次に示す「年内での経過日数を求める」プログラムを入力し，自分のパソコンでコンパイル，実行できることを確認した後，次の問に答えなさい．なお，プログラムの日本語部分は，英語，ローマ字に変更してかまいません．

```
/*年内の経過日数を求める*/
#include <stdio.h>

/*- 各月の日数 -*/
int mdays[][13] = {
  {0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31},
  {0, 31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31}
};
/*--- 西暦 year 年のカレンダー ---*/
int *calendar(int year){
  if (year % 4 == 0 && year % 100 != 0 || year % 400 == 0)
    return (int *)mdays + 13; /* (int *) 強制的に型変換する命令 */
  else
    return (int *)mdays ;
}
/*--- 西暦 y 年 m 月 d 日の年内の経過日数を求める ---*/
int dayofyear(int y, int m, int d){
  int *i;
  int days = d;
/* 日数 */
  for (i = calendar(y) + m - 1; *i != 0; i--){
    ____________________________________
    days += *i ;                                    ①
  }
  return (days);
}

int main(void){
  int year, month, day; /* 年・月・日 */
  int retry;              /* もう一度？ */

  do {
    printf("年：");  scanf("%d", &year);
    printf("月：");  scanf("%d", &month);
    printf("日：");  scanf("%d", &day);
    printf("年内で%d 日目です。¥n", dayofyear(year, month, day));
    printf("もう一度しますか (1…はい／0…いいえ)：");
    scanf("%d", &retry);
  } while (retry == 1);

  return (0);
}
```

問(1)　西暦で年を指定すると，その年のカレンダーへのポインターを返す関数 calendar を用いて，次のような式を書いた．このときの式の値がいくらになるかを求めなさい．

*calendar(2020)	(*calendar(2020)) +2	*(calendar(2020) +2)
0	2	29

問(2)　関数 dayofyear を「dayofyear(2020, 3, 1)」として呼び出した．

(ア) このとき，関数 dayofyear の for ループ内の「days += *i」は何回実行されますか．

2 回

(イ) 関数 dayofyear の for 文のループが始まる直前の下線①の位置での*i の値はいくらですか．

29

(ウ) 関数 dayofyear の for 文が終了したとき，i が指すポインターの位置は，配列 mdays の何処を指していますか．mdays[0][0]のように，配列 mdays の添え字を用いて示してください．

mdays[1][0]

問(3)　各月の日数を記憶する配列 mdays の初期値を以下のように変更した．この変更した初期値でも「年内での経過日数を正しく求める」ように，関数 *calendar と関数 dayofyear を変更したプログラム(ソースファイル名は kadai2.c とすること)を作成してください．

```
/*- 各月の日数 -*/
int mdays[][13] = {
  {-4, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31},
  {1, 31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31}
};
```

```
/*--- 西暦 y 年 m 月 d 日の年内の経過日数を求める ---*/
int dayofyear(int y, int m, int d){
  int *i;
  int days;
/* 日数 */
  for(days = d, i = calendar(y) + m - 1; 1<*i && *i<32; i-- )
    days += *i;
  return (days);
}
```

第 3 回講義資料の理解度確認練習問題のプログラムで，身体検査データを定義する構造体 PhysCheck の型を以下のように変更しても，元のプログラムと同じ働きをするように，プログラムを変更してください.

```
/*--- 身体データ型 ---*/
typedef  struct{
  double  vision;   /* 視力 */
  int     height;   /* 身長 */
} Body ;
/*--- 身体検査データ型 ---*/
typedef  struct{
  Body   body;      /* 身体データ型 ---*/
  char   name[20]; /* 氏名 */
} PhysCheck ;
```

```
#include <stdio.h>
#define  VMAX   21 /* 視力の最大値 2.1 × 10 */
/*--- 身体データ型 ---*/
typedef struct{
 double vision;  /* 視力 */
 int    height;  /* 身長 */
 } Body;

/*--- 身体検査データ型 ---*/
typedef struct{
  Body   body;    /* 身体データ型 ---*/
  char   name[20];/* 氏名 */
} PhysCheck;

/*--- 身長の平均値を求める ---*/
double ave_height(PhysCheck *dat)
{
int n=0;
  double sum = 0.0;
  while(dat->body.height > 0){
    sum += (dat++)->body.height;
    n++;
  }
  return (sum/n);
}

/*--- 視力の分布を求める ---*/
void dist_vision(PhysCheck *dat, int dist[])
{ int vision;
 while((vision=(int)(10*((dat++)->body.vision)+0.5))>0){
   if (vision <= VMAX) ++*(dist + vision) ;
 }
}
```

```
/*--- 身長の分布を求める ---*/
void dist_height(PhysCheck *dat, int dist[])
{ int height;
  while((height = (dat++)->body.height) > 0){
    height = height >=140 ? (int) ((double)(height-140)/3.0) : -1;
    if (height < VMAX && height >= 0)  ++*(dist + height);
  }
}

int main(void)
{
  PhysCheck *y, x[] = {
    {{0.3, 162}, "AKASAKA Tadao"},
    {{0.7, 173}, "KATOH Tomiaki"},
    {{2.0, 175}, "SAITOH Syouji"},
    {{1.5, 171}, "TAKEDA Shinya"},
    {{0.4, 168}, "NAGAHAMA Masaki"},
    {{1.2, 174}, "HAMADA Tetsuaki"},
    {{0.8, 169}, "MATSUTOMI Akio"},
    {{0.0,   0}, ""} /*フラグ*/
  };
  int *z, vdist[VMAX]={0}; /* 視力の分布 */
  int     hdist[VMAX]={0}; /* 身長の分布 */
  puts("■□■ 身体検査一覧表 ■□■");
  puts("  氏名             身長 視力 ");
  puts("----------------------------");
  for(y = x; y->body.height > 0; y++)
    printf("%-18.18s%4d%5.1f¥n", y->name, y->body.height, y->body.vision);
  printf("¥n 平均身長：%5.1fcm¥n", ave_height(x));

  dist_vision(x, vdist); /* 視力の分布を求める */
  printf("¥n 視力の分布¥n");
  for (z = vdist; z < vdist + VMAX; z++)
    if (*z > 0) printf("%3.1f～：%2d 人¥n", (z-vdist)/10.0, *z);

  dist_height(x, hdist); /* 身長の分布を求める */
  printf("¥n 身長の分布¥n");
  for (z = hdist ; z <  hdist + VMAX ; z++)
    if (*z > 0) printf("%3d～：%2d 人¥n", (z - hdist)*3+140, *z);
  return (0);

}
```

第 4 回講義資料の理解度確認練習問題に示した「Boyer-Moore 法による文字列探索」プログラムを参考に，テキスト中の全てのパターンを探索し，見つかったパターンの個数を表示するプログラムを作成してください．

解答例

```
/* Boyer-Moore 法による文字列探索 */
#include   <stdio.h>
#include   <string.h>
#include   <limits.h>

/*--- Boyer-Moore 法による文字列探索 ---*/
char *bm_match(char *pat, char *txt){
   char   *pt; /* txt をなぞるカーソル */
   char   *pp;/* pat をなぞるカーソル */
   int    txt_len = strlen(txt); /* txt の文字数 */
   int    pat_len = strlen(pat); /* pat の文字数 */
   int    skip[UCHAR_MAX + 1]; /* スキップテーブル */
   int    i;

   for (i = 0; i <= UCHAR_MAX; i++) /* スキップテーブルの作成 */
      skip[i] = pat_len;
   for (pp = pat; *pp != '¥0'; pp++)
      skip[*pp] =    strlen(pp) - 1;
   skip[*(pp - 1)] = pat_len;   /* パターンの最後文字の移動距離はパターンの文字数 */

   pt = txt + pat_len - 1;        /* pat の末尾と比較する txt の文字を決定 */
   while ( pt < txt + txt_len) { /* txt の比較する文字の位置が txt の末尾を越えるまで */
      pp = pat + pat_len - 1;
      while (*pt == *pp) {
         if (pp == pat) return (pt); /* 一致した文字がパターンの最初の文字になれば終了 */
         pp--;
         pt--;
      }
      pt += (skip[*pt]> pat_len - strlen(pp)) ? skip[*pt] : pat_len - strlen(pp);
   }
   return (NULL);
}

/*--- Boyer-Moore 法で文字列探索を行い，テキスト中のパターンの数を数える ---*/
int bm_count(char *pat, char *txt){
    int num = 0;
    char *pt=txt;

    while( (pt=bm_match(pat, pt))!= NULL ){
       pt++;
       num++;
    }
    return(num);
}

int main(void)
{
   int     num;
```

```
    char    s1[80]; /* テキスト */
    char    s2[80]; /* パターン */
    printf("テキスト：");
    scanf("%s", s1);
    printf("パターン：");
    scanf("%s", s2);
    num = bm_count(s2, s1); /* 文字列 s1 中の文字列 s2 の個数を数える */
    if (num == 0)
        puts("テキスト中にパターンは存在しません。");
    else
        printf("%s の文字列の中に%s の文字列は%d 個見つかりました。¥n", s1, s2, num);
    return (0);
}
```

次に示すポインタ版の「Boyer-Moore 法による文字列探索」プログラムを入力し，自分のパソコンでコンパイル，実行できることを確認してください．なお，プログラムの日本語部分は，英語，ローマ字に変更してかまいません．さらに，以下の問いに答えなさい．

```
#include  <stdio.h>
#include  <string.h>
#include  <limits.h>
/*--- Boyer-Moore 法による文字列探索 ---*/
char *bm_match(char *pat , char *txt){
  char   *pt;                   /* txt をなぞるカーソル */
  char   *pp;                   /* pat をなぞるカーソル */
  int    txt_len = strlen(txt); /* txt の文字数 */
  int    pat_len = strlen(pat); /* pat の文字数 */
  int    skip[UCHAR_MAX + 1];   /* スキップテーブル */
  int    i;

  for (i = 0; i <= UCHAR_MAX; i++) /* スキップテーブルの作成 */
    skip[i] = pat_len;
  for (pp = pat; *pp != '¥0'; pp++)
    skip[*pp] =  strlen(pp) - 1;
  skip[*(pp - 1)] = pat_len;  /* パターンの最後文字の移動距離はパターンの文字数 */
  pt = txt + pat_len - 1;       /* pat の末尾と比較する txt の文字を決定 */
  while ( pt < txt + txt_len) { /* txt の比較する文字の位置が txt の末尾を越えるまで */
    pp = pat + pat_len - 1;     /* pat の最後の文字に着目 */
    while (*pt == *pp) {        ①
      if (pp == pat) return (pt); /* 一致した文字がパターンの最初の文字になれば終了 */
      pp--;
      pt--;
    }
    pt += (skip[*pt]> strlen(pp)) ? skip[*pt] : strlen(pp);
  }
  return (NULL);
}
int main(void){
  char  *s;
  char  s1[80]; /* テキスト */
  char  s2[80]; /* パターン */

  printf("テキスト:");
  scanf("%s", s1);
  printf("パターン:");
  scanf("%s", s2);
  s = bm_match(s2, s1); /* 文字列 s1 から文字列 s2 を Boyer-Moore 法で探索 */
                                                          ②
  if (s == NULL)
    puts("テキスト中にパターンは存在しません。");
  else
    printf("%d 文字目に見つかりました。¥n", s - s1 + 1);
  return (0);
}
```

1）キーボードからテキストとして「CAGACAGAGA」を，パターンとして「AGAG」を入力したとき，次の問に答えなさい．

(ア) スキップテーブルを表す配列の skip['G']と skip['A']の値はいくらですか．

skip['A'] = 1, skip['G'] =4

(イ) 下線②の関数 bm_match(s2, s1)の戻り値のポインタが指している文字は，テキスト「CAGACAGAGA」のどの文字ですか．

CAGACAGAGA

2）キーボードからテキストとして「APCPAACBABEAAA」を，パターンとして「AAA」を入力したとき，下線部①の while の条件式（*pt == *pp）は何度評価されますか．

①	A A	P A	1 C A	P	A	A	C	B	A	B	E	A	A	A
②	A	P	C	4 P A	3 A A	2 A A	C	B	A	B	E	A	A	A
③	A	P	C	P	A A	A A	5 C A	B	A	B	E	A	A	A
④	A	P	C	P	A	A	C	B A	A A	6 B A	E	A	A	A
⑤	A	P	C	P	A	A	C	B	A	B	9 E A	8 A A	7 A A	A A
⑥	A	P	C	P	A	A	C	B	A	B	E	12 A A	11 A A	10 A A

12 回

3）Boyer-Moore 法のアルゴリズムを変更して，指定したパターンをテキストの末尾から先頭に向かって探索する関数 char * bm_reverse_text_match(char *pat, char *txt)を作成してください．すなわち，bm_reverse_text_match では，txt が指す文字列の末尾から先頭に向かって逆順で並んでいるものをテキストとして，pat が指す文字列のパターンを探索するものとします．但し，関数 bm_reverse_text_match の引数も戻り値も，関数 bm_match と同じとします．例えば，txt が指す文字列が”ABCDEF”の場合，pat が指す文字列が”ABC”なら「テキスト中にパターンは存在しません」となるが，pat が指す文字列が”CBA”なら「4 文字目に見つかりました。」となるようにしてください．

```
/*--- Boyer-Moore 法で，指定したパターンをテキストの末尾から探索する ---*/

char *bm_reverse_text_match(char *pat, char *txt){
   char   *pt;       /* txt をなぞるカーソル */
   char   *pp;       /* pat をなぞるカーソル */
   int    txt_len = strlen(txt);   /* txt の文字数 */
   int    pat_len = strlen(pat);   /* pat の文字数 */
   int    skip[UCHAR_MAX + 1]; /*  スキップテーブル */
   int    i;

   for (i = 0; i <= UCHAR_MAX; i++) /*  スキップテーブルの作成 */
      skip[i] = pat_len;
   for (pp = pat; *pp != '¥0'; pp++)
      skip[*pp] =    strlen(pp) - 1;
   skip[*(pp - 1)] = pat_len;        /*  パターンの最後文字の移動距離はパターンの文字数 */
   pt = txt + txt_len - pat_len ;   /* pat の末尾と比較する txt の文字を決定 */

   while ( pt >= txt) {            /* txt の比較する文字の位置が txt の先頭を越えるまで */
      pp = pat + pat_len - 1;   /* pat の最後の文字に着目 */
      while (*pt == *pp) {
         if (pp == pat) return (pt); /*  一致した文字がパターンの最初の文字になれば終了 */
         pp--;
         pt++;
      }
      pt -= (skip[*pt]> strlen(pp) ) ? skip[*pt] : strlen(pp) ;
   }
   return (NULL);
}
```

アルゴリズムとデータ構造 授業中練習問題５

次のプログラムは「スタックの実現例」である．このプログラムに関して，以下の問いに答えなさい．さらに，このプログラムを入力し，自分のパソコンでコンパイル，実行できることを確認してください．なお，プログラムの日本語部分は，英語，ローマ字に変更してかまいません．

```
#include <stdio.h>
#define MAX  10

/*--- int 型スタックを実現する構造体 ---*/
typedef struct {
  int max;      /* スタックの容量 */
  int ptr;      /* スタックポインタ */
  int stk[MAX]; /* スタック本体*/
} IntStack;

/*--- スタックの初期化 ---*/
int Initialize(IntStack *s, int max){
  s->ptr = 0;
  s->max = max;
  return 0;
}
/*--- スタックにデータをプッシュ ---*/
int Push(IntStack *s, int x){
  if (s->ptr >= s->max) return -1; /* スタック満杯 */
  s->stk[s->ptr] = x;
  s->ptr++;
  return 0;
}
/*--- スタックからデータをポップ ---*/
int Pop(IntStack *s, int *x){
  if (s->ptr <= 0) return -1; /* スタックは空 */
  s->ptr--;
  *x = s->stk[s->ptr];
  return (0);
}
/*--- スタックからデータをピーク ---*/
int Peek(IntStack *s, int *x){
  if (s->ptr <= 0) return -1;
  *x = s->stk[s->ptr - 1];
  return 0;
}
/*--- スタックの容量 ---*/
int Capacity(const IntStack *s){
  return s->max;
}
/*--- スタックに積まれているデータ数 ---*/
int Size(const IntStack *s){
  return s->ptr;
}
/*--- スタックの全データの表示 ---*/
void Print(const IntStack *s){
```

```
  int i;

  for(i = 0; i < s->ptr; i++)
    printf("%d ", s->stk[i]);
  putchar('¥n');
}

int main(void){
  IntStack  s;

  Initialize(&s, MAX);

  while (1) {
    int  menu, x;
    printf("現在のデータ数：%d/%d¥n",Size(&s), Capacity(&s));
    printf("(1) プッシュ  (2) ポップ  (3) ピーク  (4) 表示  (0) 終了：");
    scanf("%d", &menu);
                                                    ①
    if (menu == 0) break;

    switch (menu) {
     case 1: /* プッシュ */
       printf("データ：");
       scanf("%d", &x);
       if (Push(&s, x) == -1)
          puts("¥a エラー：プッシュに失敗しました。¥n");
       break;
     case 2: /* ポップ */
       if (Pop(&s, &x) == -1)
          puts("¥a エラー：ポップに失敗しました。¥n ");
       else
          printf("ポップしたデータは%d です。¥n", x);
       break;
     case 3: /* ピーク */
       if (Peek(&s, &x) == -1)
          puts("¥a エラー：ピークに失敗しました。¥n");
       else
          printf("ピークしたデータは%d です。¥n", x);
       break;
     case 4: /* 表示 */
       Print(&s);
       break;
     }
    }
    return 0;
}
```

1）このプログラムの動作直後に数字の２を入力しました．このとき，次の問に答えなさい．

(ア) スタックを操作するために **switch** 文中から呼び出される関数はどの関数ですか．

Pop(&s, &x)

(イ) (ア)で呼び出された関数の実行が終了し，**main** に戻った時のその関数の戻り値はいくらですか．

－１

(ウ) (ア)で呼び出された関数の実行が終了ときのスタックポインタの値は幾らですか．

０

2）スタックに「24」,「38」,「11」,「24」,「47」,「53」の順で値が積まれ，下線部①の部分でプログラムが入力待ちをしているとき，次の問に答えなさい．

(ア) **s.ptr** と **s.stk[2]**の値はいくらですか．

s.ptr: 6,　s.stk[2]: 11

(イ) ピークを指示した場合，ピークしたデータとして表示される値はいくらですか．

53

(ウ)(イ)でピークを指示し，ピークしたデータの値を表示した後，再度下線部①の部分でプログラムが入力待ちをしているときの **s.stk[s.ptr -2]**の値はいくらですか．

47

3）第３回講義中課題で用いた「身体検査データを定義する」以下の構造体 PhysCheck 型を，スタックで扱えるようにする「身体検査データ型スタックを実現する構造体」PhysCheckStack 型を定義し，「スタックの実現例」のプログラム中の IntStack 型を PhysCheckStack 型に変更しても正しく動作するようにしてください．以下では，PhysCheckStack 型の定義と，変更したプログラム中の Push 関数と Pop 関数を答えなさい．

```
/*--- 身体データ型 ---*/
typedef  struct{
  double  vision;   /* 視力 */
  int     height;   /* 身長 */
} Body ;
/*--- 身体検査データ型 ---*/
typedef  struct{
  Body   body;      /* 身体データ型 ---*/
  char   name[20]; /* 氏名 */
} PhysCheck ;
```

解答例

```
/*--- 身体検査データ型スタックを実現する構造体 ---*/
typedef struct {
  int          max;     /* スタックの容量 */
  int          ptr;      /* スタックポインタ */
  PhysCheck  stk[MAX]; /* スタック本体*/
}PhysCheckStack;

/*--- スタックにデータをプッシュ ---*/
int Push(PhysCheckStack *s, PhysCheck x){
  if (s->ptr >= s->max) return -1; /* スタック満杯 */
  s->stk[s->ptr] = x;
  s->ptr++;
  return 0;
}

/*--- スタックからデータをポップ ---*/
int Pop(PhysCheckStack *s, PhysCheck *x){
  if (s->ptr <= 0) return -1; /* スタックは空 */
  s->ptr--;
  *x = s->stk[s->ptr];
  return (0);
}
```

アルゴリズムとデータ構造 授業中練習問題6

次のプログラムは「動的スタックの実現例」である．このプログラムに関して，以下の問いに答えなさい．さらに，このプログラムを入力し，自分のパソコンでコンパイル，実行できることを確認してください．なお，プログラムの日本語部分は，英語，ローマ字に変更してかまいません

```
#include  <stdio.h>
#include  <string.h>
#include  <stdlib.h>
#define   String_Max  80

/*--- 文字列型スタックを実現する構造体 ---*/
typedef struct{
  int  max;   /* スタックの容量 */
  int  ptr;   /* スタックポインタ */
  char **stk; /* スタック本体(char* へのポインタ配列 )*/
} StringsStack;

/*--- スタックの初期化 ---*/
int Initialize(StringsStack *s, int max){
 s->ptr = 0;
 if ((s->stk = calloc(max, sizeof(char *))) == NULL) {
    s->max = 0;  /* char* の配列の確保に失敗 */
    return -1;
  }
  /* char* の配列の確保に成功 */
  s->max = max;
  return 0;
}

/*--- スタックの後始末 ---*/
void Terminate(StringsStack *s){
  if (s->stk != NULL){
    while( --s->ptr >= 0)
       free(s->stk[s->ptr]); /* 動的な文字列保存用配列を解放 */
    free(s->stk);
  }
  s->max = s->ptr = 0;
}

/*--- スタックにデータをプッシュ ---*/
int Push(StringsStack *s, char *x){
  if (s->ptr >= s->max) return -1; /* スタック満杯 */
  if ((s->stk[s->ptr] = calloc(strlen(x)+1, sizeof(char))) == NULL)
    /* データをコピーするための動的な文字列保存用配列を確保することに失敗 */
    return -1;
  strcpy(s->stk[s->ptr], x);
  s->ptr++;
  return 0;
}

/*--- スタックからデータをポップ ---*/
int Pop(StringsStack *s, char *x){
  if (s->ptr <= 0) return -1; /* スタックは空 */
  s->ptr--;
  strcpy(x, s->stk[s->ptr]);
  free(s->stk[s->ptr]); /* ポップしたので，動的な文字列保存用配列を解放 */
  return 0;
}
/*--- スタックからデータをピーク ---*/
```

```
int Peek(StringsStack *s, char *x){
  if (s->ptr <= 0) return -1;
  strcpy(x, s->stk[s->ptr - 1]);
  return 0;
}

/*--- スタックの容量 ---*/
int Capacity(const StringsStack *s){
  return s->max;
}

/*--- スタックに積まれているデータ数 ---*/
int Size(const StringsStack *s){
  return s->ptr;
}

/*--- スタックの全データの表示 ---*/
void Print(const StringsStack *s){
  int i;

  for(i = 0; i < s->ptr; i++)
    printf("%s¥n", s->stk[i]);
}

int main(void){
  StringsStack s;
  int max;

  printf("スタックの大きさを入力してください");
  scanf("%d", &max);

  if (Initialize(&s, max)==-1){
     puts("スタックの生成に失敗しました。");
    return 1;
  }

  while (1) {
    int  menu;
    char x[String_Max];

    printf("現在のデータ数：%d/%d¥n",Size(&s), Capacity(&s));
    printf("(1)プッシュ (2)ポップ (3)ピーク (4)表示 (0)終了：");
    scanf("%d", &menu);
    ―――――――――――――――― ①
    if (menu == 0) break;

    switch (menu) {
     case 1: /* プッシュ */
       printf("プッシュする文字列：");
       scanf("%s", x);
       if (Push(&s, x) == -1)
          puts("¥a エラー：プッシュに失敗しました. ¥n");
       break;
     case 2: /* ポップ */
       if (Pop(&s, x) == -1)
          puts("¥a エラー：ポップに失敗しました. ¥n");
       else{
           printf("ポップした文字列は%s, ", x);
       }
       break;
     case 3: /* ピーク */
```

```
        if (Peek(&s, x) == -1)
          puts("¥a エラー：ピークに失敗しました. ¥n");
        else{
          printf("ピークした文字列は%s, ", x);
        }
        break;
      case 4: /* 表示 */
        Print(&s);
        break;
      }
    }
  Terminate(&s);
  return 0;
}
```

1）このプログラムの動作直後に－1 を入力すると,「スタックの生成に失敗しました。」とのメッセージを表示して，プログラムが終了しました.「スタックの生成に失敗しました。」のメッセージを表示し，プログラムが終了することになった原因を説明しなさい.

このプログラムが動作直後に入力を求める値は，スタックの大きさである．従って，－1 は，スタックの大きさと解釈され，Initialize()関数の中で，スタックの実体である配列を確保するために利用されるが【calloc(-1, sizeof(char *))】，大きさが負の配列は確保できないため，配列の確保に失敗する．そのため，Initialize()の関数の値が-1 となり，「スタックの生成に失敗しました。」とのメッセージを表示して，プログラムは終了する.

2）このプログラムの動作直後に 0 を入力すると，プログラムは終了することなく動作しつづけ，下線部①の部分で入力待ちとなりました．このとき，次の問に答えなさい.

(ア) ここでポップを指示した場合，どのようなことが起こるか説明しなさい.

もし，大きさ 0 の配列が確保できたとしても，スタックにはデータが積まれていないため，ポップできず，エラーを知らせる音が鳴り，「エラー：ポップに失敗しました。」とのメッセージが表示される.

(イ) ここで，スタックに文字列" Maki"をプッシュすることを指示した場合，どのようなことが起こるか説明しなさい.

もし，大きさ 0 の配列が確保できたとしても，Initialize ()関数の中で s->ptr と s->max の値が共に 0 と設定されるので，スタックにデータを積もうとしたとき，Push()関数の if (s->ptr >= s->max) return -1 が成立する．従って，プッシュしようとする文字列" Maki"は，実際にはスタックにはプッシュされず，エラーを知らせる音が鳴り，「エラー：プッシュに失敗しました。」とのメッセージが表示される.

3）このプログラムの動作直後に 7 を入力し，スタックに文字列，”Ant”, "Duck", "Eagle", ”Pika”の順で積んだ後，下線部①の部分で入力待ちとなりました．このとき，次の問に答えなさい．

(ア)　この状態で，スタックから文字列を連続していくつ取り出せますか．

スタックに 4 つ文字列が積まれているので，連続して 4 個（4 回）の文字列が取り出せる．

(イ)　この状態から，スタックに連続して文字列をいくつ積めますか．

スタックの大きさを 7 と指定したので，スタックには全部で 7 個分の文字列が積める．今，4 つの文字列を積んだので，スタックの空き領域は 3 である．従って，この状態から 3 つの文字列を積むことができる．

4）このプログラムに，パターンの文字列を指定すると，スタック上の文字列の中から入力したパターンを，スタックの頂上から底に向かって探索し, 最初に見つけたパターンを含む文字列のスタックポインタの値を返す関数 Search を追加してください．関数 Search は以下のような動作をするものとします．追加した関数 Search を答えなさい．

- int Search(StringsStack *s , char *x)は，スタックに積まれている文字列の中から，パターンの文字列(x の先頭からパターンが入っている)で指定された文字列を，スタックの頂上から底に向かって探索し, 最初に見つけたパターンを含む文字列のスタックポインタの値を戻り値として返すようにしてください．ただし，パターンが見つからなかった場合は，-1 を返すようにしてください．

  関数 bm_mactch は，練習問題 4 で用いたものであるので，ここでは省略する．

```
/*---  スタック中のパターンの数を数える  ---*/
int Seach(StringsStack *s, char *pat){
   int pos = s->ptr ;

   while(pos>0)
      if   (bm_match(pat, s->stk[--pos]) != NULL)   return pos;
   return -1;
}
```

アルゴリズムとデータ構造　授業中練習問題7

次のプログラムは「キューの実現例」である．このプログラムに関して，以下の問いに答えなさい．さらに，このプログラムを入力し，自分のパソコンでコンパイル，実行できることを確認してください． なお,プログラムの日本語部分は,英語,ローマ字に変更してかまいません.

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

typedef struct {/*--- キューを実現する構造体 ---*/
  int max;  /* キューの容量 */
  int num;  /* 現在の要素数 */
  int front;/* 先頭要素カーソル */
  int rear; /* 末尾要素カーソル */
  int *que; /* キュー本体（の先頭要素へのポインタ） */
} IntQueue;

/*--- キューの初期化 ---*/
int Initialize(IntQueue *q, int max){
  q->num = q->front = q->rear = 0;
  if ((q->que = calloc(max, sizeof(int))) == NULL) {
    q->max = 0; /* 配列の確保に失敗 */
    return -1;
  }
  q->max = max;
  return 0;
}

/*--- キューの後始末 ---*/
void Terminate(IntQueue *q){
  if (q->que != NULL) {
    free(q->que);/* 配列を解放 */
    q->max = q->num = q->front = q->rear = 0;
  }
}

/*--- キューにデータをエンキュー ---*/
int Enque(IntQueue *q, int x){
  if (q->num >= q->max)
    return -1; /* キューは満杯 */
  else {
    q->num++;
    q->que[q->rear++] = x;
    if (q->rear == q->max) q->rear = 0;
    return 0;
  }
}

/*--- キューからデータをデキュー ---*/
int Deque(IntQueue *q, int *x){
```

```
  if (q->num <= 0)/* キューは空 */
    return -1;
  else {
    q->num--;
    *x = q->que[q->front++];
    if (q->front == q->max) q->front = 0;
    return 0;
  }
}
/*--- キューからデータをピーク ---*/
int Peek(const IntQueue *q, int *x)
{
  if (q->num <= 0)
    return -1;
  *x = q->que[q->front];
  return 0;
}

/*--- キューの容量 ---*/
int Capacity(const IntQueue *q){
  return (q->max);
}

/*--- キューに蓄えられているデータ数 ---*/
int Size(const IntQueue *q){
  return (q->num);
}

/*--- 全データの表示 ---*/
void Print(const IntQueue *q){
  int i;

  for(i = 0; i < q->num; i++)
    printf("%d ", q->que[(i + q->front) % q->max]);
  putchar('¥n');
}

int main(void){
  IntQueue  que;

  if (Initialize(&que, 7) == -1) {
    puts("キューの生成に失敗しました。");
    return 1;
  }
  while (1) {
    int  m, x;

    printf("現在のデータ数：%d/%d¥n", Size(&que), Capacity(&que));
    printf("(1) エンキュー (2) デキュー (3) ピーク (4) 表示 (0) 終了：");
    scanf("%d", &m);
```

①

```
    if (m == 0) break;

    switch(m){
    case 1: printf("データ：");   scanf("%d", &x);
      if (Enque(&que, x) == -1)
        puts("¥a エラー:データのエンキューに失敗しました。");
      break;
    case 2:
      if (Deque(&que, &x) == -1)
        puts("¥a エラー:デキューに失敗しました。");
      else
        printf("デキューしたデータは%d です。¥n", x);
      break;
    case 3: /* ピーク */
      if (Peek(&que, &x) == -1)
        puts("¥a エラー：ピークに失敗しました。");
      else
        printf("ピークしたデータは%d です。¥n", x);
      break;
    case 4: /* 表示 */
      Print(&que);
      break;
    }
  }
  Terminate(&que);
  return 0;
}
```

1）このプログラムを動作させ，キューに「47」,「8」,「11」,「24」の順で値をエンキューした後，下線部①の部分で入力待ちとなりました．このとき，次の問に答えなさい．

(ア) この状態で，キューから連続して何回デキューできますか．

4 回

(イ) この状態で，キューから連続して何回エンキューできますか．

3 回

(ウ) この状態での que.max, que.front, que.rear, que.que[2]の値を書きなさい．

que.max： 7

que.front：0

que.rear：4

que.que[2]：11

２）スタックに保存するデータ(プッシュ, ポップするデータ)が文字列となっている授業中練習問題６プログラムを参考に, このプログラムのキューに保存するデータ(エンキュー, デキューするデータ)を文字列に変更しなさい. ただし, 授業中練習問題6のプログラムで文字列のスタックとして StringsStack 型を定義したように, このプログラムの変更でも, 文字列のキューとして StringsQueue 型を定義して利用すること. なお, キューに保存できる文字列は, 動的な文字列として実現すること. ただし, main で入力できる文字列の長さは 80 文字以内とするここでは, 変更したプログラム中の StringsQueue 型の定義, Enque 関数と Deque 関数を答えなさい.

```
#include  <stdio.h>
#include  <string.h>
#include  <stdlib.h>
#define   String_Max  80

typedef struct {/*--- 文字列型キューを実現する構造体 ---*/
  int max;  /* キューの容量 */
  int num;  /* 現在の要素数 */
  int front;/* 先頭要素カーソル */
  int rear; /* 末尾要素カーソル */
  char **que; /* キュー本体(char* へのポインタ配列 )*/
} StringsQueue;

/*--- キューの初期化 ---*/
int Initialize(StringsQueue *q, int max){
  q->num = q->front = q->rear = 0;
  if ((q->que = calloc(max, sizeof(char *))) == NULL) {
    q->max = 0; /* 配列の確保に失敗 */
    return -1;
  }
  q->max = max;
  return 0;
}

/*--- キューの後始末 ---*/
void Terminate(StringsQueue *q){
  if (q->que != NULL) {
    while( q->num > 0){
      q->num--;
      free(q->que[q->front++]); /* 動的な文字列保存用配列を解放 */
      if (q->front == q->max) q->front = 0;
    }
    free(q->que);
  }
  q->max = q->num = q->front = q->rear = 0;
}

/*--- キューにデータをエンキュー ---*/
int Enque(StringsQueue *q, char *x){
  if (q->num >= q->max) /* キューは満杯 */
    return -1;
  if ((q->que[q->rear] = calloc(strlen(x)+1, sizeof(char))) == NULL)
```

```
    /* データをコピーするための動的な文字列保存用配列を確保することに失敗 */
    return -1;
  q->num++;
  strcpy(q->que[q->rear++], x);
  if (q->rear == q->max) q->rear = 0;
  return 0;
}

/*--- キューからデータをデキュー ---*/
int Deque(StringsQueue *q, char *x){
  if (q->num <= 0)/* キューは空 */
    return -1;
  q->num--;
  strcpy(x, q->que[q->front]);
  free(q->que[q->front++]);
  if (q->front == q->max) q->front = 0;
  return 0;
}
/*--- キューからデータをピーク ---*/
int Peek(const StringsQueue *q, char *x)
{
  if (q->num <= 0)
    return -1;
  strcpy(x, q->que[q->front]);
  return 0;
}

/*--- キューの容量 ---*/
int Capacity(const StringsQueue *q){
  return (q->max);
}

/*--- キューに蓄えられているデータ数 ---*/
int Size(const StringsQueue *q){
  return (q->num);
}

/*--- 全データの表示 ---*/
void Print(const StringsQueue *q){
  int i;

  for(i = 0; i < q->num; i++)
    printf("%s¥n",q->que[((i + q->front)% q->max)]);
}

int main(void){
  StringsQueue  que;

  if (Initialize(&que, 8) == -1) {
    puts("キューの生成に失敗しました。");
    return 1;
```

```
  }

  while (1) {
    int  m;
    char x[String_Max];

    printf("現在のデータ数：%d/%d¥n", Size(&que), Capacity(&que));
    printf("(1) エンキュー (2) デキュー (3) ピーク (4) 表示 (0) 終了：");
    scanf("%d", &m);

    if (m == 0) break;

    switch(m){
    case 1: printf("データ：");
      scanf("%s", x);
      if (Enque(&que, x) == -1)
        puts("¥a エラー:データのエンキューに失敗しました。");
      break;
    case 2:
      if (Deque(&que, x) == -1)
        puts("¥a エラー:デキューに失敗しました。");
      else
        printf("デキューしたデータは%s¥n", x);
      break;
    case 3: /* ピーク */
      if (Peek(&que, x) == -1)
        puts("¥a エラー：ピークに失敗しました。");
      else
        printf("ピークしたデータは%s¥n", x);
      break;
    case 4: /* 表示 */
      Print(&que);
      break;
    }
  }
  Terminate(&que);
  return (0);
}
```

アルゴリズムとデータ構造　授業中練習問題8

次のプログラムは「ポインタによる線形リストの実現例」(教科書の List9-3 とほぼ同じ)である．このプログラムに関して，以下の問いに答えなさい．さらに，このプログラムを入力し，自分のパソコンでコンパイル，実行できることを確認してください． なお，プログラムの日本語部分は，英語，ローマ字に変更してかまいません．

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

#define MEMBER_NO    1 /* 番号を表す定数値 */
#define MEMBER_NAME  2 /* 氏名を表す定数値 */

/*--- 会員データ ---*/
typedef struct {
  int  no;       /* 番号 */
  char name[20]; /* 氏名 */
} Member;
/*--- ノード ---*/
typedef struct __node {
  Member        data;  /* データ */
  struct __node *next; /* 後続ポインタ*/
} Node;
/*--- 線形リスト ---*/
typedef struct {
  Node *head; /* 先頭ノードへのポインタ */
  Node *crnt; /* 着目ノードへのポインタ */
} List;
/*--- 会員の番号の比較関数 ---*/
int MemberNoCmp(const Member *x, const Member *y){
  return x->no < y->no ? -1 : x->no > y->no ? 1 : 0;
}
/*--- 会員の氏名の比較関数 ---*/
int MemberNameCmp(const Member *x, const Member *y){
  return strcmp(x->name, y->name);
}
/*--- 会員データ（番号と氏名）の表示（改行なし）---*/
void PrintMember(const Member *x){
  printf("%d %s", x->no, x->name);
}
/*--- 会員データ（番号と氏名）の表示（改行あり）---*/
void PrintLnMember(const Member *x){
  printf("%d %s¥n", x->no, x->name);
}
/*--- 会員データ（番号と氏名）の読込み ---*/
Member ScanMember(const char *message, int sw){
  Member temp;

  printf("%s するデータを入力してください。¥n", message);

  if (sw & MEMBER_NO)   { printf("番号："); scanf("%d", &temp.no); }
  if (sw & MEMBER_NAME) { printf("氏名："); scanf("%s", temp.name); }
```

```
  return temp;
}
/*--- 一つのノードを動的に生成 ---*/
static Node *AllocNode(void){
  return calloc(1, sizeof(Node));
}
/*--- n の指すノードの各メンバーに値を設定 ----*/
static void SetNode(Node *n, const Member *x, const Node *next){
  n->data = *x;   /* データ */
  n->next = next; /* 後続ポインタ */
}
/*--- 線形リストを初期化 ---*/
void Initialize(List *list){
  list->head = NULL; /* 先頭ノード */
  list->crnt = NULL; /* 着目ノード */
}
/*--- 関数 compare によって x と一致すると判定されるノードを探索 ---*/
Node *Search(List *list, const Member *x,
             int compare(const Member *x, const Member *y)){
  Node *ptr = list->head;

  while (ptr != NULL) {
    if (compare(&ptr->data, x) == 0) {  /* キー値が一致 */
      list->crnt = ptr;
      return ptr;       /* 探索成功 */
    }
    ptr = ptr->next;    /* 後続ノードに着目 */
  }
  return NULL;          /* 探索失敗 */
}
/*--- 先頭にノードを挿入 ---*/
void InsertFront(List *list, const Member *x){
  Node *ptr = list->head;
  list->head = list->crnt = AllocNode();
  SetNode(list->head, x, ptr);
}
/*--- 末尾にノードを挿入 ---*/
void InsertRear(List *list, const Member *x){
  if (list->head == NULL)               /* 空であれば */
    InsertFront(list, x);               /* 先頭に挿入 */
  else {
    Node *ptr = list->head;
    while (ptr->next != NULL)
      ptr = ptr->next;
    ptr->next = list->crnt = AllocNode();
    SetNode(ptr->next, x, NULL);
  }
}
/*--- 先頭ノードを削除 ---*/
void RemoveFront(List *list){
  if (list->head != NULL) {
    Node *ptr = list->head->next;  /* ２番目のノードへのポインタ */
```

```
    free(list->head);              /* 先頭ノードを解放 */
    list->head = list->crnt = ptr; /* 新しい先頭ノード */
  }
}
/*--- 末尾ノードを削除 ---*/
void RemoveRear(List *list){
  if (list->head != NULL) {
    if ((list->head)->next == NULL) /* ノードが一つだけであれば */
      RemoveFront(list);            /* 先頭ノードを削除 */
    else {
      Node *ptr = list->head;
      Node *pre;

      while (ptr->next != NULL) {
        pre = ptr;
        ptr = ptr->next;
      }
      pre->next = NULL;             /* pre は末尾から 2 番目 */
      free(ptr);                    /* ptr は末尾 */
      list->crnt = pre;
    }
  }
}
/*--- 着目ノードを削除 ---*/
void RemoveCurrent(List *list){
  if (list->head != NULL) {
    if (list->crnt == list->head)  /* 先頭ノードに着目していれば */
      RemoveFront(list);           /* 先頭ノードを削除 */
    else {
      Node *ptr = list->head;

      while (ptr->next != list->crnt)
        ptr = ptr->next;
      ptr->next = list->crnt->next;
      free(list->crnt);
      list->crnt = ptr;
    }
  }
}
/*--- 全ノードを削除 ---*/
void Clear(List *list){
  while (list->head != NULL)    /* 空になるまで */
    RemoveFront(list);          /* 先頭ノードを削除 */
  list->crnt = NULL;
}
/*--- 着目ノードのデータを表示 ---*/
void PrintCurrent(const List *list){
  if (list->crnt == NULL)
    printf("着目ノードはありません。");
  else
    PrintMember(&list->crnt->data);
}
```

```
/*--- 着目ノードのデータを表示（改行付き）---*/
void PrintLnCurrent(const List *list){
  PrintCurrent(list);
  putchar('¥n');
}
/*--- 全ノードのデータをリスト順に表示 ---*/
void Print(const List *list){
  if (list->head == NULL)
    puts("ノードがありません。");
  else {
    Node *ptr = list->head;

    puts("【一覧表】");
    while (ptr != NULL) {
      PrintLnMember(&ptr->data);
      ptr = ptr->next;  /* 後続ノードに着目 */
    }
  }
}
/*--- 線形リストの後始末 ---*/
void Terminate(List *list){
  Clear(list);   /* 全ノードを削除 */
}
/*--- メニュー ---*/
typedef enum {
        TERMINATE, INS_FRONT, INS_REAR,  RMV_FRONT, RMV_REAR, PRINT_CRNT,
        RMV_CRNT,  SRCH_NO,   SRCH_NAME, PRINT_ALL, CLEAR
} Menu;
/*--- メニュー選択 ---*/
Menu SelectMenu(void){
  int  i, ch;
  char *mstring[] = {
    "先頭にノードを挿入", "末尾にノードを挿入", "先頭のノードを削除",
    "末尾のノードを削除", "着目ノードを表示",   "着目ノードを削除",
    "番号で探索", "氏名で探索", "全ノードを表示", "全ノードを削除"
  };

  do {
    for (i = TERMINATE; i < CLEAR; i++) {
      printf("(%2d) %-18.18s  ", i + 1, mstring[i]);
      if ((i % 3) == 2)
        putchar('¥n');
    }
    printf("( 0) 終了  :");
    scanf("%d", &ch);
  } while (ch < TERMINATE || ch > CLEAR);

  return (Menu)ch;
}
/*--- メイン ---*/
int main(void){
  Menu menu;
  List list;
```

```
  Member x;

  Initialize(&list);    /* 線形リストの初期化 */

  do {
    switch (menu = SelectMenu()) {
    case INS_FRONT :  /* 先頭にノードを挿入 */
      x = ScanMember("先頭に挿入", MEMBER_NO | MEMBER_NAME);
      InsertFront(&list, &x);
      break;
    case INS_REAR : /* 末尾にノードを挿入 */
      x = ScanMember("末尾に挿入", MEMBER_NO | MEMBER_NAME);
      InsertRear(&list, &x);
      break;
    case RMV_FRONT : /* 先頭ノードを削除 */
      RemoveFront(&list);
      break;
    case RMV_REAR : /* 末尾ノードを削除 */
      RemoveRear(&list);
      break;
    case PRINT_CRNT : /* 着目ノードのデータを表示 */
      PrintLnCurrent(&list);
      break;
    case RMV_CRNT :  /* 着目ノードを削除 */
      RemoveCurrent(&list);
      break;
    case SRCH_NO :  /* 番号による探索 */
      x = ScanMember("探索", MEMBER_NO);
      if (Search(&list, &x, MemberNoCmp) != NULL)
        PrintLnCurrent(&list);
      else
        puts("その番号のデータはありません。");
      break;
    case SRCH_NAME : /* 氏名による探索 */
      x = ScanMember("探索", MEMBER_NAME);
      if (Search(&list, &x, MemberNameCmp) != NULL)
        PrintLnCurrent(&list);
      else
        puts("その名前のデータはありません。");
      break;
    case PRINT_ALL : /* 全ノードのデータを表示 */
      Print(&list);
      break;
    case CLEAR : /* 全ノードを削除 */
      Clear(&list);
      break;
    }
  } while (menu != TERMINATE);

  Terminate(&list);/* 線形リストの後始末 */
  return 0;
}
```

1）このプログラムの動作直後に,「先頭にノードを挿入」を指示しました．このとき，switch 文中の列挙型の変数 menu の値と switch 文から呼びだされる関数名を答えなさい.

menu： INS_ FRON
関数名：ScanMember(″先頭に挿入″, MEMBER_NO | MEMBER_NAME)
関数名：InsertFront(&list, &x);

2）このプログラムの動作直後に,「先頭にノードを挿入」を指示し，データ（番号：1019999, 氏名：funfun）を入力しました．このとき，次の問に答えなさい.

(ア) main 関数中の list.head->data.no の内容を示しなさい.

list.head->data.no：1019999

(イ) main 関数中の list.crnt->data.name[5]の内容を示しなさい.

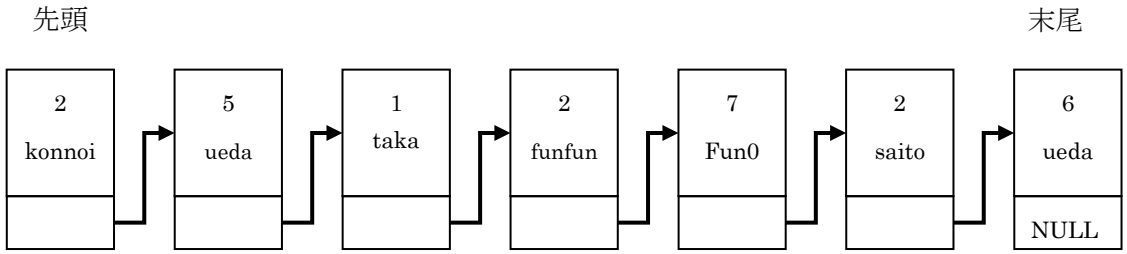
list.crnt->data.name[5]：n

（ウ）さらに,「末尾にノードを挿入」を指示し，データ（番号：1019777, 氏名：fun2）を入力しました．このとき，main 関数中の list.head->data..name[3] の内容を示しなさい.

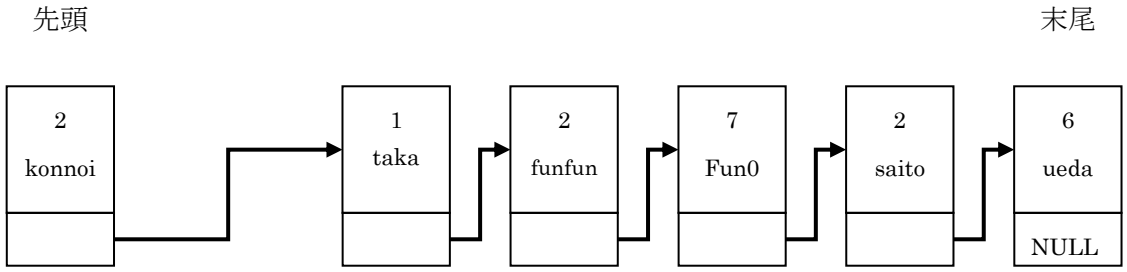
list.head->data.name[3] ：f

3）このプログラムの動作直後に,「末尾にノードを挿入」を連続し 4 回指示し, 4 つのデータを [2, funfun], [7, fun0], [2, saito], [6, ueda]の順番で入力しました．続けて，3 回連続して「先頭にノードを挿入」を連続して指示し，3 つのデータを[1, taka], [5, ueda], [2, konno]の順番で入力しました．このとき，次の問に答えなさい．

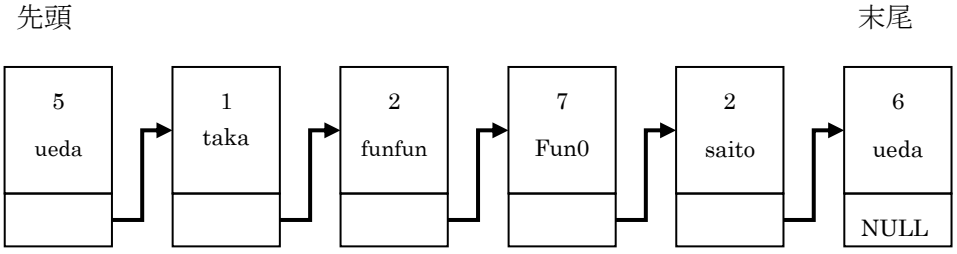
(ア) この状態での線形リストの各ノードの「番号」と「名前」，先頭から末尾に向かって順に答えてください．



(イ) この状態で，さらに「氏名で探索」で[ueda]を探索し，その後「着目ノードを削除」を指示しました．削除後の線形リストの各ノードの「番号」と「名前」，先頭から末尾に向かって順に答えてください．



(ウ) この状態で，さらに「番号で探索」で[2]を探索し，その後「着目ノードを削除」を指示しました．削除後の線形リストの各ノードの「番号」と「名前」，先頭から末尾に向かって順に答えてください．



アルゴリズムとデータ構造　授業中練習問題9

次のプログラムは「循環・重連結リストの実現例」である．このプログラムに関して，以下の問いに答えなさい．さらに，このプログラムを入力し，自分のパソコンでコンパイル，実行できることを確認してください．なお，プログラムの日本語部分は，英語，ローマ字に変更してかまいません．

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

#define MEMBER_NO       1       /* 番号を表す定数値 */
#define MEMBER_NAME     2       /* 氏名を表す定数値  */

/*--- 会員データ ---*/
typedef struct {
  int  no;        /* 番号 */
  char name[20];  /* 氏名 */
} Member;
/*--- ノード ---*/
typedef struct __node {
  Member        data;   /* データ */
  struct __node *prev;  /* 先行ノードへのポインタ */
  struct __node *next;  /* 後続ノードへのポインタ */
} Dnode;
/*--- 循環・重連結リスト ---*/
typedef struct {
  Dnode *head;  /* 先頭ダミーノードへのポインタ */
  Dnode *crnt;  /* 着目ノードへのポインタ */
} Dlist;
/*--- 会員の番号の比較関数 ---*/
int MemberNoCmp(const Member *x, const Member *y){
  return x->no < y->no ? -1 : x->no > y->no ? 1 : 0;
}
/*--- 会員の氏名の比較関数 ---*/
int MemberNameCmp(const Member *x, const Member *y){
  return strcmp(x->name, y->name);
}
/*--- 会員データ（番号と氏名）の表示（改行なし）---*/
void PrintMember(const Member *x){
  printf("%d %s", x->no, x->name);
}
/*--- 会員データ（番号と氏名）の表示（改行あり）---*/
void PrintLnMember(const Member *x){
  printf("%d %s¥n", x->no, x->name);
}
/*--- 会員データ（番号と氏名）の読込み ---*/
Member ScanMember(const char *message, int sw){
  Member temp;

  printf("%s するデータを入力してください。¥n", message);

  if (sw & MEMBER_NO)   { printf("番号："); scanf("%d", &temp.no); }
  if (sw & MEMBER_NAME) { printf("氏名："); scanf("%s", temp.name); }
```

```
  return temp;
}
/*--- 一つのノードを動的に生成 ---*/
static Dnode *AllocDnode(void){
  return calloc(1, sizeof(Dnode));
}
/*--- ノードの各メンバに値を設定 ----*/
static void SetDnode(Dnode *n, const Member *x,
                     const Dnode *prev, const Dnode *next){
  n->data = *x;   /* データ */
  n->prev = (Dnode *) prev; /* 先行ノードへのポインタ */
  n->next = (Dnode *) next; /* 後続ノードへのポインタ */
}
/*--- リストは空か ---*/
static int IsEmpty(const Dlist *list){
  return list->head->next == list->head;
}
/*--- リストを初期化 ---*/
void Initialize(Dlist *list){
  Dnode *dummyNode = AllocDnode();  /* ダミーノードを生成 */
  list->head = list->crnt = dummyNode;
  dummyNode->prev = dummyNode->next = dummyNode;
}
/*--- 着目ノードのデータを表示 ---*/
void PrintCurrent(const Dlist *list){
  if (IsEmpty(list))
    printf("着目要素はありません。");
  else
    PrintMember(&list->crnt->data);
}
/*--- 着目ノードのデータを表示（改行付き）---*/
void PrintLnCurrent(const Dlist *list){
  PrintCurrent(list);
  putchar('¥n');
}
/*--- 関数 compare によって x と一致すると判定されるノードを探索 ---*/
Dnode *Search(Dlist *list, const Member *x,
              int compare(const Member *x, const Member *y)){
  Dnode *ptr = list->head->next;

  while (ptr != list->head) {
    if (compare(&ptr->data, x) == 0) {
      list->crnt = ptr;
      return ptr;  /* 探索成功 */
    }
    ptr = ptr->next;
  }
  return NULL;     /* 探索失敗 */
}
/*--- 全ノードのデータをリスト順に表示 ---*/
void Print(const Dlist *list){
  if (IsEmpty(list))
    puts("ノードがありません。");
  else {
```

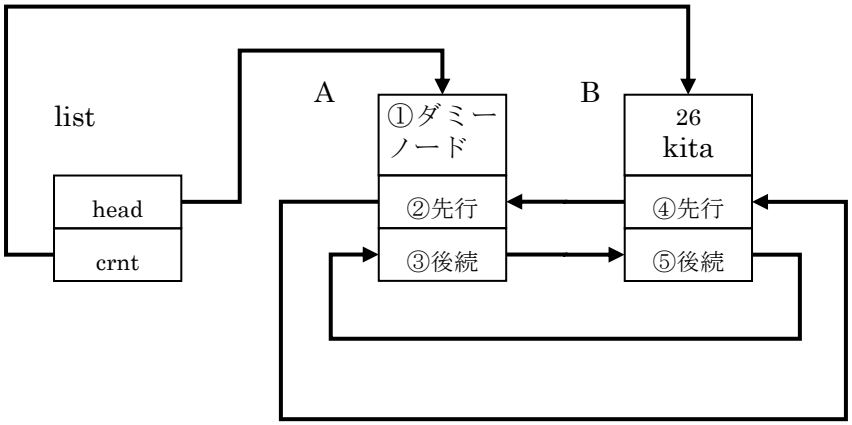
```
    Dnode *ptr = list->head->next;

    puts("【一覧表】");
    while (ptr != list->head) {
      PrintLnMember(&ptr->data);
      ptr = ptr->next;  /* 後続ノードに着目 */
    }
  }
}
/*--- p が指すノードの直後にノードを挿入 ---*/
void InsertAfter(Dlist *list, Dnode *p, const Member *x){
  Dnode *ptr = AllocDnode();
  Dnode *nxt = p->next;

  p->next = p->next->prev = ptr;
  SetDnode(ptr, x, p, nxt);
  list->crnt = ptr;     /* 挿入したノードに着目 */
}
/*--- 先頭にノードを挿入 ---*/
void InsertFront(Dlist *list, const Member *x){
  InsertAfter(list, list->head, x);
}
/*--- 末尾にノードを挿入 ---*/void
InsertRear(Dlist *list, const Member *x){
  InsertAfter(list, list->head->prev, x);
}
/*--- p が指すノードを削除 ---*/
void Remove(Dlist *list, Dnode *p){
  p->prev->next = p->next;
  p->next->prev = p->prev;
  list->crnt = p->prev;  /* 削除したノードの先行ノードに着目 */
  free(p);
  if (list->crnt == list->head)
    list->crnt = list->head->next;
}
/*--- 先頭ノードを削除 ---*/
void RemoveFront(Dlist *list){
  if (!IsEmpty(list))
    Remove(list, list->head->next);
}
/*--- 末尾ノードを削除 ---*/
void RemoveRear(Dlist *list){
  if (!IsEmpty(list))
    Remove(list, list->head->prev);
}
/*--- 着目ノードを削除 ---*/
void RemoveCurrent(Dlist *list){
  if (list->crnt != list->head)
    Remove(list, list->crnt);
}
/*--- 全ノードを削除 ---*/
void Clear(Dlist *list){
  while (!IsEmpty(list)) /* 空になるまで */
    RemoveFront(list);   /* 先頭ノードを削除 */
```

```
}
/*--- 循環・重連結リストを後始末 ---*/
void Terminate(Dlist *list){
  Clear(list);      /* 全ノードを削除 */
  free(list->head); /* ダミーノードを削除 */
}
/*--- メニュー ---*/
typedef enum {
  TERMINATE, INS_FRONT, INS_REAR,  RMV_FRONT, RMV_REAR, PRINT_CRNT,
  RMV_CRNT,  SRCH_NO,   SRCH_NAME, PRINT_ALL, CLEAR
} Menu;
/*--- メニュー選択 ---*/
Menu SelectMenu(void){
  int  i, ch;
  char *mstring[] = {
    "先頭にノードを挿入", "末尾にノードを挿入", "先頭のノードを削除",
    "末尾のノードを削除", "着目ノードを表示",  "着目ノードを削除",
    "番号で探索", "氏名で探索", "全ノードを表示", "全ノードを削除"
  };

  do {
    for (i = TERMINATE; i < CLEAR; i++) {
      printf("(%2d) %-18.18s  ", i + 1, mstring[i]);
      if ((i % 3) == 2)
        putchar('¥n');
    }
    printf("( 0) 終了  : ");
    scanf("%d", &ch);
  } while (ch < TERMINATE || ch > CLEAR);

  return (Menu)ch;
}
/*--- メイン ---*/
int main(void){
  Menu menu;
  Dlist list;

  Initialize(&list); /* 循環・重連結リストの初期化 */

  do {
    int n;
    Member x;
    Member *ptr;

    switch (menu = SelectMenu()) {
    case INS_FRONT :/* 先頭にノードを挿入 */
      x = ScanMember("先頭に挿入", MEMBER_NO | MEMBER_NAME);
      InsertFront(&list, &x);
      break;
    case INS_REAR :/* 末尾にノードを挿入 */
      x = ScanMember("末尾に挿入", MEMBER_NO | MEMBER_NAME);
      InsertRear(&list, &x);
      break;
    case RMV_FRONT : /* 先頭ノードを削除 */
```

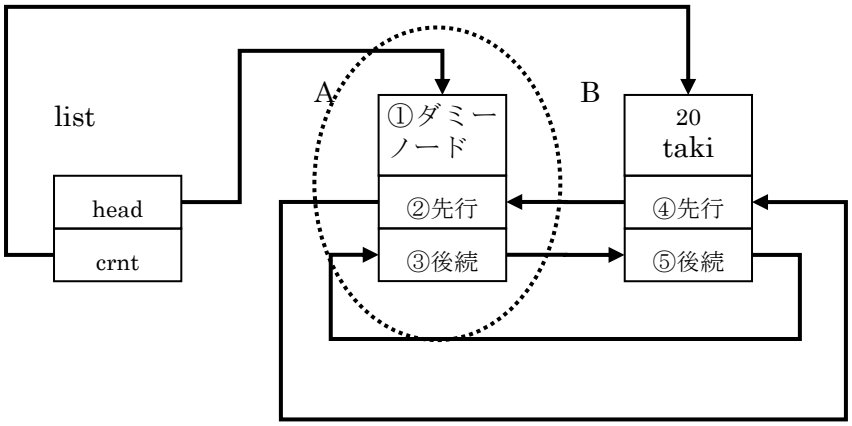
```
        RemoveFront(&list);
        break;
      case RMV_REAR : /* 末尾ノードを削除 */
        RemoveRear(&list);
        break;
      case PRINT_CRNT : /* 着目ノードのデータを表示 */
        PrintLnCurrent(&list);
        break;
      case RMV_CRNT :   /* 着目ノードを削除 */
        RemoveCurrent(&list);
        break;
      case SRCH_NO :    /* 番号による探索 */
        x = ScanMember("探索", MEMBER_NO);
        if (Search(&list, &x, MemberNoCmp) != NULL)
          PrintLnCurrent(&list);
        else
          puts("その番号のデータはありません。");
        break;
      case SRCH_NAME :  /* 氏名による探索 */
        x = ScanMember("探索", MEMBER_NAME);
        if (Search(&list, &x, MemberNameCmp) != NULL)
          PrintLnCurrent(&list);
        else
          puts("その名前のデータはありません。");
        break;
      case PRINT_ALL :   /* 全ノードのデータを表示 */
        Print(&list);
        break;
      case CLEAR : /* 全ノードを削除 */
        Clear(&list);
        break;
      }
    } while (menu != TERMINATE);

    Terminate(&list); /* 循環・重連結リストの後始末 */

    return 0;
}
```

1）このプログラムの動作直後に,「先頭にノードを挿入」を指示し，データ（番号：26, 氏名： kita）を入力しました．このとき，次の問いに答えなさい．

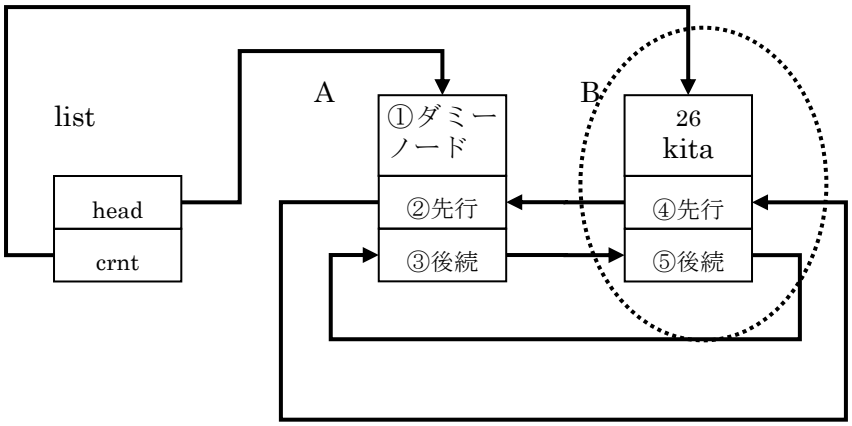
(ア) この状態での循環・重連結リストの各ノードの関係を示す図を以下に示す．この図中の空欄①～⑤に入る最も適切な語句を答えなさい．



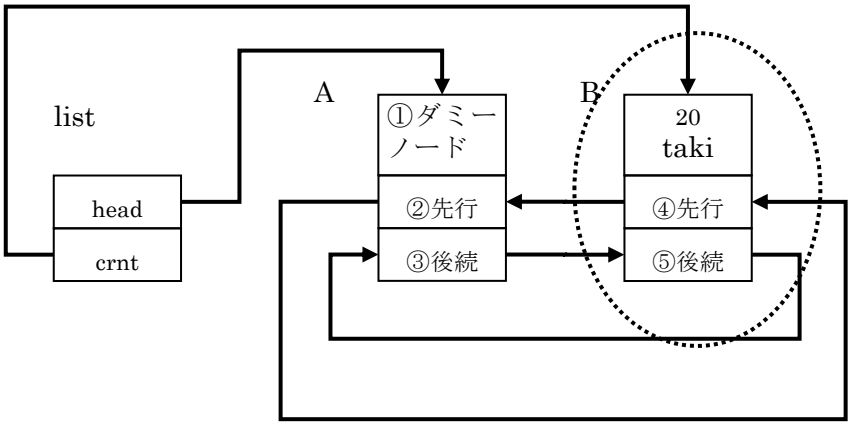
(イ) main 関数中の list.head が指すノードを(ア)の図の記号 A または B の記号で答えなさい.



(ウ) main 関数中の list.head ->next が指すノードを(ア)の図の記号 A または B の記号で答えない.

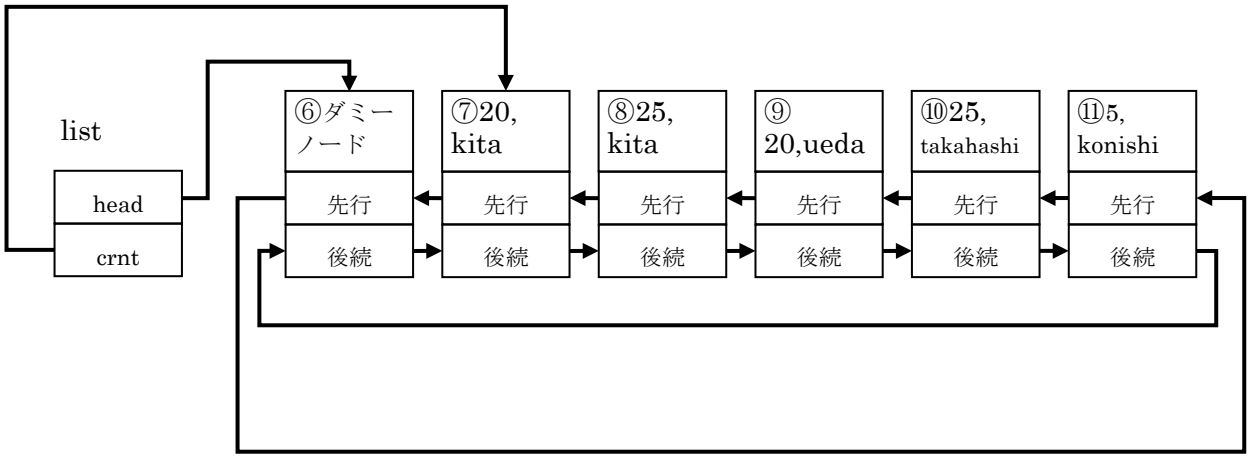


(エ) main 関数中の((list.head ->next) -> next)->prev が指すノードを(ア)の図の記号 A または B の記号で答えない.



2）このプログラムの動作直後に,「先頭にノードを挿入」を連続し 5 回指示し，5 つのデータを[5, konishi], [25，takahashi]，[20，ueda]，[25，kita]，[20, kita]の順番で入力しました．このとき，次の問に答えなさい．

(ア) この状態での循環・重連結リストの関係を示す図を以下に示す．この図の空欄⑥～⑪に入る最も適切な語句，またはノードのデータを答えなさい．



(イ) (ア)の状態で，さらに「氏名で探索」で[kita]を探索し，その後「着目ノードを削除」を指示しました．この時の削除されるノードを（ア)の図の⑥～⑪の記号で答えない．

⑦

(ウ) (ア)の状態で，さらに「番号で探索」で[20]を探索し，その後「着目ノードを削除」を指示しました．この時の削除されるノードを（ア)の図の⑥～⑪の記号で答えない．

⑦

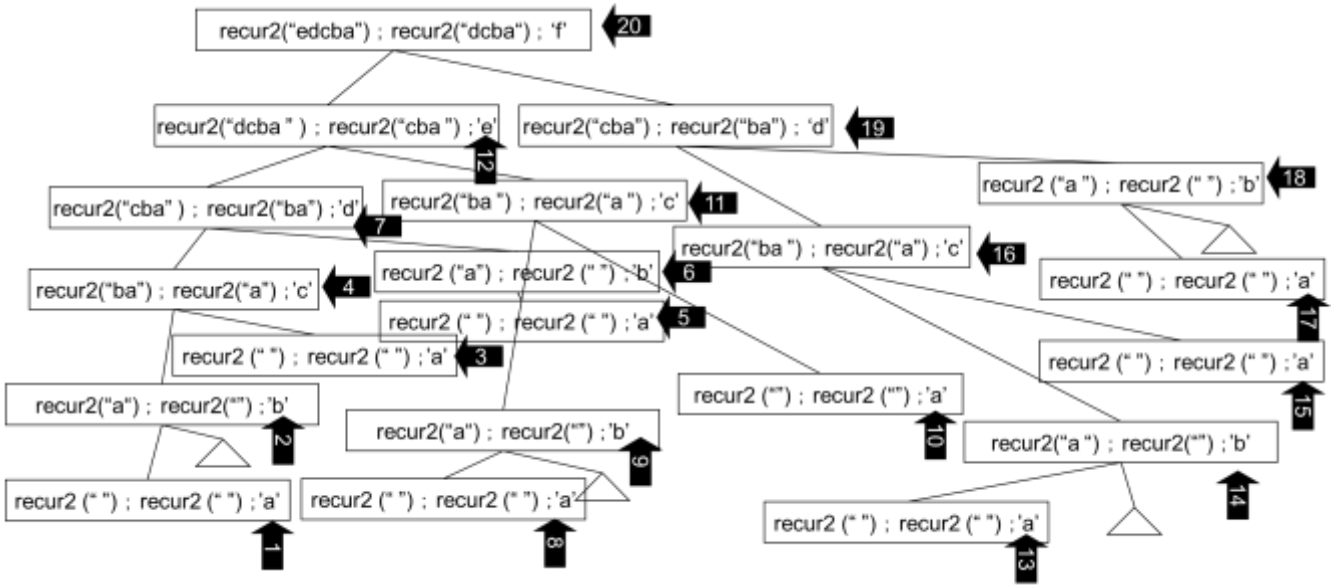
アルゴリズムとデータ構造　授業中練習問題10

次のプログラムは「再帰に対する理解を深めるための真に再帰的な関数2」である．このプログラムに関して，以下の問いに答えなさい．さらに，このプログラムを入力し，自分のパソコンでコンパイル，実行できることを確認してください．　なお，プログラムの日本語部分は，英語，ローマ字に変更してかまいません．

```
/*  再帰に対する理解を深めるための真に再帰的な関数 2 */
#include  <stdio.h>
#include  <string.h>

#define String_Max      82

/*--- 真に再帰的な関数 recur2 ---*/
void recur2(char *st)
{
  int  n = strlen(st);

  if( n  > 0 ){
   recur2( st + 1 );
   if (n >1 ) recur2( st + 2 );
   else recur2( st + 1 );
   printf("%c", *st);
  }
}

int main(void){
  char  x[String_Max];

  printf("文字列をを入力せよ：");
  scanf("%s", x);
  recur2(x);
  printf("¥n");

  return (0);
}
```

このプログラムの動作直後に文字列の”fedcba”を入力しました．このとき，次の問に答えなさい．

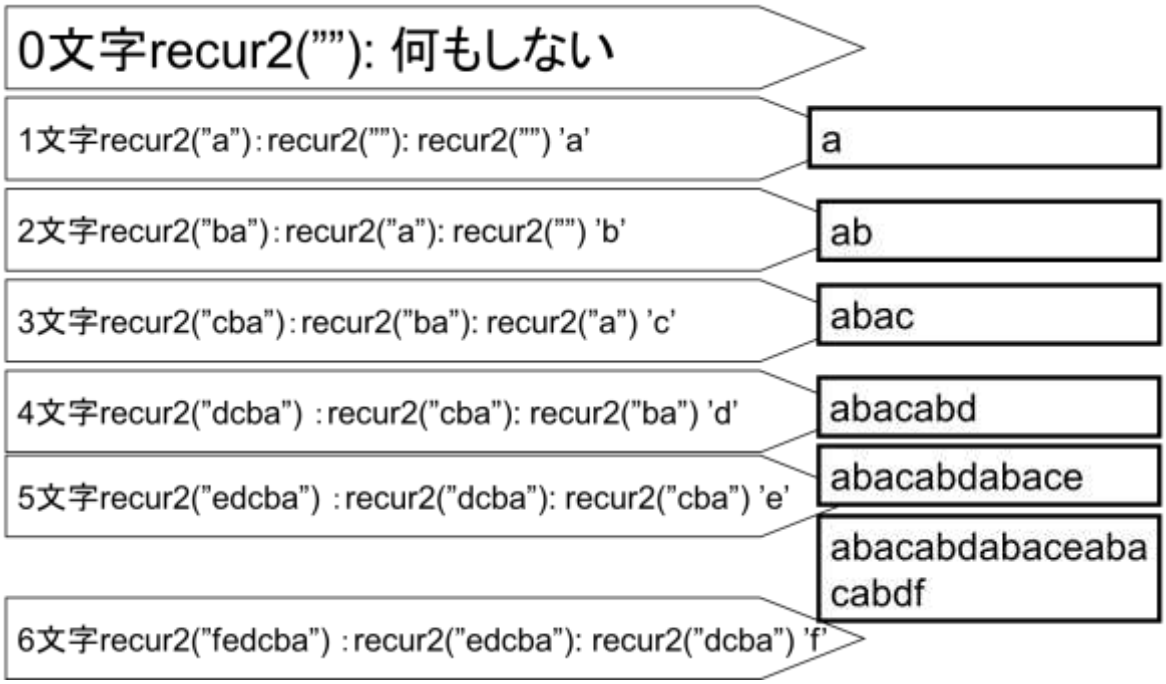
(ア) 教科書１８１ページのトップダウン解析を用いて，関数 recur2 の動作を解析し，出力する結果を求めなさい．



出力：abacabdabaceabacabdf

(イ) 教科書１８２ページのボトムアップ解析を用いて，プログラムが出力する結果を示しなさい．

出力：abacabdabaceabacabdf



アルゴリズムとデータ構造　授業中練習問題11

次のプログラムは「構造体の単純交換ソートの実現例」(教科書の List6-1 を構造体に拡張)である．このプログラムに関して，以下の問いに答えなさい．さらに，このプログラムを入力し，自分のパソコンでコンパイル，実行できることを確認してください．　なお，プログラムの日本語部分は，英語，ローマ字に変更してかまいません．

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

#define swap(type, x, y) do {type t; t = x; x = y; y = t;} while(0)

#define  ASCENDING      0 /* 昇順 */
#define  DESCENDING     1 /* 降順 */

/*--- 会員データ ---*/
typedef struct {
  int  no;        /* 番号 */
  char name[20];  /* 氏名 */
} Member;

/*--- 会員の番号の比較関数 ---*/
int MemberNoCmp(const Member *x, const Member *y){
  return x->no < y->no ? -1 : x->no > y->no ? 1 : 0;
}
/*--- 会員の氏名の比較関数 ---*/
int MemberNameCmp(const Member *x, const Member *y){
  return strcmp(x->name, y->name);
}
/*--- 会員データ（番号と氏名）の表示（改行あり）---*/
void PrintLnMember(const Member *x){
  printf("%d %s¥n", x->no, x->name);
}

/* --- 単純交換ソート --- */
void bubble(Member *a, int n,
            int compare(const Member *x, const Member *y),
            int order){
  int i, j;

  for(i = 0; i < n - 1; i++){
    for(j = n-1; j > i; j--)
      if (compare(a + j - 1 + order, a + j - order) > 0)
        swap(Member, a[j-1], a[j]);
  }
}

/*--- 全データの表示 ---*/
void Print(const Member *data, int n){
  int i;
```

```
  for(i=0; i < n; i++)
    PrintLnMember(data+i);
}

/*--- メニュー ---*/
typedef enum {
  TERMINATE, ASCEND_NO, ASCEND_NAME,
  DESCEND_NO, DESCEND_NAME, PRINT_ALL
} Menu;

/*--- メニュー選択 ---*/
Menu SelectMenu(void){
  int  i, ch;
  char *mstring[] = {
    "番号で昇順ソート", "名前で昇順ソート",
    "番号で降順ソート", "名前で降順ソート",
    "データを表示"
  };

  do {
    for (i = TERMINATE; i < PRINT_ALL; i++) {
      printf("(%2d) %-24.24s  ", i + 1, mstring[i]);
      if ((i % 3) == 2)
        putchar('¥n');
    }
    printf("( 0) 終了  :");
    scanf("%d", &ch);
  } while (ch < TERMINATE || ch > PRINT_ALL);

  return (Menu)ch;
}

/*--- メイン ---*/
int main(void){
  Menu menu;
  Member data[] = {
   {5, "watanabe"},     {7, "satoshi"},
   {6, "noyuri"},    {0, "daisuke"},
   {0, "motoko"},    {4, "agemi"},
   {9, "ito"},       {2, "ohta"},
   {1, "takashi"},   {3, "kouji"}
  };
  int ndata = sizeof(data)/sizeof(data[0]);

  do {
    int n;

    switch (menu = SelectMenu()) {
    case ASCEND_NO : /* 番号で昇順にソート */
      bubble(data, ndata, MemberNoCmp, ASCENDING);
      break;
    case ASCEND_NAME :/* 名前で昇順にソート */
```

```
      bubble(data, ndata, MemberNameCmp, ASCENDING);
      break;
    case DESCEND_NO : /* 番号で降順にソート */
      bubble(data, ndata, MemberNoCmp, DESCENDING);
      break;
    case DESCEND_NAME :/* 名前で降順にソート */
      bubble(data, ndata, MemberNameCmp, DESCENDING);
      break;
    case PRINT_ALL :  /* 全データを表示 */
      Print(data, ndata);
      break;
    }
  } while (menu != TERMINATE);

  return 0;
}
```

1) このプログラムの動作直後に, 数字の 3 を入力したときの data->name が指している文字を示しなさい.

i

2) このプログラムの動作直後に, 数字の 5 を入力したときの data->name が指している文字列を示しなさい.

watanabe

3) このプログラムの動作直後に, 数字の 3 を入力したときの bubble 関数中の compare(a+j-1+order, a+j-order)が呼び出される回数は何回ですか.

45 (= 10×9／2)

4） このプログラムの動作直後に，数字の 3 を入力したときの bubble 関数中で i=0 のとき，swap(Member , a[j-1], a[j]) が呼び出される回数は何回ですか．

8 回

5） このプログラムの動作直後に，数字の 4 を入力したときの data->name が指している文字列を示しなさい．

watanabe

6） 5)の状態から更に，数字の 2 を入力したときの data->name が指している文字列を示しなさい．

agemi

7） 5)の状態から更に，数字の 2 を入力したときの bubble 関数中の compare(a+j-1+order, a+j-order) が呼び出される回数は何回ですか．

45 （ =10×9／2）

８） 5)の状態から更に，数字の 2 を入力したときの bubble 関数中で i=0 のとき，swap(Member , a[j-1], a[j])が呼び出される回数は何回ですか．

9 回

アルゴリズムとデータ構造　授業中練習問題１２

次のプログラムは「構造体のクイックソートの実現例」(教科書の List6-9 を構造体版に拡張) である．このプログラムに関して，以下の問いに答えなさい．さらに，このプログラムを入力し，自分のパソコンでコンパイル，実行できることを確認してください． なお，プログラムの日本語部分は，英語，ローマ字に変更してかまいません．

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

#define swap(type, x, y) do {type t; t = x; x = y; y = t;} while(0)

/*--- 会員データ ---*/
typedef struct {
  int  no;         /* 番号 */
  char name[20];  /* 氏名 */
} Member;

/*--- 会員の番号の昇順比較関数 ---*/
int AscendingMemberNoCmp(const Member *x, const Member *y){
  return x->no < y->no ? -1: x->no > y->no ? 1: 0;
}
/*--- 会員の番号の降順比較関数 ---*/
int DescendingMemberNoCmp(const Member *x, const Member *y){
  return x->no < y->no ? 1: x->no > y->no ? -1: 0;
}
/*--- 会員の氏名の昇順比較関数 ---*/
int AscendingMemberNameCmp(const Member *x, const Member *y){
  return strcmp(x->name, y->name);
}
/*--- 会員の氏名の降順比較関数 ---*/
int DescendingMemberNameCmp(const Member *x, const Member *y){
  return strcmp(y->name, x->name);
}
/*--- 会員データ（番号と氏名）の表示（改行あり）---*/
void PrintLnMember(const Member *x){
  printf("%d %s¥n", x->no, x->name);
}
/*--- 全データの表示 ---*/
void Print(const Member *data, int n){
  int i;

  for(i=0; i < n; i++)
    PrintLnMember(data+i);
}
/* --- クイックソート --- */
void quick(Member *a, int left, int right,
           int compare(const Member *y, const Member *z)){
  int pl = left;
  int pr = right;
  Member x = a[(pl+pr)/2];

  do {
    while(compare(&x, a+pl)>0) pl++;
    while(compare(a+pr, &x)>0) pr--;
    if ( pl <= pr ){
      swap(Member , a[pl], a[pr]);
```

```
      pl++;
      pr--;
    }
  } while(pl <= pr );
```


```
  if ( left < pr ) quick(a, left, pr,    compare);
  if ( pl < right) quick(a, pl,   right, compare);
}

/*--- メニュー ---*/
typedef enum {
  TERMINATE, ASCEND_NO, ASCEND_NAME,
  DESCEND_NO, DESCEND_NAME, PRINT_ALL
} Menu;

/*--- メニュー選択 ---*/
Menu SelectMenu(void){
  int  i, ch;
  char *mstring[] = {
    "番号で昇順ソート", "名前で昇順ソート",
    "番号で降順ソート", "名前で降順ソート",
    "データを表示"
  };

  do {
    for (i = TERMINATE; i < PRINT_ALL; i++) {
      printf("(%2d) %-18.18s  ", i + 1, mstring[i]);
      if ((i % 3) == 2)
        putchar('¥n');
    }
    printf("( 0) 終了 :");
    scanf("%d", &ch);
  } while (ch < TERMINATE || ch > PRINT_ALL);

  return (Menu)ch;
}

/*--- メイン ---*/
int main(void){
  Menu menu;
  Member data[] = {
   {1, "takashi"},  {3, "kouji"},    {5, "umeda"},    {7, "satoshi"},
   {6, "noyuri"},   {2, "daisuke"}, {0, "motoko"},   {4, "agemi"},
   {9, "ito"},      {2, "ohta"}
};

  int ndata = sizeof(data)/sizeof(data[0]);

  do {
    switch (menu = SelectMenu()) {
    case ASCEND_NO : /* 番号で昇順にソート */
      quick(data, 0, ndata-1, AscendingMemberNoCmp);
      break;
    case ASCEND_NAME :/* 名前で昇順にソート */
      quick(data, 0, ndata-1, AscendingMemberNameCmp);
      break;
    case DESCEND_NO : /* 番号で降順にソート */
      quick(data, 0, ndata-1, DescendingMemberNoCmp);
      break;
    case DESCEND_NAME :/* 名前で降順にソート */
      quick(data, 0, ndata-1, DescendingMemberNameCmp);
```

```
      break;
    case PRINT_ALL :  /* 全データを表示 */
      Print(data, ndata);
      break;
    }
  } while (menu != TERMINATE);

  return 0;
}
```

1） このプログラムの動作直後に，数字の1を入力しました．このとき，次の問に答えなさい．

（ア）main 関数から呼び出された quick 関数の枢軸に選択された要素を示しなさい．

a[4]: : 6, "noryuri"

（イ）main 関数から呼び出された quick 関数で，swap(Member, a[pl], a[pr])が呼び出される回数は何回ですか．

2回

（ウ）main 関数から呼び出された quick 関数の矢印①の場所における，pl, pr, left, right の値はいくらですか．

pl: 7, pr: 6, left: 0, right:9

（エ）quick 関数はソートが終わるまで何回呼ばれますか．

8 回

2） このプログラムの動作直後に，数字の4を入力しました．このとき，次の問に答えなさい．

（ア） main 関数から呼び出された quick 関数の枢軸に選択された要素を示しなさい．

a[4]：：6, ″noryuri″

（イ）main 関数から呼び出された quick 関数で， swap(Member, a[pl], a[pr])が呼び出される回数は何回ですか．

2 回

（ウ） main 関数から呼び出された quick 関数の矢印①の場所における，pl, pr, left, right の値はいくらですか．

pl: 5, pr:3, left: 0, right:9

（エ） quick 関数はソートが終わるまで何回呼ばれますか．

8 回

アルゴリズムとデータ構造　授業中練習問題13

次に示す「構造体に対するヒープソートの実現例」(教科書の List6-15 を構造体版に拡張)のある．このプログラムに関して，以下の問いに答えなさい．さらに，このプログラムを入力し，自分のパソコンでコンパイル，実行できることを確認してください．　なお，プログラムの日本語部分は，英語，ローマ字に変更してかまいません．

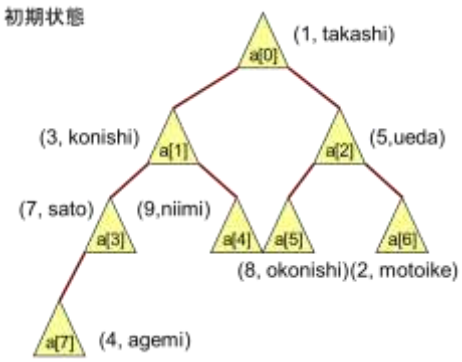
```
/* 構造体に対するヒープソートの実現例 */
#include  <stdio.h>
#include  <stdlib.h>
#include  <string.h>

#define swap(type, x, y) do {type t; t = x; x = y; y = t;} while(0)

/*--- 会員データ ---*/
typedef struct {
  int  no;         /* 番号 */
  char name[20];  /* 氏名 */
} Member;
/*--- 会員の番号の昇順比較関数 ---*/
int AscendingMemberNoCmp(const Member *x, const Member *y){
  return x->no < y->no ? -1: x->no > y->no ? 1: 0;
}
/*--- 会員の番号の降順比較関数 ---*/
int DescendingMemberNoCmp(const Member *x, const Member *y){
  return x->no < y->no ? 1: x->no > y->no ? -1: 0;
}
/*--- 会員の氏名の昇順比較関数 ---*/
int AscendingMemberNameCmp(const Member *x, const Member *y){
  return strcmp(x->name, y->name);
}
/*--- 会員の氏名の降順比較関数 ---*/
int DescendingMemberNameCmp(const Member *x, const Member *y){
  return strcmp(y->name, x->name);
}
/*--- 会員データ（番号と氏名）の表示（改行あり）---*/
void PrintLnMember(const Member *x){
  printf("%d %s¥n", x->no, x->name);
}
/*--- 全データの表示 ---*/
void Print(const Member *data, int n){
  int i;

  for(i=0; i < n; i++)
    PrintLnMember(data+i);
}
/*--- a[left]～a[right]をヒープ化 ---*/
static void updownheap(Member *a, int left, int right,
       int compare(const Member *y, const Member *z)){

  Member temp = a[left]; /* 根 */
  int child;
  int parent;
```

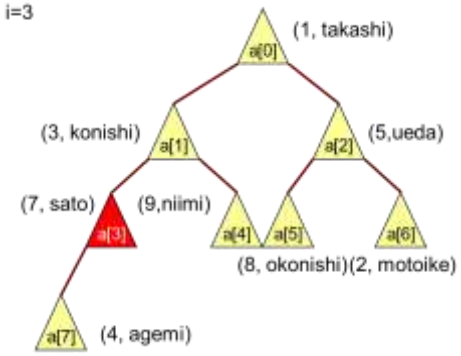
```
  for (parent = left; parent < (right + 1)/2; parent = child) {
    int  cl = parent * 2 + 1; /* 左の子 */
    int  cr = cl + 1;         /* 右の子 */
    child = (cr <= right  && compare(a + cr, a + cl) >0 ) ? cr : cl;
    /* 昇順なら大きい方, 降順なら小さい方 */
    if (compare (&temp, a + child) >= 0)        ←――――― ①
      break;
    a[parent] = a[child];
  }
  a[parent] = temp;
}
/*--- ヒープソート ---*/
void heapsort(Member *a, int n, int compare(const Member *y, const Member *z)){
  int    i;

  for (i = (n - 1) / 2; i >= 0; i--)
    updownheap(a, i, n - 1, compare); ←――――― ②

  for (i = n - 1; i > 0; i--) {
    swap(Member , a[0], a[i]);
    updownheap(a, 0, i - 1, compare);
  }
}
/*--- メニュー ---*/
typedef enum {
  TERMINATE, ASCEND_NO, ASCEND_NAME, DESCEND_NO, DESCEND_NAME, PRINT_ALL
} Menu;
/*--- メニュー選択 ---*/
Menu SelectMenu(void){
  int  i, ch;
  char *mstring[] = {
    "番号で昇順ソート", "名前で昇順ソート",
    "番号で降順ソート", "名前で降順ソート",
    "データを表示"
  };

  do {
    for (i = TERMINATE; i < PRINT_ALL; i++) {
      printf("(%2d) %-22.22s  ", i + 1, mstring[i]);
      if ((i % 3) == 2)
         putchar('¥n');
    }
    printf("( 0) 終了 :");
    scanf("%d", &ch);
  } while (ch < TERMINATE || ch > PRINT_ALL);

  return (Menu)ch;
}
/*--- メイン ---*/
int main(void){
  Menu menu;
```

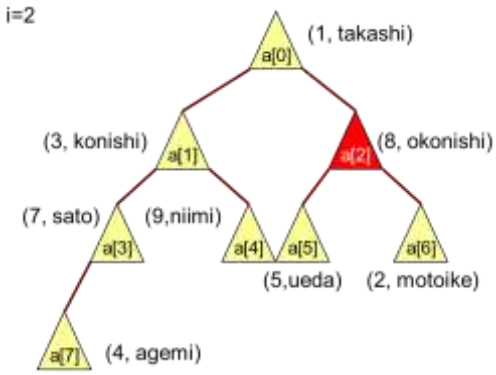
```
  Member data[] = {
    Member data[] = {
   {1, "takahashi"},{3, "konishi"},  {5, "ueda"},      {7, "sato"},
   {9, "niimi"},     {8, "okonishi"}, {2, "motoike"},  {4, "agemi"}
  };
  int ndata = sizeof(data)/sizeof(data[0]);

  do {
    switch (menu = SelectMenu()) {
    case ASCEND_NO : /* 番号で昇順にソート */
      heapsort(data, ndata, AscendingMemberNoCmp);
      break;
    case ASCEND_NAME :/* 名前で昇順にソート */
      heapsort(data, ndata, AscendingMemberNameCmp);
      break;
    case DESCEND_NO : /* 番号で降順にソート */
      heapsort(data, ndata, DescendingMemberNoCmp);
      break;
    case DESCEND_NAME :/* 名前で降順にソート */
      heapsort(data, ndata, DescendingMemberNameCmp);
      break;
    case PRINT_ALL :  /* 全データを表示 */
      Print(data, ndata);
      break;
    }
  } while (menu != TERMINATE);
  return 0;
}
```

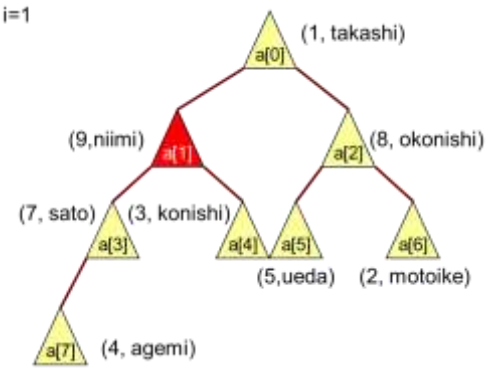
1） このプログラムの動作直後に，数字の 1 を入力しました．このとき，次の問いに答えなさい．

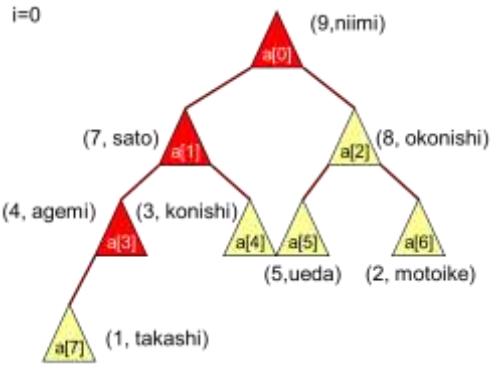
（ア）この状態で，矢印②の for ループ中の関数 updownheap()が呼出された直後の構造体配列 a の値を，i の値毎に示しなさい．









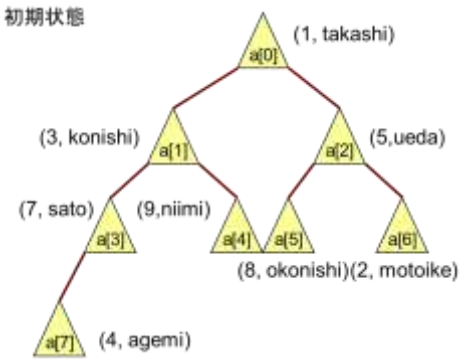


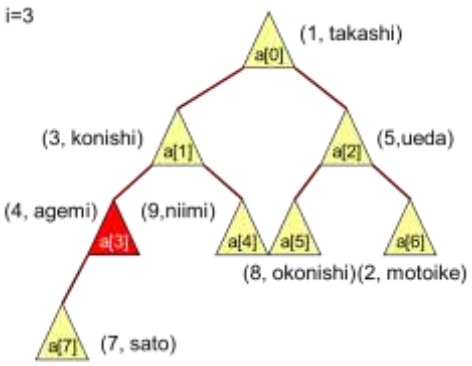
（イ）この状態で，矢印②直前の for ループが繰り返されている間に，関数 updownheap()中の矢印①の if 文が実行される回数は何回ですか．

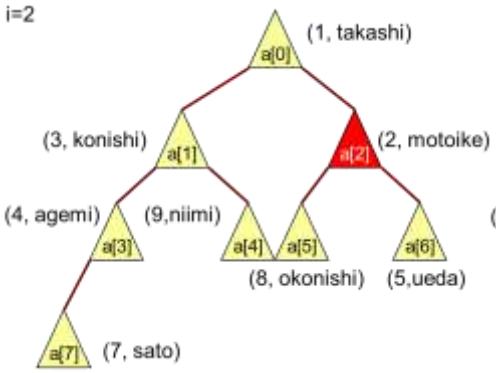
6 回

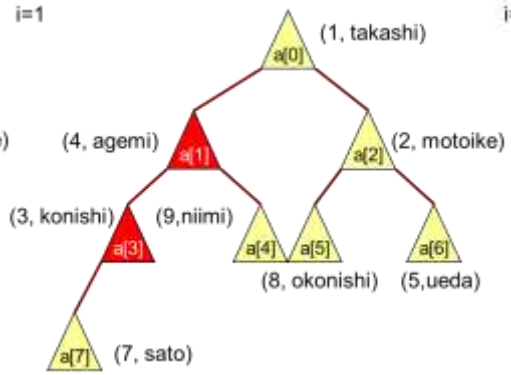
2） このプログラムの動作直後に，数字の4を入力しました．このとき，次の問いに答えなさい．

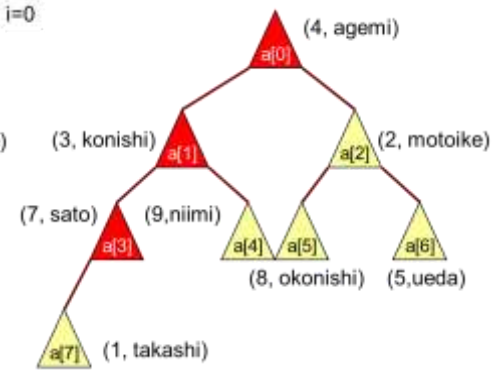
(ア) この状態で，矢印②の for ループ中の関数 updownheap()が呼出された直後の構造体配列 a の値を，i の値毎に示しなさい．











(イ) この状態で，矢印②直前の for ループが繰り返されている間に，関数 updownheap()中の矢印①の if 文が実行される回数は何回ですか．

7 回