プログラミング部門
大問例題

第 11 回 全国高等学校情報処理選手権 プログラミング部門
アルゴリズム大問例題　問題

※出題について
第 10 回大会プログラミング部門と同様に、プログラミングに関する小問 22 問と、例題 1・例題 2 のような複数設問をもつ大問を 1 題を出題いたします。
くわしい出題範囲については、HP にてご確認ください。

共通に使用される擬似言語の記述形式

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。

〔宣言，注釈及び処理〕

記述形式		説明
○		手続，変数などの名前，型などを宣言する。
/* **文** */		**文**に注釈を記述する。
処理	・**変数** ← **式**	**変数**に**式**の値を代入する。
	・**手続**（ **引数**，… ）	**手続**を呼び出し，**引数**を受け渡す。
	▲ **条件式** 　**処理** ▼	単岐選択処理を示す。 　**条件式**が真のときは**処理**を実行する。
	▲ **条件式** 　**処理 1** ―――― 　**処理 2** ▼	双岐選択処理を示す。 　**条件式**が真のときは**処理 1** を実行し，偽のときは**処理 2** を実行する。
	■ **条件式** 　**処理** ■	前判定繰返し処理を示す。 　**条件式**が真の間，**処理**を繰り返し実行する。
	■ 　**処理** ■ **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**を繰り返し実行する。
	■ **変数：初期値，条件式，** 　**増分処理** ■	繰返し処理を示す。 　開始時点で**変数**に**初期値**（定数又は式で与えられる）が格納され，**条件式**が真の間，**処理**を繰り返す。また，繰り返すごとに，**変数**に**増分**（定数又は式で与えられる）を加える。

〔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　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。

〔論理型の定数〕

true，false

例題 1 （H20 春-FE 午後問 2）

次のプログラムの説明及びプログラムを読んで，設問 1，2 に答えよ。

〔プログラムの説明〕

長さが Textlen の文字列 SourceText の中で，長さが Patlen の文字列 Pattern と一致する部分の文字列（以下，部分文字列という）の出現回数を数える関数 MatchCounter である。ここで，0＜Patlen≦Textlen である。

(1) MatchCounter の処理手順は，次のとおりである。

① 一致する部分文字列の出現回数を数える変数 Counter の値を 0 に初期化する。

② SourceText の比較開始位置を先頭から順に 1 文字ずつ後ろにずらしながら，その比較開始位置から始まる長さ Patlen の文字列と Pattern が一致するかどうかを調べ，一致したら出現回数 Counter の値に 1 を加算する。

③ Counter の値を返す。

(2) MatchCounter の引数と返却値の仕様を表に示す。文字列は文字型配列の各要素に 1 文字ずつ格納されている。また，各配列の添字は 0 から始まる。

表 MatchCounter の引数と返却値の仕様

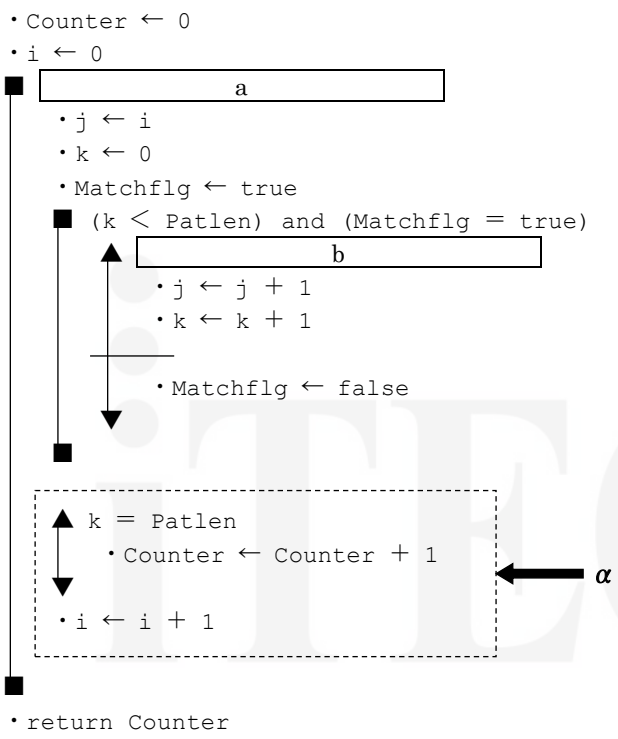
引数／返却値	データ型	入力／出力	意味
SourceText[]	文字型	入力	検索される文字列が格納されている 1 次元配列
Textlen	整数型	入力	検索される文字列の長さ
Pattern[]	文字型	入力	検索する文字列が格納されている 1 次元配列
Patlen	整数型	入力	検索する文字列の長さ
返却値	整数型	出力	SourceText の中で Pattern と一致した部分文字列の出現回数

〔プログラム〕

○**整数型**：MatchCounter(**文字型**：SourceText[], **整数型**：Textlen,
文字型：Pattern[], **整数型**：Patlen)

○**整数型**：Counter, i, j, k

○**論理型**：Matchflg



設問1　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。

a に関する解答群

ア　i ＋ Patlen ≦ Textlen　　イ　i ＋ Patlen ＜ Textlen

ウ　i ＋ Textlen ≦ Patlen　　エ　i ＋ Textlen ＜ Patlen

b に関する解答群

ア　SourceText[i] ＝ Pattern[k]

イ　SourceText[j] ＝ Pattern[k]

ウ　SourceText[k] ＝ Pattern[i]

エ　SourceText[k] ＝ Pattern[j]

設問2　Pattern と一致した部分文字列は以降の検索対象から外すように，このプログラムを変更する。

例えば，SourceText と Pattern が図に示す文字列であるとき，プログラムでは，比較開始位置を①，②，③，…と移動して，一致する部分文字列の出現回数を求めるので，下線の部分文字列と二重下線の部分文字列がともに数えられる。これに対して，一致した部分文字列を検索対象から外す場合は，比較開始位置を①，②，⑤，…と移動するので，二重下線の部分文字列は数えられない。

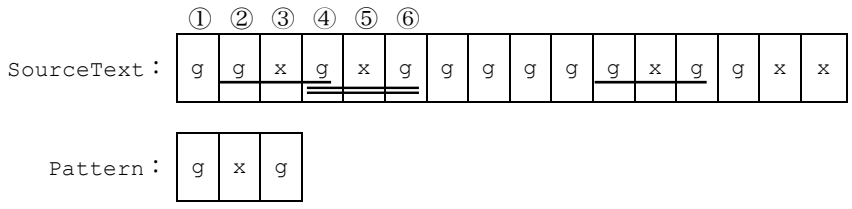
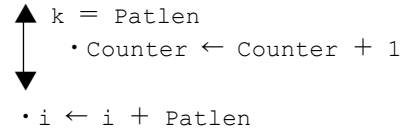


図　比較開始位置を変更する場合の例

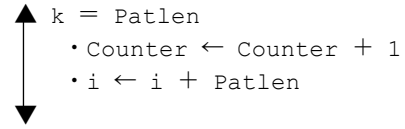
このとき，プログラム中の**α**部分の変更内容として正しい答えを，解答群の中から選べ。

解答群

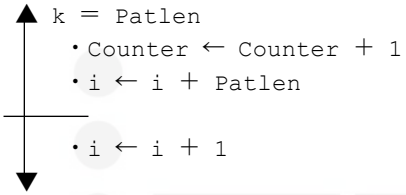
ア


k ＝ Patlen
・Counter ← Counter ＋ 1
・i ← i ＋ Patlen

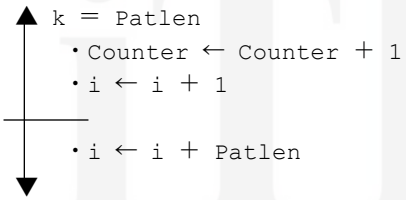
イ


k ＝ Patlen
・Counter ← Counter ＋ 1
・i ← i ＋ Patlen

ウ


k ＝ Patlen
・Counter ← Counter ＋ 1
・i ← i ＋ Patlen
・i ← i ＋ 1

エ


k ＝ Patlen
・Counter ← Counter ＋ 1
・i ← i ＋ 1
・i ← i ＋ Patlen

例題２

(H22 春-FE 午後問 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。

〔プログラムの説明〕

プログラム Sort は配列に格納された整数値のデータを再帰的に分割し，分割したデータの値の大小を比較しながら併合していくことでデータを昇順に整列するプログラムである。Sort は併合に副プログラム Merge を使用する。

(1) num 個（num≧1）のデータを配列 list に格納して Sort を呼び出すと，整列された結果が配列 list に返却される。

(2) Sort では，次の手順で配列 list に格納された整数値のデータを整列する。

① 配列 list に格納されているデータを，先頭から num÷2 個と残り num－num÷2 個とに分割して，二つの配列 slist1 と slist2 に格納し，それぞれの配列に対して再帰的に Sort を呼び出す。ここで，配列 slist1 と slist2 の大きさは省略されているが，必要な領域は確保されている。この再帰的な呼出しは，引数で渡される配列 list のデータの個数が 1 になると終了する。

② Merge を使用し，二つの配列 slist1 と slist2 を併合して一つの配列 list にする。

(3) Merge では，次の手順で，整列済の二つの配列 slist1 と slist2 を併合し，整列した一つの配列 list を作成する。

① 配列 slist1 又は slist2 のどちらか一方の要素がなくなるまで，次の②を繰り返す。

② 配列 slist1 と slist2 の要素を順に比較して，小さい方から順に配列 list に格納する。

③ 配列 slist1 又は slist2 の残った要素を配列 list に追加する。

(4) Sort と Merge の引数の仕様を表 1，2 に示す。配列の添字は 0 から始まる。

表 1　Sort の引数の仕様

引数名／返却値	データ型	入力／出力	意味
list[]	整数型	入力及び出力	データが格納されている 1 次元配列
num	整数型	入力	配列 list のデータの個数

表 2　Merge の引数の仕様

引数名／返却値	データ型	入力／出力	意味
slist1[]	整数型	入力	整列済のデータが格納されている 1 次元配列
num1	整数型	入力	配列 slist1 のデータの個数
slist2[]	整数型	入力	整列済のデータが格納されている 1 次元配列
num2	整数型	入力	配列 slist2 のデータの個数
list[]	整数型	出力	併合したデータを格納する 1 次元配列

次のデータを例にして，整列処理の流れを図に示す。

配列 list のデータ：5, 7, 4, 2, 3, 8, 1

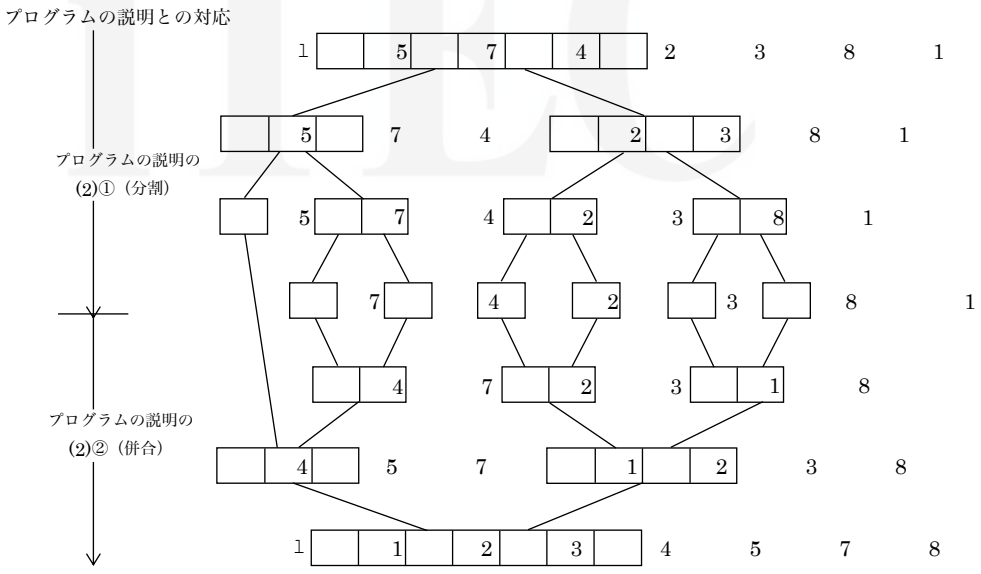


図　整列処理の流れ

〔プログラム〕

```
/* プログラム Sort */
○Sort(整数型:list[], 整数型:num)
○整数型:i, num1, num2
○整数型:slist1[], slist2[]          /* 配列の宣言 */
        a
  ・ num1 ← num ÷ 2                 /* slist1 の要素数計算 */
  ・ num2 ← num − num1              /* slist2 の要素数計算 */
  ■ i:0, i ＜ num1, 1
    ・ slist1[i] ← list[i]
  ■

  ■ i:0, i ＜ num2, 1
    ・ slist2[i] ←  [ b ]
  ■

  ・ Sort(slist1, num1)
  ・ Sort(slist2, num2)
  ・ Merge(slist1, num1, slist2, num2, list)
                                    ← α
/* プログラム Sort の終わり */

/* 副プログラム Merge */
○Merge(整数型:slist1[], 整数型:num1,
       整数型:slist2[], 整数型:num2,
       整数型:list[] )
○整数型:i, j
・ i ← 0
・ j ← 0
■ [ c ]
  ▲ slist1[i] ＜ slist2[j]
    ・ list[i＋j] ← slist1[i]
    ・ i ← i ＋ 1
  ―
    ・ list[i＋j] ← slist2[j]
    ・ j ← j ＋ 1
  ▼
■

■ (i ＜ num1) or (j ＜ num2)
  ▲ i ＜ num1
    ・ list[i＋j] ← slist1[i]
    ・ i ← i ＋ 1
  ―                                 ← β
    ・ list[i＋j] ← slist2[j]
    ・ j ← j ＋ 1
  ▼
■
/* 副プログラム Merge の終わり */
```

設問1　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。

a に関する解答群

ア　num ≧ 0　　イ　num ≧ 1

ウ　num ＞ 1　　エ　num ＞ 2

b に関する解答群

ア　list[i]　　イ　list[num＋i]

ウ　list[num1＋i]　　エ　list[num2＋i]

c に関する解答群

ア　(i ＜ num1) and (j ＜ num2)　　イ　(i ＜ num1) or (j ＜ num2)

ウ　(j ＜ num1) and (i ＜ num2)　　エ　(j ＜ num1) or (i ＜ num2)

オ　(i＋j) ＜ (num1＋num2)　　カ　(i＋j) ≦ (num1＋num2)

キ　(i＋j) ＞ (num1＋num2)　　ク　(i＋j) ≧ (num1＋num2)

設問2　最初に与えられた配列 list のデータが次の場合，プログラム Sort の**α**における配列 list の内容の移り変わりとして正しい答えを，解答群の中から選べ。

配列 list のデータ：3, 8, 2, 7, 5, 1

なお，解答群の“→”は，内容が左から右へ移り変わっていくことを示している。

解答群

ア　2 → 3 → 2,3 → 2,3,8 → 1 → 5 → 1,5 → 1,5,7 → 1,2,3,5,7,8

イ　3 → 8 → 3,8 → 2,3,8 → 7 → 5 → 7,5 → 1,5,7 → 1,2,3,5,7,8

ウ　2,8 → 2,3,8 → 1,5 → 1,5,7 → 1,2,3,5,7,8

エ　3,8 → 2,3,8 → 7,5 → 1,5,7 → 1,2,3,5,7,8

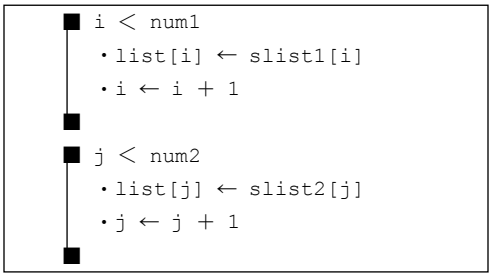
オ　2,3,8 → 1,5,7 → 1,2,3,5,7,8

カ　3,8,2 → 7,5,1 → 1,2,3,5,7,8

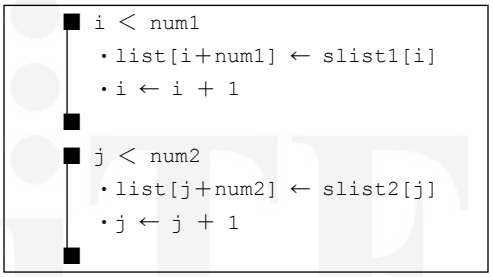
設問３　副プログラム Merge の**β**部分と同じ結果を得る処理として正しい答えを，解答群の中から選べ。

解答群

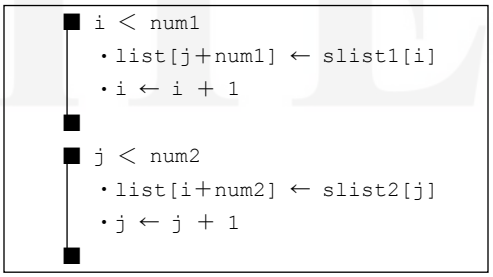
ア



イ



ウ



エ