分けたものが図３である。

d3	d2	d1	d0	c2	c1	c0
0	0	1	0	1	0	0

図３　ビットの値を振り分ける

シンドロームを計算すると，s0～s2 は次のように計算できる。

s0 ＝ （d0 d1 d2 c0）の偶数パリティ ＝ （0 1 0 0）の偶数パリティ ＝ 1

s1 ＝ （d0 d1 d3 c1）の偶数パリティ ＝ （0 1 0 0）の偶数パリティ ＝ 1

s2 ＝ （d1 d2 d3 c2）の偶数パリティ ＝ （1 0 0 1）の偶数パリティ ＝ 0

よって，シンドロームのビット列は 011 となり，［　(1)　］のビットが誤っていることがわかる。同様に，ハミング符号のビット列が 0000001 の場合は［　(2)　］のビットが，0111111 の場合は［　(3)　］のビットが誤っていることがわかる。

(1) ～ (3) の解答群

ア．c0　　イ．c1　　ウ．c2　　エ．d0

オ．d1　　カ．d2　　キ．d3

＜設問２＞　次の関数の説明を読み，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。なお，ビット位置は右端から 0,1,2,…と数える。

［関数の説明］

makeHammingCode 関数

引　数：num(整数値)

機　能：引数で受け取った整数値の下位４ビットをデータビットとして図１の形式に従ったハミング符号を生成する。

戻り値：ハミング符号

getBit 関数

引　数：data(整数値)，n(整数値)

機　能：引数で受け取った data の第 n ビットの値を取り出して右端のビットに設定する。

戻り値：取り出したビットを右端ビット(第 0 ビット)に，他のビットは 0 にした値

［プログラム］

```
int makeHammingCode(int num) {
    int d[4], c0, c1, c2, i;
    for(i=0; i<4; i++) {        /* 1ビットずつ取り出す */
        d[i] = getBit(num, i);
    }
    c0 = (d[0] + d[1] + d[2]) % 2;  /* 検査ビットc0の計算 */
    c1 = (d[0] + d[1] + d[3]) % 2;  /* 検査ビットc1の計算 */
    c2 = (d[1] + d[2] + d[3]) % 2;  /* 検査ビットc2の計算 */
    return [   (4)   ];  /* データのビット列と検査ビットを合わせて返却 */
}

int getBit(int data, int n) {
    return [   (5)   ];
}
```

(4) の解答群

ア. **(num & 0xf) + c2 + c1 + c0**

イ. **(num & 0xf) * 4 + c2 + c1 + c0**

ウ. **((num & 0xf) << 3) + (c2 << 2) + (c1 << 1) + c0**

エ. **((num & 0xf) << 4) + (c2 << 3) + (c1 << 2) + (c0 << 1)**

(5) の解答群

ア. **0x1**　　イ. **0x1 >> n**

ウ. **(data >> n) & 0x1**　　エ. **data >> n**

＜設問３＞　次の関数の説明を読み，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。なお，関数 hammingCheck 中で呼び出す関数 getBit は設問２の関数 getBit と同じ関数である。また，ビット位置は右端から 0,1,2,…と数えるものとし，誤りが発生するビット数は 1 ビット以下とする。

［関数の説明］

getData 関数

引　数：hammingCode(整数値)

機　能：引数で受け取ったハミング符号(hammingCode)からデータビットを取り出すが，誤りがある場合はハミング符号の訂正を行ってからデータビットを取り出して右端に揃える。

戻り値：データビットを右端に揃えた値

hammingCheck 関数

引　数：hammingCode(整数値)

機　能：引数で受け取ったハミング符号のシンドロームを計算する。

戻り値：シンドローム

hammingCorrect 関数

引　数：hammingCode(整数値)，syndrome(整数値)

機　能：引数で受け取ったハミング符号(hammingCode)の誤りをシンドローム(syndrome)により訂正する。

戻り値：修正したハミング符号

［プログラム］

```
int getData(int hammingCode) {
    int p, syndrome, ret;
    ret = hammingCode;
    /* ハミング符号のシンドロームを取得 */
    syndrome = hammingCheck(hammingCode);
    if ( (6) ) {
        /* 誤り訂正を行った結果で ret を更新する */
        ret = hammingCorrect(hammingCode, syndrome);
    }
    return (7) ;
}

int hammingCheck(int hammingCode) {
    int i, s0, s1, s2;
    int h[7];
    /* 各ビットの値を取り出す */
    for(i=0; i<7; i++) {
        h[i] = getBit(hammingCode, i);
    }
    /* シンドロームの計算 */
    s0 = (h[5] + h[4] + h[3] + h[0]) % 2;
    s1 = (h[6] + h[4] + h[3] + h[1]) % 2;
    s2 = (h[6] + h[5] + h[4] + h[2]) % 2;
    /* シンドロームを返却 */
    return (8) ;
}

int hammingCorrect(int hammingCode, int syndrome) {
    /* pos[]はシンドロームによるビット位置を格納 */
    int pos[8] = {0, 0, 1, 3, 2, 5, 6, 4};
    /* 該当するビット位置を 0，他を 1 にしたマスクを作成 */
    int mask = 0x7f ^ (0x1 << pos[syndrome]);
    /* ハミング符号の該当するビットを右端に取り出し反転する */
    int bit = getBit(hammingCode, pos[syndrome]) ^ 0x1;
    /* ビット列を合成して返却 */
    return (hammingCode & mask) | (bit << pos[syndrome]);
}
```

(6) の解答群

ア. **syndrome > 0**　　イ. **syndrome == 0**

ウ. **syndrome % 2 == 0**　　エ. **syndrome % 3 == 0**

(7) の解答群

ア. **ret**　　イ. **ret >> 3**

ウ. **ret & 0x1**　　エ. **ret & 0xf**

(8) の解答群

ア. **s2 + s1 + s0**

イ. **(s2 + s1 + s0) * 2**

ウ. **(s2 << 2) + (s1 << 1) + s0;**

エ. **(s2 << 4) + (s1 << 2) + (s0 << 1)**

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。

COUNTIF 関数
　範囲に含まれるセルのうち，条件に一致するセルの個数を返す。
　書式：COUNTIF(範囲，条件)

IF 関数
　条件が真のときに真の場合，偽のときに偽の場合の計算結果や値を返す。
　書式：IF(条件，真の場合，偽の場合)

INDEX 関数
　範囲の中から行位置と列位置で指定したセルの値を返す。位置は 1 から始まる相対値である。
　書式：INDEX(範囲，行位置，列位置)

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)

SUM 関数
　指定した範囲に含まれる数値の合計値を返す。
　書式：SUM(範囲)

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うどん店では日々の売上を管理するため，表計算ソフトを利用している。Ｊうどん店で販売している商品の種類と価格は「価格表」ワークシートに入力しており，商品コードの昇順に並んでいる。なお，基本メニューの価格は，大きさにより金額が決定する。

	A	B	C	D	E	F	G	H
1	基本メニュー表				価格表			
2	種類	商品コード	商品名		種類	商品コード	商品名	価格
3	基本メニュー	S001	かけうどん(温)		大きさ	S101	並盛	300
4		S002	かけうどん(冷)			S102	大盛	410
5		S003	ざるうどん			S103	得盛	520
6		S004	釜あげうどん		ご飯	S201	いなり	110
7						S202	おむすび	140
8					トッピング	S301	牛すき肉	230
9						S302	とろろ	70
10						S303	温泉玉子	70
11						S304	生玉子	70
12						S305	油あげ	150
13						S306	野菜かき揚げ	140
14						S307	かしわ天	150
15						S308	えび天	160
16						S309	いか天	130
17						S310	さつまいも天	110
18						S311	かぼちゃ天	110
19						S312	ちくわ天	120

図１　「価格表」ワークシート

＜設問１＞　次の「販売データ」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式を解答群から選べ。

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	XX年XX月XX日　（水曜日）　売上表																						
2	商品名	基本メニュー				大きさ			ご飯		トッピング												合計金額
3	注文番号	かけうどん（温）	かけうどん（冷）	ざるうどん	釜あげうどん	並盛	大盛	得盛	いなり	おむすび	牛すき肉	とろろ	温泉玉子	生玉子	油あげ	野菜かき揚げ	かしわ天	えび天	いか天	さつまいも天	かぼちゃ天	ちくわ天	
4	価格	－	－	－	－	300	410	520	110	140	230	70	70	70	150	140	150	160	130	110	110	120	
5	1	1				1			1				1							1			590
6	2		1					1			1			1									820
7	3	1					1						1						2		1		850
8	4				1		1					1											480
9	5				1			1						1	1						2		960
:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:
203																							
204																							
205	総計	57	23	27	93	131	43	26	52	85	45	79	67	83	34	46	63	7	31	58	34	63	158,270

図２　「販売データ」ワークシート

・セル A2～A4 と W2 の項目名を入力し，セル B2～V3 は「価格表」ワークシートの種類と商品名から複写した。

・セル F1 に販売日を，セル K1 に曜日を表示するための式を入力し，セルの結合などを行って書式を整えた。

・４行目の価格は，「価格表」ワークシートから商品名で検索をし，表示する。セル F4 に次の式を入力し，セル G4～V4 まで複写した。

= (1)

・5 行目からは，１人のお客様から注文を受けた商品の数量を 1 行に入力する。

・ W 列は注文ごとの合計金額を表示する。セル W5 に注文ごとの合計金額を求める式を入力し，セル W6～W204 まで複写した。なお，W 列は 0 であれば表示しないような書式を設定している。

・ A 列の注文番号は，毎日 1 から始まる連番を表示する。セル A5 には 1 を入力し，セル A6 以降は 1 行前の合計金額(W 列)が 0 より大きい場合に表示する。なお，一日を通して 200 を超える注文はないものとし，204 行目まで領域を確保している。セル A6 に次の式を入力し，セル A7～A204 まで複写した。

= (2)

・ 205 行目の総計は，商品ごとの売上個数の合計と合計金額の合計を表示する。セル B205 に次の式を入力し，セル C205～W205 まで複写した。

= (3)

(1) の解答群

ア. INDEX(価格表!$F3:$H19, MATCH(F3, 価格表!$G3:$G19, 0), 3)
イ. INDEX(価格表!$F3:$H19, MATCH(F3, 価格表!G3:G19, 0), 3)
ウ. INDEX(価格表!F3:H19, MATCH(F3, 価格表!$G3:$G19, 0), 3)
エ. INDEX(価格表!$F3:H19, MATCH(F3, 価格表!G3:G19, 0), 3)

(2) の解答群

ア. IF(W$5＞0, A5－1, ″″)　　イ. IF(W5＞0, A5－1, ″″)
ウ. IF(W$5＞0, A5＋1, ″″)　　エ. IF(W5＞0, A5＋1, ″″)

(3) の解答群

ア. COUNT($B5:$B204)　　イ. COUNT(B5:B204)
ウ. SUM($B5:$B204)　　エ. SUM(B5:B204)

＜設問２＞　次の「年間売上一覧表」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式を解答群から選べ。

	A	B	C	D	E
1					単位：千円
2	月	前年	XX年	XX年累計	移動年計
3	1	4,432	4,829	4,829	55,950
4	2	4,583	4,382	9,211	55,749
5	3	4,294	4,493	13,704	55,948
6	4	4,728	4,682	18,386	55,902
:	:	:	:	:	
14	12	4,582	4,689	57,132	57,132
15	合計	55,553	57,132	－	－

図３　「年間売上一覧表」ワークシート

・セル A2～E2 に項目名を入力し，セル A3～A14 は月数を入力した。

・セル B3～C14 の売上金額は，「販売データ」ワークシートから月ごとの売上金額を集計した値である。

・D 列の XX 年累計は,C 列に集計した XX 年 1 月からの値を累計したものを表示する。セル D3 に次の式を入力した。

＝［　(4)　］

さらに，セル D4 に次の式を入力し，セル D5～D14 まで複写した。

＝［　(5)　］

・ E 列の移動年計は，当月を含め過去１年間の売上金額の累計を表示する。セル E3 に次の式を入力した。

＝［　(6)　］

さらに，セル E4 に次の式を入力し，セル E5～E14 まで複写した。

＝［　(7)　］

(4)，(5) の解答群

ア．B3　　イ．C3
ウ．D$3 ＋ B4　　エ．D$3 ＋ C4
オ．D3 ＋ B4　　カ．D3 ＋ C4

(6) の解答群

ア．SUM(B3:B14)　　イ．SUM(B3:B14) ＋ C3
ウ．SUM(B4:B14) ＋ C3　　エ．SUM(B5:B14) ＋ C4

(7) の解答群

ア．E$3 ＋ B4 ＋ C4　　イ．E$3 ＋ B4 － C4

ウ．E3 ＋ B4 － C4　　エ．E3 － B4 ＋ C4

＜設問３＞　次のグラフに関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。

図３の「年間売上一覧表」ワークシートからグラフを作成する。このグラフは，売上の季節変動など含めた長期的な傾向を見ることができ，［(8)］と呼ばれている。このグラフより，［(9)］状態であるといえる。

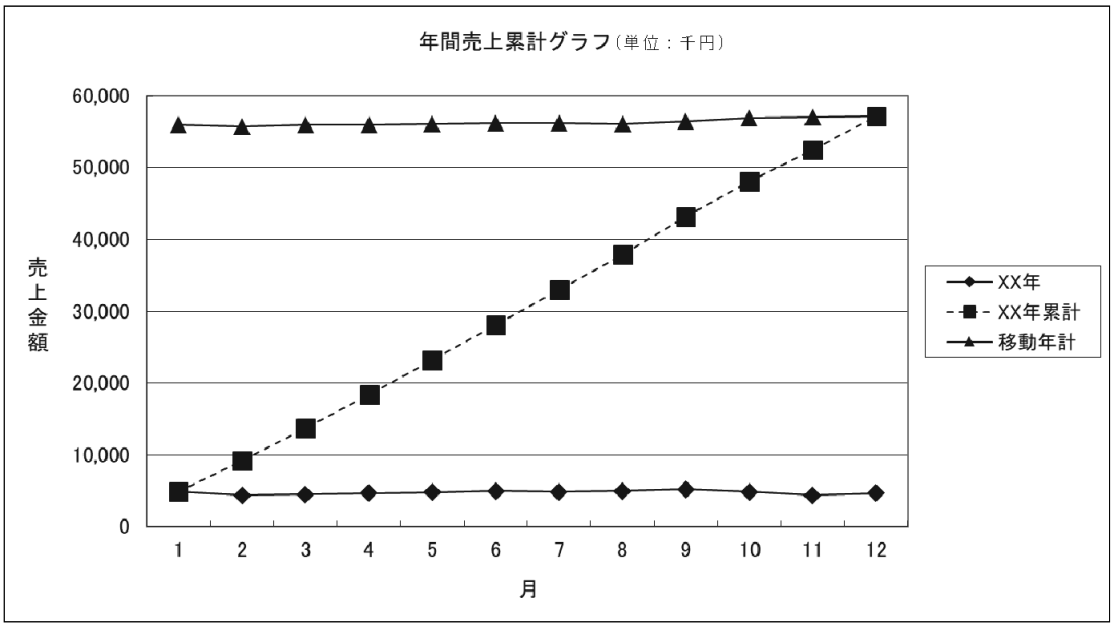


図４　年間売上一覧表

(8) の解答群

ア．Ｊチャート　　イ．Ｚグラフ

ウ．積み上げ面グラフ　　エ．レーダチャート

(9) の解答群

ア．現状維持で前年から当年にかけて特に変動がない

イ．前年と比較して売上が下がり，衰退傾向にある

ウ．前年と比較して売上が伸び，増加傾向にある

問題を読みやすくするために,
このページは空白にしてあります。

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]

DAT 番地から始まる５語の連続した領域に格納済みのデータにパリティビットを付加するプログラム ECC である。

パリティチェックとは，対象データ中の１のビットの数が偶数個または奇数個になるようにパリティビットで調整して記録するもので，受け取ったデータに対して１のビットの数が記録したときと同じ偶数個か奇数個かでビットの誤りを検査する方式である。偶数個で調整する場合を偶数パリティ方式，奇数個で調整する場合を奇数パリティ方式と呼ぶ。

プログラムでは偶数パリティ方式を採用し，図に示すように１語の符号ビットを除く 15 ビットに対するパリティビットを符号ビットに設定し(水平パリティ)，５語にまたがる同一ビット位置のパリティビットを PRTY 番地に設定する(垂直パリティ)。なお，DAT 番地以降に格納されたデータの符号ビットは 0 か 1 か不明である。

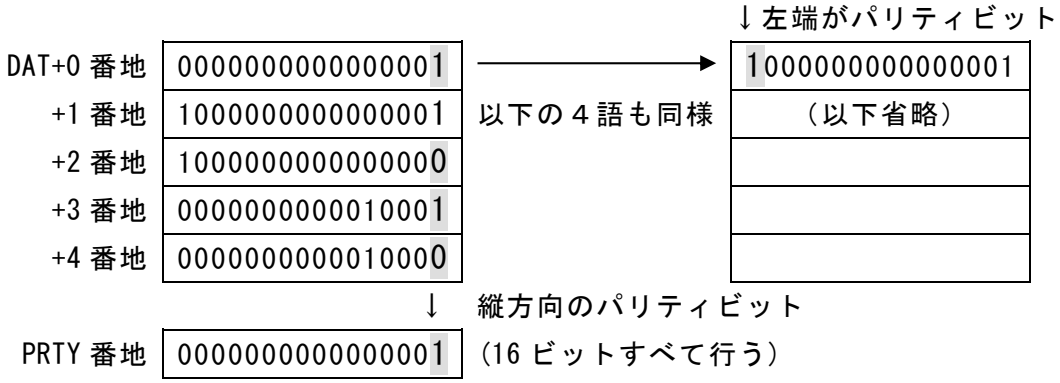


図　水平・垂直パリティの例

［プログラム］

行番号	ラベル	命令	オペランド	コメント
100	ECC	START		
110		RPUSH		
120		LD	GR1,=0	
130	LOOP1	LD	GR3,=0	;水平パリティビットの初期値
140		LD	GR0,DAT,GR1	
150		(1)		;符号ビットを0にする
160		ST	GR0,DAT,GR1	
170	LOOP2	SRL	GR0,1	;ビットチェック
180		JOV	CHG	
190		JZE	NEXT	
200		JUMP	LOOP2	
210	CHG	(2)		;水平パリティビットの調整
220		JUMP	LOOP2	
230	NEXT	XOR	GR3,DAT,GR1	;水平パリティビットの格納
240		ST	GR3,DAT,GR1	
250		(3)		;次の語の準備
260		CPA	GR1,=5	
270		JMI	LOOP1	
280		LD	GR3,=0	;垂直パリティビットの初期値
290		LD	GR1,=0	
300	LOOP3	XOR	GR3,DAT,GR1	;垂直パリティビットの調整
310		LAD	GR1,1,GR1	
320		CPA	GR1,=5	
330		(4)		
340		ST	GR3,PRTY	;垂直パリティビットの格納
350		RPOP		
360		RET		
370	DAT	DS	5	
380	PRTY	DS	1	
390		END		

＜設問１＞　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。

(1) の解答群

ア．AND　GR0,=#0000　　イ．AND　GR0,=#7FFF

ウ．AND　GR0,=#8000　　エ．AND　GR0,=#FFFF

(2) の解答群

ア．**XOR GR3,=#0000**　　イ．**XOR GR3,=#7FFF**

ウ．**XOR GR3,=#8000**　　エ．**XOR GR3,=#FFFF**

(3) の解答群

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

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

(4) の解答群

ア．**JMI LOOP3**　　イ．**JPL LOOP3**

ウ．**JUMP LOOP3**　　エ．**JZE LOOP3**

＜設問２＞　次のプログラムの変更に関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。

次のように，行番号 170 の命令を変更してもプログラムの処理結果に影響を与えない。 ただし，(1)～(4)は設問１の正しい答えが入っているものとする。

［変更後の命令］

行番号	ラベル	命令	オペランド
170	**LOOP2**	(5)	

(5) の解答群

ア．**AND GR0,=#0001**　　イ．**AND GR0,=#8000**

ウ．**SLA GR0,1**　　エ．**SLA GR0,2**

＜設問３＞　次のプログラムの変更に関する記述中の［　　］に入れるべき適切な字句を解答群から選べ。

次のように，パリティビットの初期値設定である行番号 130 と行番号 280 の２か所を変更すると，偶数パリティ方式から奇数パリティ方式に変更できる。

［変更後の命令］

行番号	ラベル	命令	オペランド
130	**LOOP1**	(6)	
280		(7)	

(6)，(7) の解答群

ア．**LD GR3,=#0000**　　イ．**LD GR3,=#0001**

ウ．**LD GR3,=#8000**　　エ．**LD GR3,=#FFFF**

＜メモ欄＞