```
/*****************************************************************************
 *    ghostBuster.c
 *
 *        This program is a sample database of ghostbuster.
 *        There are display all of events, add new event, search event to modify
 *        or delete and dump file.
 *
 *    Created by Setthawut Leelawatthanapanit(Saab) ID : 3466
 *        18 NOVEMBER 2017
 *
 *****************************************************************************
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "ghostBuster.h"

/* Main functions. Get the menu from user until the user exits the program. */
 int main()
    {
    DATE_T date;              /* Get date today. */
    char input[LENGTH] = {0}; /* Get input from the terminal. */
    int choice = 0;           /* Store value to choose the case. */
    controlDatabase(1);
    printf("Welcome to RUN RAN RUN Ghostbuster Company!\n");
    dateToday(&date);
    printf("Time: %02d-%02d-%02d %02d:%02d\n\n", date.day, date.month, date.year,
            date.hour, date.minute);
    while(1)
        {
        printf("Menu :\n");
        printf("    1 - DISPLAY ALL EVENT\n");
        printf("    2 - ADD NEW EVENT\n");
        printf("    3 - SEARCH EVENT\n");
        printf("    4 - DUMP FILE\n");
        printf("    5 - EXIT\n");
        printf("What options do you want?: ");
        getMenu();
        }
    exit(0);
    }
```

```
/* PUBLIC FUNCTION. This function gets menu from users
 * and then call function that the users want to do.
 */
void getMenu()
    {
    char input[LENGTH] = {0}; /* Get input from the terminal. */
    int choice = 0;           /* Keep value to select case. */
    while(1)
        {
        memset(input, 0, sizeof(input));
        fgets(input, sizeof(input), stdin);
        if(strlen(input) == 2)
             {
             if(input[0] == '1')
             { /* User selects showing all of information. */
             printAllEvent();
             break;
             }
             else if(input[0] == '2')
             { /* User selects adding information. */
             controlAdd();
             break;
             }
             else if(input[0] == '3')
             { /* User selects searching information. */
             controlSearch();
             break;
             }
             else if(input[0] == '4')
             { /* Use selects dumping file. */
             controlDump();
             break;
             }
             else if(input[0] == '5')
             { /* User selects exiting the program. */
             controlExit();
             break;
             }
            }
        /* User gets otherwise. */
        printf("#SYSTEM: Input is invalid, try again: ");
        }
    }
```

```
/* PUBLIC FUNCTION. This function is called when user wants to add information.
 * Get information from the user. During add information,
 * If the user hits to return, program will go to main function to get a menu.
 */
void controlAdd()
    {
    EVENT_T event;            /* Struct of the information. */
    char input[LENGTH] = {0}; /* Get input from the terminal. */
    int status = 0;     /* Keep value correction after validate information. */
    printf("\n==================== Add information ====================\n\n");
    status = askDate(&event.dateEvent, NULL);
    /* Users get information before. */
    if(status)
        status = askName(event.nameReport, 1);
    if(status)
        status = askPhone(event.phoneReport);
    if(status)
        status = askType(event.typeEvent);
    if(status)
        status = askLocation(&event.latitude, 1);
    if(status)
        status = askLocation(&event.longitude, 2);
    if(status)
        status = askDate(&event.dateInvest, &event.dateEvent);
    if(status)
        status = askName(event.nameInvest, 2);
    if(status)
        status = askResult(&event.result);
    if(status == 0)
        { /* If user hits <CR>, go back to menu. */
        printf("\n#SYSTEM: Cancel add new event.\n");
        printf("#SYSTEM: Go back to menu.\n");
        printf("\n============================================================\n\n");
        return;
        }
    event.eventCode[0] = event.dateEvent.year;
    event.eventCode[1] = runEventCode(event.dateEvent.year);
    if (event.eventCode[1] != 0)
        { /* Can add new data. */
        printf("\nDisplay new event.\n");
        printEvent(event);
        addEvent(event);
        }
    else if (event.eventCode[1] == 0)
        { /* Data is maximum right now. */
        printf("\n#SYSTEM: Can't add new data anymore.\n");
        printf("\nThe event code is maximum right now.\n");
        }
    printf("============================================================\n\n");
    return;
    }
```

```
/* PUBLIC FUNCTION. This is main search function. It is called when user wants
 * to use search function. Ask the user how to search until the user hits to
 * return to main function to main function.
 */
void controlSearch()
    {
    char input[LENGTH] = {0}; /* Get input from the terminal. */
    printf("\n=================== Search information ==================\n");
    while(1)
        {
        printf("\n    Select how to search that you want:\n");
        printf("         1 - Search by event code\n");
        printf("         2 - Search by others\n");
        printf("    What do you want to search? (Hit <CR> to return): ");
        memset(input, 0, sizeof(input));
        fgets(input, sizeof(input), stdin);
        if (strlen(input) == 1)
            { /* User hits to return to main function. */
            printf("\n============================================================\n\n");
            return;
            }
        else if ((input[0] == '1') && (strlen(input) == 2))
            controlSearchCode();   /* User selects searching by event code. */
        else if ((input[0] == '2') && (strlen(input) == 2))
            controlSearchOthers(); /* User selects searching by others. */
        else                       /* User selects otherwise. */
            printf("#SYSTEM: Input is invalid, try again.\n");
        }
    }
```

```
/* PUBLIC FUNCTION. This function is called when user search by event code.
 * Get event code from the user, print that information if it has.
 * and then ask the user to modify or delete information.
 */
void controlSearchCode()
    {
    char input[LENGTH] = {0};  /* Get input from the terminal. */
    int eventCode[2];          /* Store event code from user. */
    int position = 0;          /* Get the position of information. */
    int status = 0;            /* Keep value correction after validate information. */
    status = askEventCode(eventCode);
    if (status == 0) /* User hits <CR>. */
        return;
    status = searchEventCode(eventCode, &position);
    if (status == 1) /* User gets event code. */
        {
        printf("    What do you want to do? (1 = MODIFY, 2 = DELETE): ");
        while(1)
            {
            memset(input, 0, sizeof(input));
            fgets(input, sizeof(input), stdin);
            if (strlen(input) == 1) /* If user hits <CR>, go back to menu. */
                return;
            else if ((input[0] == '1') && (strlen(input) == 2))
                { /* User gets 1 to modify information. */
                modifyEvent(position);
                return;
                }
            else if ((input[0] == '2') && (strlen(input) == 2))
                { /* User gets 2 to delete information. */
                deleteEvent(position);
                return;
                }
            else  /* User inputs wrong. */
                printf("#SYSTEM: Input is invalid, try again: ");
            }
        }
    }
```

```
/* PUBLIC FUNCTION.
 * This function is called when user search by others such as
 * event year, event type and result.
 * Ask the user to get event year, type of event or result.
 * Then get event code from the user, print that information if it has.
 * and then ask the to modify or delete that information.
 */
void controlSearchOthers()
    {
    char type[SHORTLEN] = {0};    /* Get event type from user. */
    int eventYear = 0;            /* Get event year from user. */
    int result = 0;               /* Get result from user. */
    int countData = 0;            /* Count the number of information. */
    int checkAsk = 0;             /* Keep the number of asking. */
    int *pPosition = NULL;        /* Keep position of information. */
    printf("\n#SYSTEM: Please input to search information.\n");
    printf("#SYSTEM: Hit <CR>, if you don't want to search.\n");
    if (askEventYear(&eventYear) == 1)
       { /* Ask for event year, If user doesn't input, don't search in data. */
        pPosition = searchEventYear(eventYear, &countData);
        checkAsk++;
       }
    if (askType(type) == 1)
       { /* Ask for event type, If user doesn't input, don't search in data. */
        pPosition = searchEventType(type, pPosition, &countData);
       checkAsk++;
       }
    if (askResult(&result) == 1)
       { /* Ask for result, If user doesn't input, don't search in data. */
        pPosition = searchResult(result, pPosition, &countData);
       checkAsk++;
       }
    if (checkAsk == 0) /* If user doesn't input anything. */
       return;
    else if (pPosition == NULL) /* User doesn't get information. */
        {
        printf("#SYSTEM: The information is not found.\n");
        return;
        }
    else if (pPosition != NULL)  /* Users finish getting information. */
        {
        printf("\n");
        printEachEvent(pPosition, countData);
        free(pPosition);
        controlSearchCode();
        return;
        }
    }
```

```
/* PUBLIC FUNCTION. This function is called when user wants to dump file. */
void controlDump()
    {
    printf("\n======================== Dump file ========================\n\n");
    printf("#SYSTEM: Now program is writing text file.\n");
    controlDatabase(3);
    printf("#SYSTEM: Dump text file success!\n");
    printf("#SYSTEM: Text file name '%s'.\n\n", DUMPFILE);
    printf("===========================================================\n\n");
    return;
    }
```

```
/* PUBLIC FUNCTION. This function is called when user get a choice
 * that exit the program. Ask the user again to exit the program.
 *     If the user gets Y, the program will exit.
 *     If the user gets N, the program still works.
 */
void controlExit()
    {
    char input[LENGTH] = {0};  /* Get input from the terminal. */
    printf("Do you want to exit program? (Y/N): ");
    while(1)
        {
        memset(input, 0, sizeof(input));
        fgets(input, sizeof(input), stdin);
        sscanf(input, "%s", input);
        if (strlen(input) == 1)
            {
            if ((input[0] == 'N') || (input[0] == 'n'))
                { /* Users don't want to exit the program. */
                printf("\n");
                break;
                }
            else if ((input[0] == 'Y') || (input[0] == 'y'))
                { /* Users want to exit the program. */
                controlDatabase(4);
                printf("#SYSTEM: Program shuts down.\n\n");
                exit(0);
                }
            }
        printf("#SYSTEM: Input is invalid, try again (Y/N): ");
        }
    }
```