平成１７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，設問に答えよ。

〔プログラムの説明〕

関数 print_string は，欧文ピッチ処理（文字固有の字幅に従って字送りする）を行って印刷するとき，単語の途中で改行されないように英文を出力するプログラムである。

(1) 関数 print_string の引数は，次のとおりである。

line_w	１行の行幅（ポイント数）
str_list	出力する英文を構成する単語の配列（最後の要素には，NULL が格納されている）
char_list	出力する単語を構成する文字とその文字幅（ポイント数）のリスト（構造体 CHARPROF の配列）
space_w	空白文字の文字幅（ポイント数）

(2) 文字幅は，文字ごとに次に示す構造体 CHARPROF で定義される。

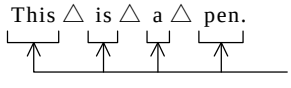
```
typedef struct { char char_p; /* 文字 */
          int  char_w; /* 文字幅（ポイント数） */
               } CHARPROF;
```

(3) 単語幅は，単語を構成する各文字の文字幅の和である。単語幅を求めるために，関数 word_width を用いる。

(4) 単語を出力しようとしたときに，１行の行幅を超える場合は，その単語が次の行の先頭になるように出力する。ただし，どの単語幅も行幅を超えることはない。

(5) 単語は，空白を含まない文字列である。単語と単語の間には，１文字の空白文字を出力する。ただし，行の最後に出力する単語の後には，空白文字は出力しない。

（例）



注　△は空白をしめす。

〔プログラム〕

```
#include <stdio.h>

typedef struct { char char_p; /* 文字 */
          int  char_w; /* 文字幅（ポイント数） */
               } CHARPROF;

void print_string(int, char *[],
                          CHARPROF *, int);
int word_width(char *, CHARPROF *);

void print_string(int line_w,
      char *str_list[], CHARPROF *char_list,
                              int space_w) {
   int cur_w = 0, str_w, idx;

   for (idx = 0; [ a ] ; idx++) {
      str_w = word_width(str_list[idx],
                                 char_list);
      [ b ] ;
      if (cur_w == str_w)    /* 最初の単語？ */
         printf("%s", str_list[idx]);
      else {
         cur_w += space_w;
         if (cur_w <= line_w)
            printf(" %s", str_list[idx]);
         else {
            [ c ] ;
            printf("\n%s", str_list[idx]);
         }
      }
   }
   putchar('\n');
}

int word_width(char *str, CHARPROF *char_list) {
   int print_w = 0, idx;

   while (*str != '\0') {
      for (idx = 0; [ d ] ; idx++);
      print_w += char_list[idx].char_w;
      str++;
   }
   return print_w;
}
```

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 に関する解答群

ア　idx <= line_w

イ　idx < line_w

ウ　idx + 1 < line_w

エ　str_list[idx] != NULL

オ　str_list[idx] = NULL

カ　str_list[idx] == NULL

b，c に関する解答群

ア　cur_w += space_w

イ　cur_w += str_w

ウ　cur_w = 0

エ　cur_w = space_w

オ　cur_w = space_w + str_w

カ　cur_w = str_w

d に関する解答群

ア　*str != *char_list[idx].char_p

イ　*str != char_list[idx].char_p

ウ　*str == *char_list[idx].char_p

エ　*str == char_list[idx].char_p

オ　str != char_list[idx].char_p

カ　str == char_list[idx].char_p

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 make_fare_table は，ある鉄道路線における始発駅から終着駅までの隣接駅間の距離から，その鉄道路線上の全駅間の運賃を求め，運賃表を作成するプログラムである。

(1)　関数 make_fare_table の引数は，次のとおりである。

num	始発駅から終着駅までの駅数
dist_list	隣接駅間の距離（km）を格納した配列（図１参照）
cost_list	距離運賃体系表（構造体 COSTUNIT の配列）
fare_table	運賃表 fare_table[m][n]（0≦ m≦ num−1，0≦ n≦ num−1 かつ m≠n）には，駅 m と駅 n の間の運賃を格納する。 fare_table[k][k]（0 ≦ k ≦ num−1）には，ゼロを格納する。

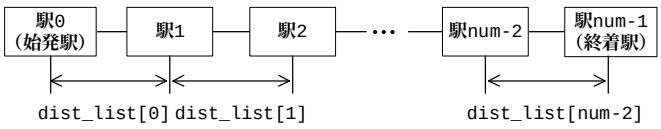


図１　引数 num と dist_list の関係

(2)　距離から運賃を求めるために，関数 calc_fare を用いる。

関数 calc_fare の引数及び返却値は，次のとおりである。

dist	２駅間の距離（km）
cost_list	距離運賃体系表（構造体 COSTUNIT の配列）
返却値	運賃（円）

(3)　２駅間を幾つかの区間に分け，区間ごとに計算した運賃を合計したものを，２駅間の運賃とする。

(4)　各区間の運賃は，次の ① ～ ④ のとおりである（図２参照）。

① 最初の 20 km 以下の区間（区間[0]）の運賃は，5 km 増えるごとに 100 円加算する。

② 20 km を超え 100 km 以下の区間（区間[1]）の運賃は，10 km 増えるごとに 180 円加算する。

③ 100 km を超え 500 km 以下の区間（区間[2]）の運賃は，50 km 増えるごとに 850 円加算する。

④ 500 km を超える区間（区間[3]）の運賃は，100 km 増えるごとに 1,650 円加算する。

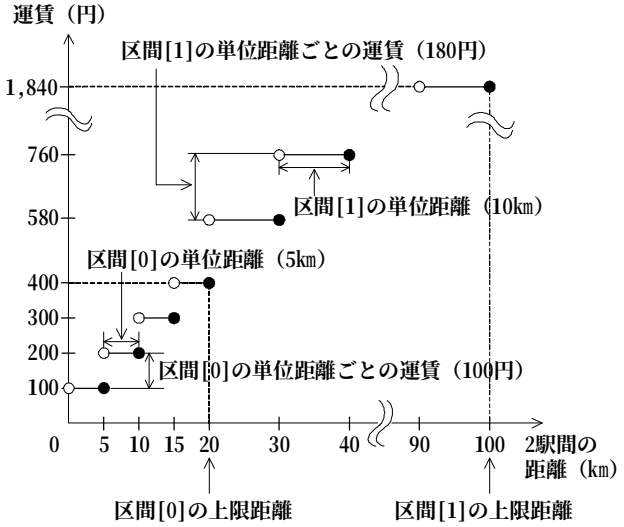


図２　２駅間の距離と運賃の関係

(5)　例えば，距離が 385 km ある駅間の運賃の求め方は，次のとおりである。

① 最初の 20 km 以下の区間（区間 [0]）の運賃は，5 km 増えるごとに 100 円加算されるので，次の式で求められる。

100 円×ceil(20km÷5km)＝400 円

② 20 km を超え 100 km 以下の区間（区間 [1]）の運賃は，10 km増えるごとに 180円加算されるので，次の式で求められる。

180円×ceil((100km−20km)÷10km)＝1,440円

③ 100 km を超え 385 km 以下の区間（区間 [2]）の運賃は，50 km 増えるごとに 850 円加算されるので，次の式で求められる。

850円×ceil((385km−100km)÷50km)＝5,100円

運賃は，① ～ ③ の合計で，6,940円になる。ここで，計算式中の ceil(X) は，X の小数点以下を切り上げた値を表す。

(6) 各区間の上限距離と単位距離は，距離運賃体系表で与える。距離運賃体系表は，上限距離の小さい区間から順に各要素を格納する。最後の要素の上限距離はゼロを格納し，距離の上限がないことを表す。

表　距離運賃体系表

区間 cost_list	上限距離（km） max_dist	単位距離（km） unit_dist	単位距離ごとの運賃（円） unit_cost
[0]	20	5	100
[1]	100	10	180
[2]	500	50	850
[3]	0	100	1,650

(7) 各区間の上限距離，単位距離及び単位距離ごとの運賃は，次に示す構造体 COSTUNIT で定義される。

```
typedef struct { int max_dist;
                   /* 上限距離（km） */
                 int unit_dist;
                   /* 単位距離（km） */
                 int unit_cost;
                   /*単位距離ごとの運賃（円） */
} COSTUNIT;
```

〔プログラム〕

```
#include <math.h>

typedef struct { int max_dist;
                   /* 上限距離（km） */
                 int unit_dist;
                   /* 単位距離（km） */
                 int unit_cost;
                   /* 単位距離ごとの運賃（円） */
                } COSTUNIT;

void make_fare_table(int, double *,
                     COSTUNIT *, int **);
int calc_fare(double, COSTUNIT *);

void make_fare_table(int num, double *dist_list,
       COSTUNIT *cost_list, int **fare_table) {
    int    idx0, idx1;
    double dist;

    for (idx0 = 0; idx0 < num; idx0++) {
        fare_table[idx0][idx0] = 0;
        dist = 0.0;
        for (idx1 = idx0 + 1; idx1 < num;
                                    idx1++) {
            [ a ] ;
α ➡     fare_table[idx0][idx1]
                = fare_table[idx1][idx0]
                = calc_fare(dist,cost_list);
    }
    }
}

int calc_fare(double dist, COSTUNIT *cost_list) {
    int fare = 0, idx = 0;
    int lower_limit;
              /*区間の下限（直前の区間の上限距離） */
    int upper_limit;
             /* 区間の上限（現在の区間の上限距離） */
    lower_limit = 0;
    upper_limit = cost_list[0].max_dist;
    while ( [ b ] ) {
        fare += ceil( [ c ] / (double)
                      cost_list[idx].unit_dist)
              * cost_list[idx].unit_cost;
        lower_limit = upper_limit;
        upper_limit = cost_list[ [ d ] ].
                                    max_dist;
    }
    fare += ceil( [ e ] / (double)
                      cost_list[idx].unit_dist)
             * cost_list[idx].unit_cost;
    return fare;
}
```

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　dist += dist_list[idx0]

イ　dist += dist_list[idx0 - 1]

ウ　dist += dist_list[idx1]

エ　dist += dist_list[idx1 - 1]

オ　dist = dist_list[idx0]

カ　dist = dist_list[idx0 - 1]

キ　dist = dist_list[idx1]

ク　dist = dist_list[idx1 - 1]

b に関する解答群

ア　upper_limit != 0 && dist > (double)lower_limit

イ　upper_limit != 0 && dist > (double)upper_limit

ウ　upper_limit != 0 || dist > (double)lower_limit

エ　upper_limit != 0 || dist > (double)upper_limit

オ　upper_limit == 0 && dist <= (double)lower_limit

カ　upper_limit == 0 && dist <= (double)upper_limit

キ　upper_limit == 0 || dist <= (double)lower_limit

ク　upper_limit == 0 || dist <= (double)upper_limit

c, e に関する解答群

ア　((double)upper_limit - dist)

イ　(dist - (double)lower_limit)

ウ　(double)(upper_limit - lower_limit)

エ　(double)lower_limit

オ　(double)upper_limit

カ　dist

d に関する解答群

ア　++idx　　イ　–idx

ウ　idx　　エ　idx++

オ　idx--　　カ　idx + 1

キ　idx - 1

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。

鉄道路線の運営会社が，始発駅及び終着駅以外の途中駅を境に２社に分割され，運賃の計算方法が次のように変更された。

(1)　どちらか１社の鉄道路線だけを利用する場合は，従来どおりの方法で運賃を計算する。

(2)　２社の鉄道路線を乗り継いで利用する場合は，各社の利用部分の運賃を個別に計算し，それを合計する。

これに対応するためには，関数 make_fare_table の引数に，境となる駅の番号 term_no（1 ≦ term_no ≦ num－2）を追加し，プログラム中のαを次のように変更すればよい。

```
if (idx0 < term_no && idx1 > term_no)
    fare_table[idx0][idx1]
                = fare_table[idx1][idx0]
                = calc_fare(dist, cost_list)
                 + fare_table[idx0][term_no];
else
    fare_table[idx0][idx1]
                = fare_table[idx1][idx0]
                = calc_fare(dist, cost_list);
if ( [        ] )
    dist = 0.0;
```

解答群

ア　idx0 == term_no - 1

イ　idx0 == term_no

ウ　idx0 == term_no + 1

エ　idx1 == term_no - 1

オ　idx1 == term_no

カ　idx1 == term_no + 1

平成１７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〕

関数 update_master は，処理年月を与えて，月刊誌の定期購読者マスタファイルの更新を行うプログラムである。

(1)　関数 update_master の引数は，次のとおりである。

omf_name　旧定期購読者マスタファイル名

trf_name　トランザクションファイル名

nmf_name　新定期購読者マスタファイル名

b_year　処理年月の西暦年

b_month　処理年月の月

(2)　新旧の定期購読者マスタファイル及びトランザクションファイルのレコード様式は次のとおりである。

購読者コード	空白	雑誌コード	空白	購読終了年月
12 けた	1 けた	12 けた	1 けた	6 けた

①　購読者コード及び雑誌コードは，空白を含まない 12 文字の英数字列である。

② 購読終了年月は，YYYYMM の形式で 6 けたの数字列である。

③ レコードの終端には，改行文字 '\n' が付いている。

④ レコードは，購読者コードを第１キー，雑誌コードを第２キーとして昇順に整列されている。

⑤ 購読者は，複数の雑誌を購読することはあるが，同一の雑誌を重複して購読することはない。

(3) 旧定期購読者マスタファイル及びトランザクションファイルのレコードに誤りはないものとする。

(4) トランザクションファイルには，旧定期購読者マスタファイルを更新するための，新規購読，購読期間延長及び購読打切りの３種類のレコードがある。

① 新規購読及び購読期間延長レコードの購読終了年月は，与えられた処理年月以降になっている。

② 購読打切りレコードの購読終了年月は，“999999”となっている。

(5) 次のどちらかの条件を満足するレコードを，抽出レコードとする。

① 旧定期購読者マスタファイルのレコードのうち，購読者コードと雑誌コードの組合せがトランザクションファイルには含まれていないもの

② トランザクションファイルに含まれるレコード

(6) 抽出レコードのうち，購読終了年月が与えられた処理年月以降であり，かつ“999999”ではないものだけを，新定期購読者マスタファイルに出力する。

(7) プログラム中で使用している関数 strcmp の仕様は，次のとおりである。

```
int strcmp(char *string1, char *string2)
```

機能：　引数で指定される二つの文字列 string1 と string2 を比較し，その大小関係を返す。

返却値：負の数（string1 が string2 より小さいとき）
　　0　（string1 が string2 に等しいとき）
　　正の数（string1 が string2 より大きいとき）

〔プログラム〕

```
#include <stdio.h>
#include <string.h>

void update_master(char *, char *, char *,int, int);

void update_master(char *omf_name, char *trf_name,
                   char *nmf_name, int b_year,
                                   int b_month)
{
    FILE *oldmf,      /* 旧定期購読者マスタファイル */
         *trf,        /* トランザクションファイル */
         *newmf;      /* 新定期購読者マスタファイル */
    char om_usrid[13], om_magid[13], t_usrid[13],
                                     t_magid[13];
    long b_date, om_ldate, t_ldate, m_sts, t_sts, flg;

    oldmf = fopen(omf_name, "r");
    trf   = fopen(trf_name, "r");
    newmf = fopen(nmf_name, "w");
    b_date = b_year * 100 + b_month;
    m_sts = fscanf(oldmf, "%12s %12s %6ld\n",
                  om_usrid, om_magid, &om_ldate);
    t_sts = fscanf(trf, "%12s %12s %6ld\n",
                  t_usrid, t_magid, &t_ldate);

    while ( [ a ] ) {
       if ( m_sts == EOF )
          flg = 1;
       else if ( t_sts == EOF )
          flg = -1;
       else if ( (flg = strcmp(om_usrid, t_usrid))
                                          == 0 )
          flg = strcmp(om_magid, t_magid);

       if ( flg < 0 ) {
          if ( om_ldate >= b_date )
             fprintf(newmf, "%12.12s %12.12s
                              %6ld\n", [ b ]);
          m_sts = fscanf(oldmf, "%12s %12s %6ld\n",
                    om_usrid, om_magid, &om_ldate);
       } else {
          if ( [ c ] )
             fprintf(newmf, "%12.12s %12.12s
                               %6ld\n", [ d ]);
          if ( flg == 0 )
             m_sts = fscanf(oldmf, "%12s %12s
                    %6ld\n",om_usrid, om_magid,
                                    &om_ldate);
          t_sts = fscanf(trf, "%12s %12s %6ld\n",
                      t_usrid, t_magid, &t_ldate);
       }
    }
    fclose(oldmf);
    fclose(trf);
    fclose(newmf);
}
```

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 に関する解答群

ア　(m_sts == EOF) && (t_sts == EOF)

イ　(m_sts == EOF) || (t_sts == EOF)

ウ　(m_sts != EOF) && (t_sts != EOF)

エ　(m_sts != EOF) || (t_sts != EOF)

オ　m_sts == EOF

カ　m_sts != EOF

キ　t_sts == EOF

ク　t_sts != EOF

b, d に関する解答群

ア　om_usrid, om_magid, om_ldate

イ　om_usrid, om_magid, t_ldate

ウ　om_usrid, t_magid, om_ldate

エ　om_usrid, t_magid, t_ldate

オ　t_usrid, om_magid, om_ldate

カ　t_usrid, om_magid, t_ldate

キ　t_usrid, t_magid, om_ldate

ク　t_usrid, t_magid, t_ldate

c に関する解答群

ア　b_date == 999999

イ　b_date != 999999

ウ　om_ldate == 999999

エ　om_ldate != 999999

オ　t_ldate == 999999

カ　t_ldate != 999999

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１～３に答えよ。

〔プログラムの説明〕
ある大学における成績出力用プログラムである。

(1)　この大学では，学期終了時に学生の履修科目成績一覧を図１に示すレコード様式でファイルに出力する。学生数は 15,000，科目数は 2,000 であり，得点は 0 ～ 100 の整数値である。

	学生キー	学生氏名	科目名	得点	科目名	得点
(例)	02498	基本五郎	情報処理 2	25	化学実験	50

	科目名	得点
…	英会話基礎 1	67

図１　履修科目成績一覧のレコード様式

(2)　出力に必要なデータは，図２に示す３種類のファイルに記述されている。

科目キー	科目名
0000	情報処理１
0001	情報処理 2
0002	数値解析
⋮	
0752	化学実験
⋮	
1997	英会話基礎 3
1998	英会話基礎 4
1999	英語応用

科目ファイル
course.txt

学生キー	学生氏名
00000	設計花子
00001	情報太郎
00002	開発次郎
⋮	
02498	基本五郎
⋮	
14997	試験守
14998	単元三郎
14999	言語大輔

学生ファイル
student.txt

科目キー	学生キー	得点
1932	00472	34
1932	09755	79
0357	10253	41
⋮		
0752	02498	50
⋮		
0173	13712	13
0173	00638	78
0173	03581	61

成績情報ファイル
record.txt

図２　読込みファイル群の例

(3)　プログラムは，はじめに関数 init を呼ぶことで，図２に示した３種類のファイルから図３に示すようなリスト構造をメモリ上に構成する。科目キー courseKey の科目名は，char 型の配列 courseName[courseKey] に格納される。また，学生キー studentKey の学生情報は STUDENT 型の構造体 student[studentKey] で表現され，メンバ studentName には学生氏名が，メンバ rFirstCourse にはその学生成績情報を表すリストの先頭へのポインタが格納される。学生の各履修科目についての成績情報は RECORD 型の構造体で表現し，メンバ score には得点，メンバ courseName には科目名へのポインタ，メンバ rNextCourse には他の履修科目の成績情報へのポインタが格納される。

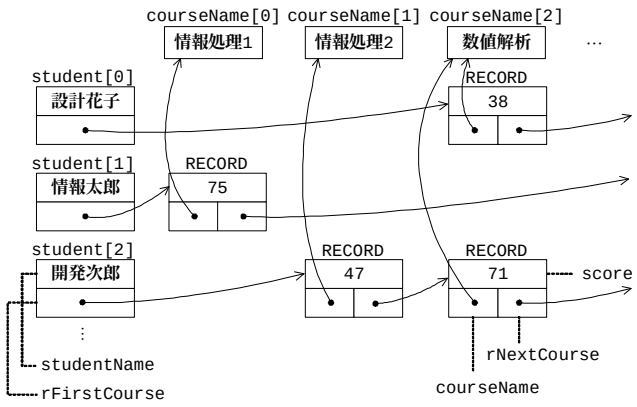


図３　データ構造例

(4) プログラム中で定義されているその他の関数の説明は次のとおりである。

```
void regist(int courseKey, int studentKey,
                                    short s);
```
機能：新たに RECORD 型の構造体を生成し，メンバ score に得点 s を代入する。また，この得点が学生キー studentKey の科目キー courseKey についてのものであるという情報を設定する。

```
void writeCourses();
```
機能：全学生分の履修科目成績一覧をファイルに出力する。

(5) このプログラムでは，次のライブラリ関数を用いる。

```
void *malloc(size_t size)
```
機能：size バイトの領域をメモリ上に割り付け，割り付けられた領域へのポインタを返す。メモリ割付けに失敗した場合は NULL を返す。

〔プログラム〕

```
#include <stdio.h>
#include <stdlib.h>
#define NUM_STUDENT 15000 /* 学生数 */
#define NUM_COURSE 2000   /* 科目数 */
#define MAX_WORD_LENGTH 21

struct RECORD {          /* 成績情報 */
   char *courseName;     /* 科目名へのポインタ */
   short score;          /* 得点 */
   struct RECORD *rNextCourse;
                         /* 次の履修科目の成績情報
                                    へのポインタ */
};

struct STUDENT {         /* 学生情報 */
   char studentName[MAX_WORD_LENGTH];
                         /* 学生氏名 */
   struct RECORD *rFirstCourse;
                         /* 一つ目の履修科目の成績
                               情報へのポインタ */
} student[NUM_STUDENT];

char courseName[NUM_COURSE][MAX_WORD_LENGTH];
                         /* 科目名の配列 */

void init();
void regist(int, int, short);
void writeCourses();

void init(){
   FILE  *fStudent = fopen("student.txt", "r");
   FILE  *fCourse = fopen("course.txt", "r");
   FILE  *fRecord = fopen("record.txt", "r");
   int studentKey, courseKey, score;
   while(fscanf(fCourse, "%d", &courseKey)
                                    != EOF){
      fscanf(fCourse, "%s", courseName
                                [courseKey]);
   }
   while(fscanf(fStudent, "%d",&studentKey)
                                    != EOF){
      fscanf(fStudent, "%s", student
                [studentKey].studentName);
      student[studentKey].rFirstCourse = NULL;
   }
   while(fscanf(fRecord, "%d %d %d",&courseKey,
                &studentKey,&score) != EOF){
      regist(courseKey, studentKey, (short)
                                    score);
   }
   fclose(fStudent);
   fclose(fCourse);
   fclose(fRecord);
}

void rigist(int courseKey, int studentKey,
                                    short s){
   struct RECORD *p;          ← α
   if ((p = (struct RECORD *)malloc(sizeof
                (struct RECORD))) == NULL){
      printf("メモリエラーです\n";
      exit(-1);
   }
   P->rNextCourse = student[studentKey].rFirstCourse;
   student[studentKey].rFirstCourse = [   a   ];   ← β
   p->courseName = [   b   ];
  p->score = s;
}

void writeCourses(){
   FILE *fOutput = fopen("output.txt", "w");
   struct RECORD *p;
   int i;
   for (i = 0; i < NUM_STUDENT; i++){
      fprintf(fOutput, "%05d %-24s ", i,
                    student[i].studentName);
      p = student[i].rFirstCourse;
      while(p != NULL){
         fprintf(fOutput, "%-24s %3d ",
                        [   c   ] );
         p = p->rNextCourse;
      }
      fprintf(fOutput, "\n");
   }
   fclose(fOutput);
}
```

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　&p　　イ　&p++　　ウ　NULL

エ　p　　オ　p++

b に関する解答群

ア　&courseName

イ　*courseName[courseKey]

ウ　courseName

エ　courseName[courseKey]

オ　p->rNextCourse->courseName

c に関する解答群

ア　&p->courseName, &p->score

イ　&p->courseName, p->score

ウ　*p->courseName, *p->score

エ　*p->courseName, p->score

オ　p->courseName, &p->score

カ　p->courseName, *p->score

キ　p->courseName, p->score

設問２　図１に例示した履修科目成績一覧において，科目名の出力順として正しい答えを，解答群の中から選べ。

解答群

ア　科目キーの値が小さい科目ほど先に出力される。

イ　科目キーの値が小さい科目ほど後に出力される。

ウ　先に関数 regist で登録した科目ほど先に出力される。

エ　先に関数 regist で登録した科目ほど後に出力される。

オ　履修者数の少ない科目ほど先に出力される。

カ　履修者数の少ない科目ほど後に出力される。

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。

図１に例示した履修科目成績一覧において，得点の降順に出力されるように，関数 regist のα，βを次のとおりに変更した。

なお，［ a ］には正しい答えが既に入っているものとする。

処置	プログラムの変更内容
αを置換	struct RECORD *p; struct RECORD *q;

βを置換	q = student[studentKey].rFirstCourse;
	if ([d]) {
	　p->rNextCourse = q;
	　student[studentKey].rFirstCourse
	　　= [a];
	} else {
	　while([e]){
	　　q = q->rNextCourse;
	　}
	　p->rNextCourse = q->rNextCourse;
	　q->rNextCourse = [a] ;
	}

解答群

ア　q != NULL && q->score < s

イ　q != NULL && q->score > s

ウ　q == NULL || q->score < s

エ　q == NULL || q->score > s

オ　q->rNextCourse != NULL &&
　　q->rNextCourse->score < s

カ　q->rNextCourse != NULL &&
　　q->rNextCourse->score > s

キ　q->rNextCourse == NULL ||
　　q->rNextCourse->score < s

ク　q->rNextCourse == NULL ||
　　q->rNextCourse->score > s

平成１７年度　春期　ＦＥ　午後解答　C 言語

問６

設問
a－エ　　b－イ　　c－カ　　d－イ

問 10

設問１
a－エ　　b－イ　　c－ウ　　d－ア　　e－イ

設問２
オ

平成１７年度　秋期　ＦＥ　午後解答　C 言語

問６

設問
a－カ　　b－ア　　c－カ　　d－ク

問 10

設問１
a－エ　　b－エ　　c－キ

設問２
エ

設問３
d－ウ　　e－カ