次の問8は必須問題です。必ず解答してください。

問8　次のプログラムの説明及びプログラムを読んで，設問1～3に答えよ。

整数型関数 BitTest は，8 ビットのデータ中の指定したビット位置にあるビットの値を検査して，結果を返す。整数型関数 BitCount は，8 ビットのデータ中にある1のビットの個数を返す。

なお，本問において，演算子“&”，“|”は，二つの8ビット論理型データの対応するビット位置のビット同士について，それぞれ論理積，論理和を求め，8 ビット論理型で結果を得るものとする。また，"～"B という表記は，8 ビット論理型定数を表す。

〔プログラム1の説明〕

整数型関数 BitTest を，次のとおりに宣言する。

○整数型関数：BitTest（8 ビット論理型：Data，8 ビット論理型：Mask）

検査される 8 ビットのデータは入力用の引数 Data に，検査をするビット位置の情報は入力用の引数 Mask に，それぞれ格納されている。Mask 中のビットの値が 1 であるビット位置に対応した Data 中のビットを検査して，次の返却値を返す。ここで，Mask 中には1のビットが1個以上あるものとする。

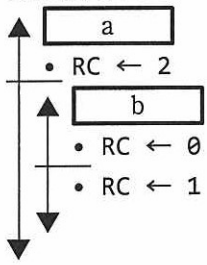
返却値　0：検査した全てのビットが0
　　1：検査したビット中に0と1が混在
　　2：検査した全てのビットが1

例えば，図1の例1では，Mask のビット番号 7～5 の 3 ビットが 1 であるので，Data のビット番号 7～5 の 3 ビットの値を検査し，0 と 1 が混在しているので返却値 1 を返す。例2では，Mask のビット番号 4 と 0 の 2 ビットが 1 であるので，Data のビット番号 4 と 0 の 2 ビットの値を検査し，どちらも 1 であるので返却値 2 を返す。

図1　BitTest の実行例

〔プログラム 1〕

```
○整数型関数：BitTest（8 ビット論理型：Data，8 ビット論理型：Mask）
○整数型：RC                    /* 返却値 */
```




```
• return RC                    /* RC を返却値として返す */
```

〔プログラム 2，3 の説明〕

整数型関数 BitCount を，次のとおりに宣言する。

○整数型関数：BitCount（8 ビット論理型：Data）

検査される 8 ビットのデータは入力用の引数 Data に格納されている。

このためのプログラムとして，基本的なアルゴリズムを用いたプログラム 2 と，処理効率を重視したプログラム 3 を作成した。

プログラム 2，3 中の各行には，ある処理系を想定して，プログラムの各行を 1 回実行するときの処理量（1, 2, …）を示してある。選択処理と繰返し処理の終端行の処理量は，それぞれの開始行の処理量に含まれるものとする。

なお，演算子“－”は，両オペランドを 8 ビット符号なし整数とみなして，減算を行うものとする。

〔プログラム 2〕

（処理量）

```
        ○整数型関数：BitCount（8 ビット論理型：Data）
        ○8 ビット論理型：Work
        ○整数型：Count，Loop
1       • Work ← Data
1       • Count ← 0
4       ■ Loop: 0，Loop ＜ 8，1
3         ▲ Work の最下位ビットが 1
1           • Count ← Count ＋ 1
          ▼
1         • Work を右へ 1 ビット論理シフトする
          ■
2       • return Count                 /* Count を返却値として返す */
```

〔プログラム3〕

(処理量)

```
        ○整数型関数：BitCount（8ビット論理型：Data)
        ○8ビット論理型：Work
        ○整数型：Count
1       • Work ← Data
1       • Count ← 0
2       ■ Work 中に1のビットがある
1         • Count ← Count + 1
3         • Work ← Work & (Work − 1)   ←――― α
          ■
2       • return Count                 /* Count を返却値として返す */
```

設問1　プログラム1中の［　　］に入れる正しい答えを，解答群の中から選べ。

解答群

ア　(Data & Mask) = "00000000"B　　イ　(Data & Mask) = Data

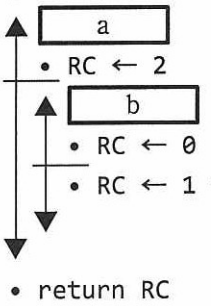
ウ　(Data & Mask) = Mask　　エ　(Data | Mask) = "00000000"B

オ　(Data | Mask) = Mask

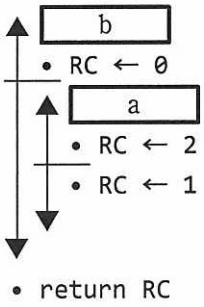
設問2　次の記述中の［　　］に入れる正しい答えを，解答群の中から選べ。

プログラム1は，Mask 中に1のビットが1個以上あることを前提としている。ここで，この前提を取り除いて，Mask 中の1のビットが0個の場合は返却値0を返すようにしたい。そのために，プログラム1の処理部分について，次の修正案①～③を考えた。ここで，修正案①は，プログラム1のままで何も変更しない。また，［ a ］と［ b ］には，設問1の正しい答えが入っているものとする。

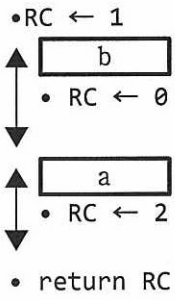
修正案①(変更なし)



修正案②



修正案③



これらの修正案のうち，正しく動作するのは［ c ］である。

解答群

ア　修正案①　　イ　修正案②　　ウ　修正案③

エ　修正案①及び②　　オ　修正案①及び③　　カ　修正案②及び③

設問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。

プログラム2，3の処理効率について考えてみる。表1にプログラム2，3の処理量の比較結果を示す。

表1　プログラム2，3の処理量の比較

	最小	最大
プログラム2	72	［ d ］
プログラム3	［ e ］	54

プログラム3では，**α**の行での変数 Work の更新において効率の良いアルゴリズムが使われている。例えば，プログラム3で引数 Data の内容が "01101010"B であったとき，繰返し処理において**α**の行の2回目の実行が終了した時点で変数 Work の内容は，"［ f ］"B になっている。このようなビット変換の処理によって，繰返し処理の繰返し回数は，検査されるデータ中の1のビットの個数と同じになる。

dに関する解答群

ア　80　　イ　88　　ウ　104　　エ　112

eに関する解答群

ア　6　　イ　10　　ウ　20　　エ　22

fに関する解答群

ア　00000011　　イ　00000110　　ウ　00001010

エ　01010000　　オ　01100000　　カ　10100000