```
// Name: Rodrigo Ignacio Rojas Garcia
// Course Number: ECE 2230
// Section: 001
// Semester: Spring 2017
// Assignment Number: 2

// Library Declartion Section
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "guitar.h"
#include "guitardb.h"
#include "list.h"

// Allocate dynamic memory for guitar_s strucutre and returns it's address
guitar_t guitar_init(void)
{
    guitar_t item;
    item = (guitar_t)calloc(1,sizeof(struct guitar_s));
    return item;
};

// Function gutiar_fill will prompt the user to enter each field required for the i
tem and store it in it.
int guitar_fill(guitar_t item)
{
    char input[MAXCHARACTERS];
    char description[MAXCHARACTERS];
    char enum_temp;
    int c1;
    printf("Guitar Make (MAX 19 Characters): ");
    fgets(input, MAXCHARACTERS, stdin);
    for (c1 = 0; c1 < 19 && input[c1] != '\n'; c1++)
    {
        description[c1] = input[c1];
    }
    description[c1] = '\0';
    strcpy(item->make, description);
    memset(input, '\0', sizeof(input));
    printf("Guitar Model (MAX 19 Characters): ");
    fgets(input, MAXCHARACTERS, stdin);
    for (c1 = 0; c1 < 19 && input[c1] != '\n'; c1++)
    {
        description[c1] = input[c1];
    }
    description[c1] = '\0';
    strcpy(item->model, description);
    memset(input, '\0', sizeof(input));
    printf("Guitar Submodel (MAX 19 Characters): ");
    fgets(input, MAXCHARACTERS, stdin);
    for (c1 = 0; c1 < 19 && input[c1] != '\n'; c1++)
    {
        description[c1] = input[c1];
    }
    description[c1] = '\0';
    strcpy(item->submodel, description);
    memset(input, '\0', sizeof(input));
    printf("Guitar Type (S-Solid Body, A-Arch-Top Hallow Body, T-Semi-Hollow): ");
    fgets(input, MAXCHARACTERS, stdin);
    sscanf(input, "%c", &enum_temp);
    item->gtype = enum_temp;
    printf("Year: ");
    fgets(input, MAXCHARACTERS, stdin);
    sscanf(input, "%d", &item->year);
    printf("Guitar Finish (MAX 24 Characters): ");
    fgets(input, MAXCHARACTERS, stdin);
    for (c1 = 0; c1 < 19 && input[c1] != '\n'; c1++)
    {
        description[c1] = input[c1];
    }
    description[c1] = '\0';
    strcpy(item->finsh, description);
    memset(input, '\0', sizeof(input));
    printf("Number of Strings: ");
    fgets(input, MAXCHARACTERS, stdin);
    sscanf(input, "%d", &item->strings);
    printf("Number of Pickups: ");
    fgets(input, MAXCHARACTERS, stdin);
    sscanf(input, "%d", &item->pickups);
    printf("For the Pickup types, choose one of the following: \n");
    printf("H - Humbucker\nC - Single Coil\nP - P90\nF - Filterton\nN - None\n");
    printf("Neck Pickup Type: ");
    fgets(input, MAXCHARACTERS, stdin);
    sscanf(input, "%c", &enum_temp);
    item->neck = enum_temp;
    printf("Middle Pickup Type: ");
    fgets(input, MAXCHARACTERS, stdin);
    sscanf(input, "%c", &enum_temp);
    item->middle = enum_temp;
    printf("Bridge Pickup Type: ");
    fgets(input, MAXCHARACTERS, stdin);
    sscanf(input, "%c", &enum_temp);
    item->bridge = enum_temp;
    return 0;
};

// Function guitar_setid sets the id_number passed in the function to the item pass
ed in the function
int guitar_setid(guitar_t item, key_t id_number)
{
    item->id_number = id_number;
    return 0;
}

// Function guitar_getid returns the id_number of the item passed on the function
key_t guitar_getid(guitar_t item)
{
    return item->id_number;
}

// Function guitar_print will print all charactersitics of the item on the list
int guitar_print(guitar_t item)
{
    printf("\nGuitar ID: %d\n", item->id_number);
    printf("Make: %s\n", item->make);
    printf("Model: %s\n", item->model);
    printf("Submodel: %s\n", item->submodel);
    if (item->gtype == 'S')
    {
        printf("Type: Solid body\n");
    }
    else if (item->gtype == 'A')
    {
        printf("Type: Arch-Top Hollow Body\n");
    }
    else if (item->gtype == 'T')
    {
        printf("Type: Semi-Hollow (thinline)\n");
    }
    else
    {
        printf("Error\n");
    }
    printf("Year: %d\n", item->year);
```

```
    printf("Finish: %s\n", item->finsh);
    printf("Number of Strings: %d\n", item->strings);
    printf("Number of Pickups: %d\n", item->pickups);
    if (item->neck == 'H')
    {
        printf("Neck Pickup Type: Humbucker\n");
    }
    else if (item->neck == 'C')
    {
        printf("Neck Pickup Type: Single Coil\n");
    }
    else if (item->neck == 'P')
    {
        printf("Neck Pickup Type: P90\n");
    }
    else if (item->neck == 'F')
    {
        printf("Neck Pickup Type: Filtertron\n");
    }
    else if (item->neck == 'N')
    {
        printf("Neck Pickup Type: None\n");
    }
    else
    {
        printf("Error\n");
    }
    if (item->middle == 'H')
    {
        printf("Middle Pickup Type: Humbucker\n");
    }
    else if (item->middle == 'C')
    {
        printf("Middle Pickup Type: Single Coil\n");
    }
    else if (item->middle == 'P')
    {
        printf("Middle Pickup Type: P90\n");
    }
    else if (item->middle == 'F')
    {
        printf("Middle Pickup Type: Filtertron\n");
    }
    else if (item->middle == 'N')
    {
        printf("Middle Pickup Type: None\n");
    }
    else
    {
        printf("Error\n");
    }
    if (item->bridge == 'H')
    {
        printf("Bridge Pickup Type: Humbucker\n");
    }
    else if (item->bridge == 'C')
    {
        printf("Bridge Pickup Type: Single Coil\n");
    }
    else if (item->bridge == 'P')
    {
        printf("Bridge Pickup Type: P90\n");
    }
    else if (item->bridge == 'F')
    {
        printf("Bridge Pickup Type: Filtertron\n");
    }
    else if (item->bridge == 'N')
    {
        printf("Bridge Pickup Type: None\n");
    }
    else
    {
        printf("Error\n");
    }
    printf("\n");

    return 0;
}

// Function guitar_compare will compare two items passed in the function and will c
all function guitar_getid to compare the id numbers of the two items. If
// they have the same id number it will return 0, if the item that wants to be adde
d to the list is greater then it will return a 1, if it is less than it will return
 a -1
int guitar_compare(guitar_t item1, guitar_t item2)
{
    if (guitar_getid(item1) == guitar_getid(item2))
    {
        return 0;
    }
    else if (guitar_getid(item1) > guitar_getid(item2))
    {
        return 1;
    }
    else if (guitar_getid(item1) < guitar_getid(item2))
    {
        return -1;
    }
    return 0;
}


// Function guitar_free frees allocated memory used to store an item
void guitar_free(guitar_t item)
{
    free(item);
}
```