アルゴリズムとデータ構造　授業中練習問題13

次のプログラムは「２分探索木の実現例」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．

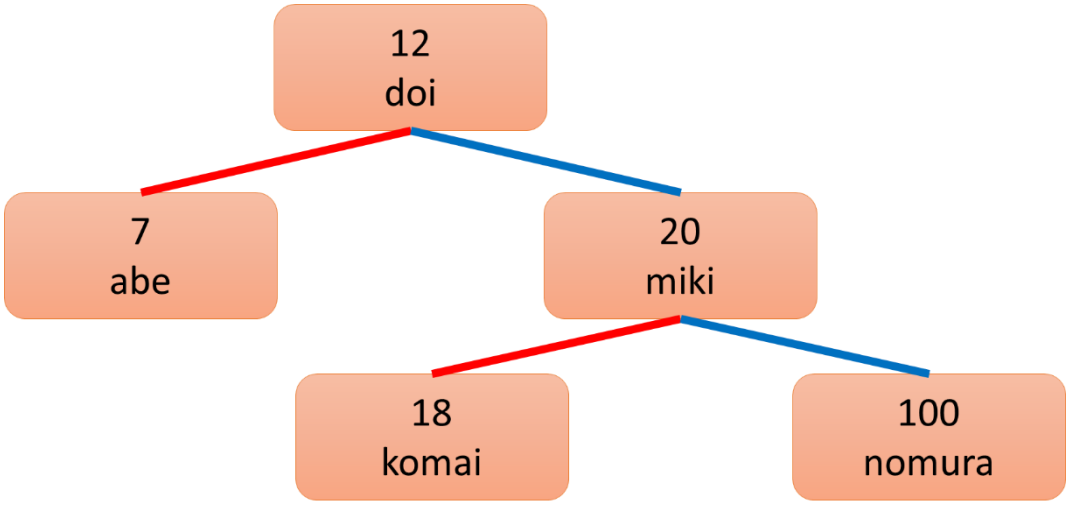
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#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 __bnode {
  Member          data; /* データ */
  struct __bnode *left; /* 左子ノードへのポインタ */
  struct __bnode *right;/* 右子ノードへのポインタ */
} BinNode;
/*--- 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);
}
/*--- 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確保 ---*/
static BinNode *AllocBinNode(void){
  return calloc(1, sizeof(BinNode));
}
```

```
/*--- ノードの各メンバに値を設定 ----*/
static void SetBinNode(BinNode *n, const Member *x,
                       const BinNode *left, const BinNode *right){
  n->data  = *x;   /* データ */
  n->left  = (BinNode *)left; /* 左子ノードへのポインタ */
  n->right = (BinNode *)right;/* 右子ノードへのポインタ */
}
/*--- 探索 ---*/
BinNode *Search(BinNode *p, const Member *x){
  int cond;

  if (p == NULL)
    return NULL;        /* 探索失敗 */
  else if ((cond = MemberNameCmp(x, &p->data)) == 0)
    return p;           /* 探索成功 */
  else if (cond < 0)
    return Search(p->left,  x); /* 左部分木から探索 */
  else
    return Search(p->right, x); /* 右部分木から探索 */
}
/*--- ノードを挿入 ---*/
BinNode *Add(BinNode *p, const Member *x){
  int cond;

  if (p == NULL) {
    p = AllocBinNode();
    SetBinNode(p, x, NULL, NULL);
  } else if ((cond = MemberNameCmp(x, &p->data)) == 0)
    printf("【エラー】%s は既に登録されています。¥n", x->name);
  else if (cond < 0)
    p->left  = Add(p->left, x);
  else
    p->right = Add(p->right, x);
  return p;
}
/*--- ノードを削除 ---*/
int Remove(BinNode **root, const Member *x){
  BinNode *next, *temp;
  BinNode **left;
  BinNode **p = root;

  while (1) {
    int cond;
    if (*p == NULL) {
      printf("【エラー】%s は登録されていません。¥n", x->name);
      return -1;          /* そのキーは存在しない */
    } else if ((cond = MemberNameCmp(x, &(*p)->data)) == 0)
      break;              /* 探索成功 */
    else if (cond < 0)
      p = &((*p)->left); /* 左部分木から探索 */
    else
      p = &((*p)->right);/* 右部分木から探索 */
  }
  if ((*p)->left == NULL)
```

```
    next = (*p)->right;
  else {
    left = &((*p)->left);
    while ((*left)->right != NULL)
      left = &(*left)->right;
    next = *left;
    *left = (*left)->left;
    next->left  = (*p)->left;
    next->right = (*p)->right;
  }
  temp = *p;
  *p = next;
  free(temp);

  return 0;
}

/*--- 全ノードのデータを表示 ---*/
void PrintTree(const BinNode *p){
  if (p != NULL) {
    PrintTree(p->left);
    PrintLnMember(&p->data);
    PrintTree(p->right);
  }
}
/*--- 全ノードの削除 ---*/
void FreeTree(BinNode *p){
  if (p != NULL) {
    FreeTree(p->left);
    FreeTree(p->right);
    free(p);
  }
}
/*--- メニュー ---*/
typedef enum {
  TERMINATE, ADD, REMOVE, SEARCH, PRINT_ALL
} Menu;
/*--- メニュー選択 ---*/
Menu SelectMenu(void){
  int  i, ch;
  char *mstring[] = {"挿入", "削除", "探索", "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;
}
```

```
/*--- メイン関数 ---*/
int main(void){
  Menu    menu;
  BinNode *root = NULL; /* ２分探索木の根へのポインタ */

  do {
    Member x;
    BinNode *temp;

    switch (menu = SelectMenu()) {
      /*--- ノードの挿入 ---*/
    case ADD :
      x = ScanMember("挿入", MEMBER_NO | MEMBER_NAME);
      root = Add(root, &x);
      break;
      /*--- ノードの削除 ---*/
    case REMOVE :
      x = ScanMember("削除", MEMBER_NAME);
      Remove(&root, &x);
      break;
      /*--- ノードの探索 ---*/
    case SEARCH :
      x = ScanMember("探索", MEMBER_NAME);
      if ((temp = Search(root, &x)) != NULL)
        PrintLnMember(&temp->data);
      break;
      /*--- 全ノードの表示 ---*/
    case PRINT_ALL :
      puts("【一覧表】");
      PrintTree(root);
      break;
    }
  } while (menu != TERMINATE);

  FreeTree(root);

  return 0;
}
```

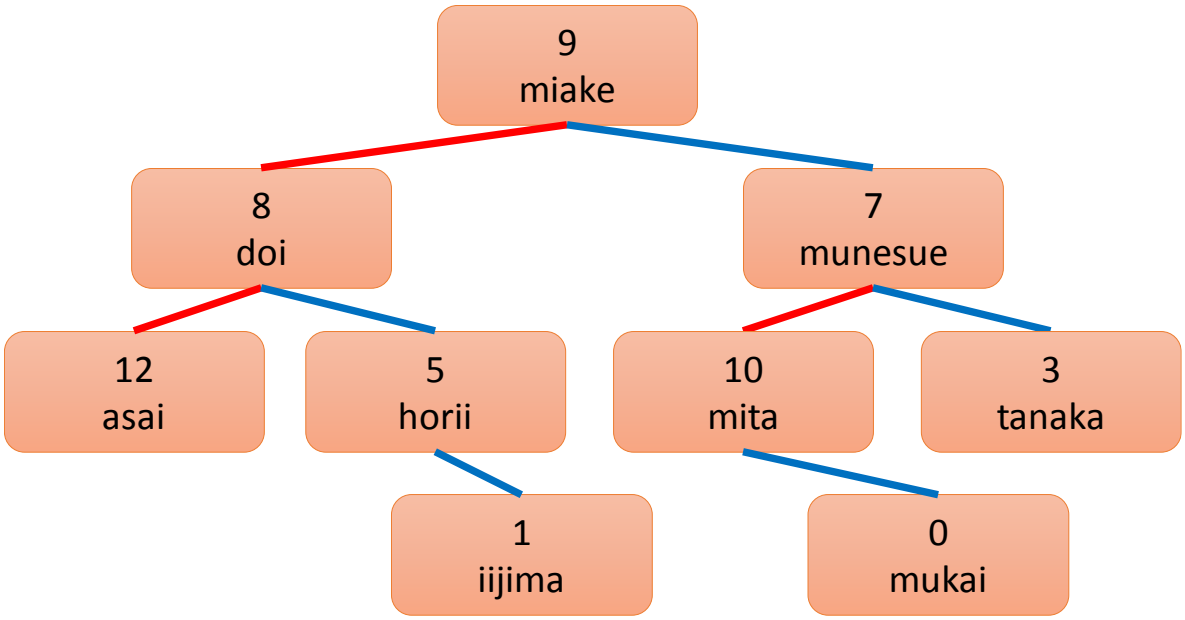
1）このプログラムの動作直後に，「挿入」を連続して 5 回指示した結果，以下の図に示す 2 分探索木が得られた．このとき，次の問に答えなさい．



(ア) 上図の 2 分探索木を得るためには，どのような順番でデータを入力する必要があるのか，入力するデータの順番の例を二つ以上答えなさい．

(イ) この状態で，キーの値を”"nomura”として「探索」を指示しました．どのような順序で探索が行われるかを，探索のする順番を答えない．

2）このプログラムの動作直後に，「挿入」を連続して 9 回指示した結果，以下の図に示す 2 分探索木が得られた．このとき，次の問に答えなさい．



(ア) この状態の 2 分探索木に対し，「帰りがけ順」でノードの表示を行うとすれば，どのように表示されますか．

(イ) この状態の 2 分探索木に対し，「通りがけ順」でノードの表示を行うとすれば，どのように表示されますか．