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

#define MAX_EVENTS 100
#define MAX_DESC 100

// ANSI colors
#define COLOR_RESET "\033[0m"
#define COLOR_EVENT "\033[1;32m"
#define COLOR_TODAY "\033[1;34m"

typedef struct {
    int day, month, year;
    char desc[MAX_DESC];
} Event;

Event events[MAX_EVENTS];
int eventCount = 0;

// Utility functions
int isLeapYear(int year) {
    return ((year % 4 == 0 && year % 100 != 0) || year % 400 == 0);
}

int getDaysInMonth(int month, int year) {
    int daysInMonth[] = { 31, 28, 31, 30, 31, 30,
                          31, 31, 30, 31, 30, 31 };
    if (month == 2 && isLeapYear(year))
        return 29;
    return daysInMonth[month - 1];
}

int getStartDay(int month, int year) {
    if (month < 3) {
        month += 12;
        year -= 1;
    }
    int k = year % 100;
    int j = year / 100;
    return (1 + (13 * (month + 1)) / 5 + k + k / 4 + j / 4 + 5 * j) % 7;
}
```

```
int hasEvent(int day, int month, int year, char *desc) {
    for (int i = 0; i < eventCount; i++) {
        if (events[i].day == day && events[i].month == month && events[i].year == year) {
            if (desc) strcpy(desc, events[i].desc);
            return 1;
        }
    }
    return 0;
}

void addEvent() {
    if (eventCount >= MAX_EVENTS) {
        printf("Max event limit reached!\n");
        return;
    }
    Event e;
    printf("Enter event date (day month year): ");
    scanf("%d %d %d", &e.day, &e.month, &e.year);
    getchar(); // clear buffer
    printf("Enter event description: ");
    fgets(e.desc, MAX_DESC, stdin);
    e.desc[strcspn(e.desc, "\n")] = '\0';
    events[eventCount++] = e;
    printf("Event added!\n");
}

void searchEvent() {
    int d, m, y;
    printf("Enter date to search (day month year): ");
    scanf("%d %d %d", &d, &m, &y);
    int found = 0;
    for (int i = 0; i < eventCount; i++) {
        if (events[i].day == d && events[i].month == m && events[i].year == y) {
            printf("Event: %s\n", events[i].desc);
            found = 1;
        }
    }
    if (!found) printf("No event on that date.\n");
}

void printCalendar(int month, int year) {
    char *months[] = { "", "January", "February", "March", "April", "May", "June",
                       "July", "August", "September", "October", "November", "December" };
```

```
    time_t t = time(NULL);
    struct tm *today = localtime(&t);

    printf("\n     %s %d\n", months[month], year);
    printf("Su Mo Tu We Th Fr Sa\n");

    int startDay = getStartDay(month, year);
    int days = getDaysInMonth(month, year);

    for (int i = 0; i < startDay; i++)
        printf("   ");

    for (int day = 1; day <= days; day++) {
        char desc[MAX_DESC];
        int isEvent = hasEvent(day, month, year, desc);
        int isToday = (day == today->tm_mday && month == today->tm_mon + 1 && year ==
today->tm_year + 1900);

        if (isEvent)
            printf(COLOR_EVENT "[%2d]" COLOR_RESET, day);
        else if (isToday)
            printf(COLOR_TODAY " %2d " COLOR_RESET, day);
        else
            printf(" %2d ", day);

        if ((startDay + day) % 7 == 0)
            printf("\n");
    }
    printf("\n");
}

int main() {
    int month, year, choice;

    time_t t = time(NULL);
    struct tm *now = localtime(&t);
    month = now->tm_mon + 1;
    year = now->tm_year + 1900;

    while (1) {
        printCalendar(month, year);
        printf("\nMenu:\n");
        printf("1. Previous Month\n");
        printf("2. Next Month\n");
```

```
        printf("3. Add Event\n");
        printf("4. Search Event by Date\n");
        printf("5. Exit\n");
        printf("Choose an option: ");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                month--;
                if (month < 1) {
                    month = 12;
                    year--;
                }
                break;
            case 2:
                month++;
                if (month > 12) {
                    month = 1;
                    year++;
                }
                break;
            case 3:
                addEvent();
                break;
            case 4:
                searchEvent();
                break;
            case 5:
                exit(0);
            default:
                printf("Invalid choice. Try again.\n");
        }
    }
    return 0;
}
```