平成１８年度　春期　ＦＥ　午後問題　C 言語

問６ 次の C 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〕

Helge von Koch が考案したコッホ曲線と呼ばれるフラクタル図形を描画する関数 KochCurve である。フラクタル図形は，図形の一部を拡大すると，再び，同様の図形が現れる自己相似性をもち，コッホ曲線はその代表的な図形である。

(1) 図１の ① ～ ③ にコッホ曲線の生成手順を示す。

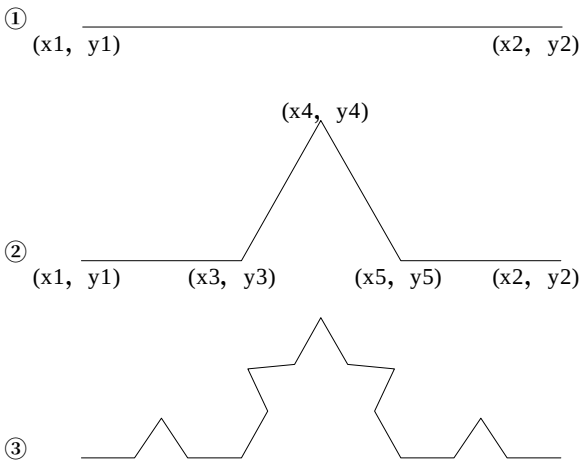

図 1　コッホ曲線の生成手順

① 始点の座標を（x1, y1），終点を（x2, y2）とする 2 点を結ぶ線分を引く。

② 線分を 3 等分し，中央の線分を一辺とする正三角形を始点から終点に向かって左側に作り，その頂点座標を求める。始点（x1, y1）から，正三角形の 3 頂点（x3, y3），（x4, y4），（x5, y5），そして終点（x2, y2）を順番に結んだ四つの線分を引く。

③ 得られた四つの線分それぞれを①の線分とみなし，②の操作を行う。

この操作を繰り返して得られるのがコッホ曲線である。

(2) 関数 DrawLine は，引数で指定された 2 点を結ぶ線分を描画する関数であり，既に用意されている。関数の仕様は，次のとおりである。

呼出し形式：

```
void DrawLine( int x1, int y1, int x2,
                                    int y2 );
```

引数：　x1 及び y1　線分の始点座標（x1, y1）
　　x2 及び y2　線分の終点座標（x2, y2）

返却値：　なし

(3) 関数 KochCurve は，引数で指定された 2 点（始点及び終点）の座標を基点にしてコッホ曲線を描画する関数であり，仕様は次のとおりである。

呼出し形式：

```
void KochCurve( int x1, int y1, int x2,
                          int y2, int dim );
```

引数：　x1 及び y1　コッホ曲線の始点座標（x1, y1）
　　x2 及び y2　コッホ曲線の終点座標（x2, y2）
　　dim　(1) の説明での繰返し回数
　　0 のとき図１の ① が描画される。

返却値：　なし

(4) このプログラムを実行すると，図１の ② に示すような図形が描画される。また，行番号 25 を次のとおりに変更して実行すると，③ に示すような図形が描画される。

```
25    KochCurve( 0, 180, 180, 180, 2 );
```

なお，座標軸は，原点（0, 0）から，右向きを x 軸の正方向，下向きを y 軸の正方向とする。

〔プログラム〕

（行番号）

```
 1  #include <stdio.h>
 2  #include <math.h>
 3  #define PI 3.141593

 4  void DrawLine( int, int, int, int );
 5  void KochCurve( int x1, int y1, int x2,
                           int y2, int dim )
 6  {
 7      int x3, y3, x4, y4, x5, y5;

 8      if ( [ a ] )
 9          DrawLine( x1, y1, x2, y2 );
10      else {
11          x3 = ( 2 * x1 + x2 ) / 3;
12          y3 = ( 2 * y1 + y2 ) / 3;
13          x5 = ( x1 + 2 * x2 ) / 3;
14          y5 = ( y1 + 2 * y2 ) / 3;
15          x4 = x3 + (x5 - x3) * cos(PI/3) +
                      (y5 - y3) * sin(PI/3);
16          y4 = y3 - (x5 - x3) * sin(PI/3) +
                      (y5 - y3) * cos(PI/3);
17          KochCurve( x1, y1, x3, y3,
                                [ b ] );
18          KochCurve( x3, y3, x4, y4,
                                [ b ] );
19          KochCurve( x4, y4, x5, y5,
                                [ b ] );
20          KochCurve( x5, y5, x2, y2,
                                [ b ] );
21      }
22  }
23  void main()
24  {
```

```
25      KochCurve( 0, 180, 180, 180, 1 );
26  }
```

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　dim <= 0　　イ　dim <= 1　　ウ　dim == 1

エ　dim != 0　　オ　dim != 1

b に関する解答群

ア　dim--　　イ　dim - 1　　ウ　dim

エ　dim + 1　　オ　dim++

設問２　図２の図形を描画するためには，関数 main の行番号 25 をどのように変更すればよいか。正しい答えを，解答群の中から選べ。

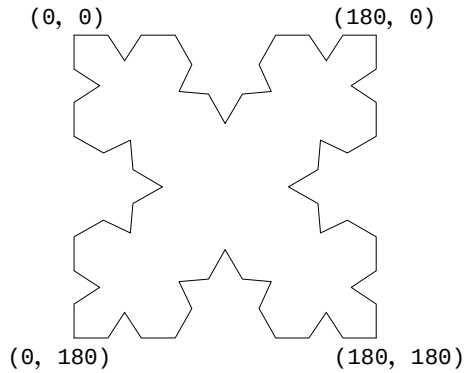

図２　描画したい図形

解答群

ア
```
KochCurve(   0,   0,   0, 180, 2 );
KochCurve(   0, 180, 180, 180, 2 );
KochCurve( 180, 180, 180,   0, 2 );
KochCurve( 180,   0,   0,   0, 2 );
```

イ
```
KochCurve(   0,   0, 180,   0, 2 );
KochCurve( 180,   0, 180, 180, 2 );
KochCurve( 180, 180,   0, 180, 2 );
KochCurve(   0,   0,   0, 180, 2 );
```

ウ
```
KochCurve(   0,   0, 180,   0, 2 );
KochCurve( 180,   0, 180, 180, 2 );
KochCurve( 180, 180,   0, 180, 2 );
KochCurve(   0, 180,   0,   0, 2 );
```

エ
```
KochCurve(   0,   0, 180,   0, 2 );
KochCurve( 180,   0, 180, 180, 2 );
KochCurve(   0,   0,   0, 180, 2 );
KochCurve(   0, 180, 180, 180, 2 );
```

オ
```
KochCurve( 180,   0, 180, 180, 2 );
KochCurve( 180, 180,   0, 180, 2 );
KochCurve(   0, 180,   0,   0, 2 );
KochCurve(   0,   0, 180,   0, 2 );
```

問 10　次の C 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〕

入力された英単語を検索し，訳語を表示するプログラムである。

(1)　データは，木構造で登録されている。英単語を構成するそれぞれの文字を構造体 LETTERで表現する。構造体 LETTER のメンバ c には英小文字の文字符号が，メンバ follow にはその文字に続く文字を示す構造体 LETTER へのポインタが，メンバ other にはその文字に代わる文字を示す構造体 LETTER へのポインタが格納されている。メンバ trans には，p_root が指す根から順にたどってできる英単語に対する訳語の文字列へのポインタが格納されている。メンバ follow, other, trans には，該当するものがなければ空ポインタ定数（NULL）が格納されている。

例えば，五つの英単語“man”,“main”,“mail”,“mailer”,“name”が登録されている場合の構造は，図１のとおりとなる。

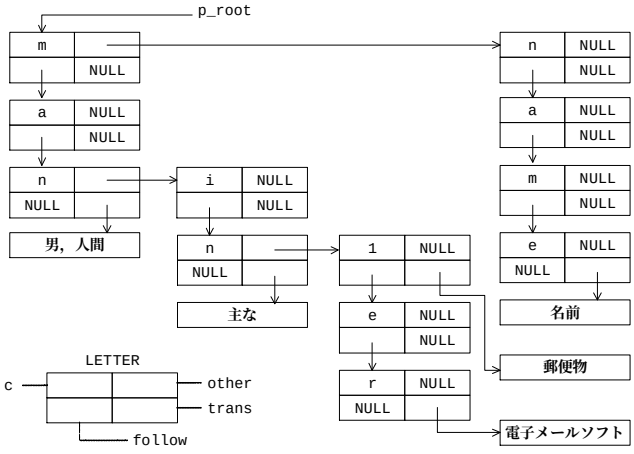

図１　英単語の登録例

(2)　ユーザインタフェースは，図２に示すように，入力された英単語を表示するための検索語入力領域と，訳語を表示するための訳語表示領域からなる（以後，検索語入力領域に表示されている文字列を，入力文字列と呼ぶ）。

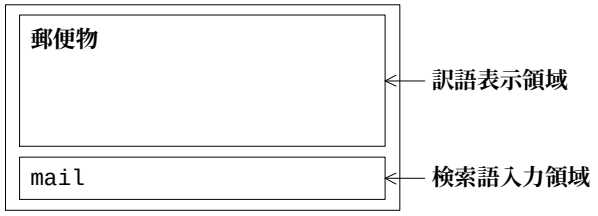

図２　ユーザインタフェースの例

(3)　入力文字列で始まる単語の候補がなくなった場合は，検索語入力の途中でも訳語表示領域上に“単語が見つかりません”を表示し，検索を終了する。

(4) ユーザインタフェースに関する次の関数が用意されているものとする。

void displayArea(char *s)
機能：訳語表示領域をクリアした後，文字列 s を表示する。

void clearField()
機能：検索語入力領域をクリアする。

void appendToField(char c)
機能：検索語入力領域に文字 c を追加表示する。

char inputChar()
機能：キーボードから 1 文字を読み込む。返却値は，入力された文字が英小文字の場合は文字符号，それ以外の場合は 0 となる。

〔プログラム〕

```
typedef struct letter {
   char c;          /* 英単語を構成する文字 */
   struct letter *follow;
             /* この文字に続く文字へのポインタ */
   struct letter *other;
        /* この文字に代わる別の文字へのポインタ */
   char *trans;    /* 訳語の文字列へのポインタ */
} LETTER;

void displayArea(char *);
void clearField();
void appendToField(char);
char inputChar();

LETTER *p_root;

void searchWord(){
   LETTER *p = p_root;
   char c;
   clearField();    /* 検索語入力領域のクリア */
   displayArea("");   /* 訳語表示領域のクリア */
   while(c = inputChar()){  /* 1 文字入力 */
      displayArea("");
      appendToField(c);
          /* 検索語入力領域に入力文字を追加 */
          /* 終端又は文字のいずれかと一致するまで
                                        検索 */
      while(( [ a ] ) && (p->c != c)){
         p = [ b ];
      }
      /* 文字と一致しなかった（入力文字列で始まる
                                  英単語がない）*/
      if( [ c ] ){
         displayArea("単語が見つかりません");
         return;
      }
α→
      /* 入力文字列と一致する英単語がある */
      if(p->trans != NULL){
         displayArea(p->trans);
         }
         p = [ d ];
      }
   }
}
```

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，c に関する解答群

ア　p != NULL
イ　p == NULL
ウ　p->follow != NULL
エ　p->follow == NULL
オ　p->other != NULL
カ　p->other == NULL
キ　p->trans != NULL
ク　p->trans == NULL

b，d に関する解答群

ア　p->follow　　イ　p->follow->other
ウ　p->other　　エ　p->other->follow

設問２　入力補完機能を追加することにした。例えば図１の登録例で，図３ に示すように，初めに“m”を入力した場合は，それに一意に続く“a”を補完して検索語入力領域に追加表示する。また，初めに“n”を入力した場合は，“ame”を追加表示する。関数 searchWord の修正は，次の表のとおりである。表中の [　　] に入れる正しい答えを，解答群の中から選べ。

m	⟶	ma
n	⟶	name

図３　補完例

処置	プログラムの変更内容

α の部分に追加	while((p->trans == NULL) &&
	([　　])){
	p = p->follow;
	appendToField(p->c);
	}

解答群

ア　p->follow != NULL
イ　p->follow == NULL
ウ　p->follow->other != NULL
エ　p->follow->other == NULL

オ　p->other != NULL

カ　p->other == NULL

平成１８年度　秋期　ＦＥ　午後問題　C言語

問６　次のCプログラムの説明及びプログラムを読んで，設問に答えよ。

〔プログラムの説明〕

プログラムが生成した各けたの数字が異なる４けたの目標数を，なるべく少ない回数の推測で当てるゲームGuessNumberである。回答者は，各けたが異なる４けたの推測数を入力し，当たらなかった場合，次の情報を得ることができる。

① 目標数に含まれていて，けたも一致している数字の個数（以下，Hit数という）

② 目標数に含まれているが，けたが一致していない数字の個数（以下，Blow数という）

例えば，目標数が“１６３２”で推測数が“７６１３”の場合，“６”はけたも一致しているのでHit数が1，“１”と“３”はけたが一致していないのでBlow数が２となる。

(1) 目標数はプログラムが自動的に生成する。生成した数字列はchar型の配列targetに格納される。推測数を文字列として標準入力から読み込み，char型の配列numに格納したうえで照合する。目標数と完全に一致するまで繰り返す。

(2) 次の関数が用意されている。

void createRandomNumber(char[] target)
機能：４けたの目標数を文字列として左のけたから順に配列targetに格納する。各けたの数字はすべて異なっている。

int isValidNumber(char[] num)
機能：文字列num中の各文字がすべて異なる数字のときTRUEを，それ以外のときFALSEを返す。

〔プログラム〕

```
#include <stdio.h>
#define DIGITS 4   /* けた数 */
#define TRUE  1
#define FALSE 0

void createRandomNumber(char[]);
int isValidNumber(char[]);
void GuessNumber();
int isMatch(char[], char[]);

void GuessNumber(){
   char target[DIGITS + 1]; /* 目標数の保存領域 */
   char num[DIGITS + 1];    /* 推測数の保存領域 */
   int count = 0;           /* 推測回数 */

createRandomNumber(target);  /* 目標数の生成 */
   do{
      printf("[%d回目] 各けたが異なる%dけたの数を
           入力してください:",++count, DIGITS);
      scanf("%4s", num);
      while( [ a ] ){
         printf("入力が正しくありません。再度入力
                                してください:");
         scanf("%4s", num);
      }
   }while( [ b ] );
}

int isMatch(char target[], char num[]){
   int i, j, numHit = 0, numBlow = 0;

   for(i = 0; i < DIGITS; i++){
      for(j = 0; j < DIGITS; j++){
         if( [ c ] ){
            if( [ d ] ){
               numHit++;
            } else {
               numBlow++;
            }
         }
      }
   }
   if( [ e ] ){
      printf("正解です。\n");
      return TRUE;
   }
   printf("%sは%dHit,%dBlowです。\n\n", num,
                              numHit, numBlow);
   return FALSE;
}
```

設問　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，b に関する解答群

ア　isMatch(target, num) == FALSE

イ　isMatch(target, num) == TRUE

ウ　isValidNumber(num) == FALSE

エ　isValidNumber(num) == TRUE

c に関する解答群

ア　num[i] == num[j]

イ　target[i] == num[i]

ウ　target[i] == num[j]

エ　target[i] == target[j]

オ　target[j] == num[j]

d に関する解答群

ア　i != j　　イ　i < j

ウ　i <= j　　エ　i == j

オ　i > j　　カ　i >= j

e に関する解答群

ア　numHit != DIGITS

イ　numHit + numBlow == DIGITS

ウ　numHit + numBlow >= DIGITS

エ　numHit == DIGITS

オ　numHit == numBlow

カ　numHit > numBlow

問 10　次の C 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〕

関数 pattern_match_string は，対象文字列を先頭から１文字ずつ順に調べ，パターン文字列が表現している条件を満足しているかどうかを判定するプログラムである。

(1)　条件文字列は，対象文字列の各文字（対象文字）に対する条件を表現した１文字以上の文字列（パターン文字列）を連結したものである。対象文字とパターン文字列の対応関係は，図のとおりである。

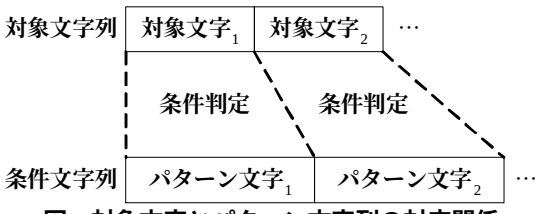

図　対象文字とパターン文字列の対応関係

(2)　関数 pattern_match_string の引数は，次のとおりである。
target：対象文字列
cond：条件文字列

(3)　対象文字は，英数字だけである。

(4)　パターン文字列に含まれる文字は，次のとおりである。
①　英数字
②　“＄”，“［”，“］”，“｛”，“｝”，“@”

(5)　対象文字列及び条件文字列に誤りはない。

(6)　表の例では，対象文字列はいずれの条件文字列で表現される条件も満足している。

表　対象文字列と条件文字列の例

対象文字列	FE2006
条件文字列	FE2@06
	F$U2006

〔プログラム〕

```
#include <string.h>

int pattern_match_string(char *, char *);

static char numeral[] =
                    "0123456789"; /* 数字 */
static char lower[] =
  "abcdefghijklmnopqrstuvwxyz"; /* 英小文字 */
static char upper[] =
  "ABCDEFGHIJKLMNOPQRSTUVWXYZ"; /* 英大文字 */

int pattern_match_string(char *target,
                                 char *cond){
  int flg = 0, i;

  while(*target != '\0' && *cond != '\0'
                                  && flg == 0){
      if(*cond == '$'){   /* 先頭が'$'の場合 */
        cond++;
        if(*cond == 'N'){
          for(i = 0; numeral[i] != *target
                && numeral[i] != '\0'; i++);
          if(numeral[i] != *target) flg = 1;
        }else if(*cond == 'L'){
          for(i = 0; lower[i] != *target
                 && lower[i] != '\0'; i++);
          if(lower[i] != *target) flg = 1;
        }else if(*cond == 'U'){
          for(i = 0; upper[i] != *target
                 && upper[i] != '\0'; i++);
          if(upper[i] != *target) flg = 1;
        }else if(*cond != *target){
          flg = 1;
        }
      }else if(*cond == '['){
                        /* 先頭が'['の場合 */
        flg = 1;
        for(cond++; *cond != ']'; cond++)
          if(*target == *cond) flg = 0;
      }else if(*cond == '{'){
                        /* 先頭が'{'の場合 */
        for(cond++; *cond != '}'; cond++)
          if(*target == *cond) flg = 1;
      }else if(*cond != '@'){
                     /* 先頭が'@'でない場合 */
        if(*target != *cond) flg = 1;
      }
      if(flg == 0){
        target++;
        cond++;
      }
```

```
  }
  if(flg != 0){
    return strlen(target);
  }else if(*target == *cond){
    return 0;
  }else{
    return -1;
  }
}
```

設問１　パターン文字列に関する次の説明中の［　　］に入れる正しい答えを，解答群の中から選べ。

(1)　対象文字が任意の英数字であることを表現するパターン文字列は，［　a　］である。

(2)　対象文字が数字であることを表現するパターン文字列は，［　b　］である。

(3)　対象文字が“ X ”,“ Y ”,“ Z ”のいずれかの文字であることを表現するパターン文字列は，［　c　］である。

(4)　対象文字が“ X ”,“ Y ”,“ Z ”のいずれの文字でもないことを表現するパターン文字列は，［　d　］である。

a，b に関する解答群

ア　“ @ ”　　イ　“ $A ”　　ウ　“ $L ”
エ　“ $LU ”　　オ　“ $N ”　　カ　“ $U ”
キ　“ $@ ”

c，d に関する解答群

ア　“ [XYZ] ”　　イ　“ {XYZ} ”
ウ　“ XYZ ”　　エ　“ $[XYZ$] ”
オ　“ ${XYZ$} ”

設問２　関数 pattern_match_string の返却値に関する次の説明中の［　　］に入れる正しい答えを，解答群の中から選べ。

(1)　対象文字列のすべての文字が条件文字列で与えた条件を満足している場合，返却値として［　e　］を返す。

(2)　対象文字列の途中の文字に条件文字列で与えた条件を満足しないものがあった場合，そこで判定を終了し，返却値として［　f　］を返す。

(3)　対象文字列と条件文字列のいずれかが途中で終了してしまった場合（'\0'が見つかった場合），そこで判定を終了し，返却値として －1 を返す。

解答群

ア　0
イ　－1
ウ　対象文字列の文字数
エ　検査した対象文字数
オ　検査して条件を満足していた対象文字数
カ　検査していない対象文字数 ＋ 1
キ　検査していない対象文字数

平成１８年度　春期　ＦＥ　午後解答　C 言語

問 6

設問１
a－ア　　b－イ

設問２
ア

問 10

設問１
a－ア　　b－ウ　　c－イ　　d－ア

設問２
エ

平成１８年度　秋期　ＦＥ　午後解答　C 言語

問 6

設問
a－ウ　　b－ア　　c－ウ　　d－エ　　e－エ

問 10

設問１
a－ア　　b－オ　　c－ア　　d－イ

設問２
e－ア　　f－カ