



## Green University of Bangladesh

Department of EEE (Day) || Spring, 2023

### Calendar Manager

**Course Title:** Computer Programming Lab ||

**Code:** CSE 102 || **Section:** 232 D2

Students Details

Name	Id
Naimul Hassan Sourov	232001055

**Submission Date:** 14 June 2024

**Course Teacher's Name:** MD SAKLAIN MORSHED

Lab Project Status
Marks: Signature: Comments: Date:

# Table of Contents

1. Introduction .....	3
1.1 Title of the Project.....	3
1.2 Introduction .....	3
1.3 Why Choose Calendar Manager.....	3
1.4 Problem Statement.....	3
1.5 Objectives.....	3
2. Implementation of the Project .....	4
2.1 Tools & Technologies .....	4
2.3 Concepts Used .....	4
2.4 Learning Topics .....	4
2.4.1 Conditional Logic .....	4
2.4.2 Loops .....	4
2.4.3 Algorithmic Logic.....	4
2.4.4 Arrays .....	4
2.4.5 Modular Programming.....	4
2.5 Code Implementation .....	4
2.5.1 Main Function and Menu Display .....	5
2.5.2 Determining Leap Years: .....	8
2.5.3 Days in Month Calculation .....	8
2.5.4 Day of the Week Calculation.....	8
2.5.5 Printing Calendar for Month and Year .....	8
2.5.6 Date Operations .....	9
3. Performance Evaluation .....	11
3.1 Test Result / Output.....	11
3.2 Achievements .....	14
3.3 Challenges Faced.....	14
4. Future Work .....	15
4.1 Additional Features .....	15

5. Conclusion .....	15
Discussion .....	15
References .....	15

## 1. Introduction

### 1.1 Title of the Project

Calendar Manager

### 1.2 Introduction

This Calendar Manager project offers functionalities such as printing a calendar for any year, determining the day of the week for a specific date, calculating the day number within a year, computing the difference between two dates, and adding days to a specific date. With an easy-to-use menu-driven interface, Calendar Manager makes handling dates straightforward and efficient.

### 1.3 Why Choose Calendar Manager

- Comprehensive date functionalities
- Efficient and accurate calculations
- User-friendly, menu-driven interface
- Modular, expandable codebase

### 1.4 Problem Statement

The project is designed to address the need for managing and interacting with dates effectively, providing functionalities like viewing calendars, calculating date differences, and modifying dates.

### 1.5 Objectives

- Offer a range of functions for viewing, calculating, and managing dates.
- Ensure precise and fast date calculations using reliable algorithms.
- Provide an easy-to-use interface suitable for all user levels.
- Serve as a resource for learning date/time management and programming concepts.
- Design the code for easy updates and extensions by future developers.

## 2. Implementation of the Project

### 2.1 Tools & Technologies

- Hp laptop
- Code: Blocks

### 2.3 Concepts Used

- Conditional Logic
- Loops
- Algorithmic Logic
- Arrays
- Modular Programming

### 2.4 Learning Topics

#### *2.4.1 Conditional Logic*

Used to determine whether a year is a leap year and how many days are in each month.

#### *2.4.2 Loops*

For iterating over days, months, and years, and for controlling the flow of the application based on user input.

#### *2.4.3 Algorithmic Logic*

Implementing algorithms like Sakamoto's method for calculating the day of the week demonstrates proficiency in logical problem-solving.

#### *2.4.4 Arrays*

Employed to store and manage names of months and days of the week, facilitating easy access and manipulation.

#### *2.4.5 Modular Programming*

The project is divided into functions, each handling specific tasks such as printing a calendar, determining leap years, or handling date differences, enhancing code readability and maintainability.

### 2.5 Code Implementation

The Calendar Manager project uses C programming language to implement several features that revolve around calendar and date management. Below are key aspects of the code implementation:

### *2.5.1 Main Function and Menu Display*

The main function drives the menu-based interface, allowing users to select different operations. The function prints the available options to the user.

```

1  #include <stdio.h>
2  #include <stdlib.h>
3
4  // Function prototypes
5  int isLeapYear(int year);
6  int getDaysInMonth(int month, int year);
7  int getDayOfWeek(int day, int month, int year);
8  void printMonth(int month, int year);
9  void printYear(int year);
10 void printDayOfWeek(int day, int month, int year);
11 int dayOfYear(int day, int month, int year);
12 void dateDifference(int day1, int month1, int year1, int day2, int month2, int year2);
13 void addDays(int day, int month, int year, int daysToAdd);
14 void displayMenu();
15
16 int main() {
17     int choice;
18     do {
19         displayMenu();
20         scanf("%d", &choice);
21
22         switch (choice) {
23             case 1: {
24                 int year;
25                 printf("Enter year: ");
26                 scanf("%d", &year);
27                 printYear(year);
28                 break;
29             }
30             case 2: {
31                 int day, month, year;
32                 printf("Enter day: ");
33                 scanf("%d", &day);
34                 printf("Enter month: ");
35                 scanf("%d", &month);
36                 printf("Enter year: ");
37                 scanf("%d", &year);
38                 printDayOfWeek(day, month, year);
39                 break;
40             }
41             case 3: {
42                 int day, month, year;
43                 printf("Enter day: ");
44                 scanf("%d", &day);
45                 printf("Enter month: ");
46                 scanf("%d", &month);

```

```

47     printf("Enter year: ");
48     scanf("%d", &year);
49     printf("Day of year: %d\n", dayOfYear(day, month, year));
50     break;
51 }
52 case 4: {
53     int day1, month1, year1, day2, month2, year2;
54     printf("Enter first date (day month year): ");
55     scanf("%d %d %d", &day1, &month1, &year1);
56     printf("Enter second date (day month year): ");
57     scanf("%d %d %d", &day2, &month2, &year2);
58     dateDifference(day1, month1, year1, day2, month2, year2);
59     break;
60 }
61 case 5: {
62     int day, month, year, daysToAdd;
63     printf("Enter day: ");
64     scanf("%d", &day);
65     printf("Enter month: ");
66     scanf("%d", &month);
67     printf("Enter year: ");
68     scanf("%d", &year);
69     printf("Enter number of days to add: ");
70     scanf("%d", &daysToAdd);
71     addDays(day, month, year, daysToAdd);
72     break;
73 }
74 case 0:
75     printf("Exiting program...\n");
76     break;
77 default:
78     printf("Invalid choice. Please try again.\n");
79 }
80 } while (choice != 0);
81
82 return 0;
83 }
84
85 void displayMenu() {
86     printf("\nMenu:\n");
87     printf("1. Print calendar for a year\n");
88     printf("2. Find day of the week for a specific date\n");
89     printf("3. Find day number within the year\n");
90     printf("4. Calculate difference between two dates\n");
91     printf("5. Add days to a date\n");
92     printf("0. Exit\n");
93     printf("Enter your choice: ");
94 }

```

### 2.5.2 Determining Leap Years:

The function determines if a year is a leap year. It uses logical conditions to check if the year is divisible by 4, not divisible by 100 unless it's also divisible by 400.

```
96 // Function to check if a year is a leap year
97 int isLeapYear(int year) {
98     return (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0);
99 }
```

### 2.5.3 Days in Month Calculation

The function returns the number of days in each month, considering leap years.

```
101 // Function to get the number of days in a month
102 int getDaysInMonth(int month, int year) {
103     switch(month) {
104         case 1: return 31;
105         case 2: return isLeapYear(year) ? 29 : 28;
106         case 3: return 31;
107         case 4: return 30;
108         case 5: return 31;
109         case 6: return 30;
110         case 7: return 31;
111         case 8: return 31;
112         case 9: return 30;
113         case 10: return 31;
114         case 11: return 30;
115         case 12: return 31;
116         default: return 0;
117     }
118 }
```

### 2.5.4 Day of the Week Calculation

The function uses Sakamoto's method to calculate the day of the week for a given date.

```
120 // Function to get the day of the week for a given date using Sakamoto's method
121 int getDayOfWeek(int day, int month, int year) {
122     static int t[] = { 0, 3, 2, 5, 0, 3, 5, 1, 4, 6, 2, 4 };
123     if (month < 3) {
124         year -= 1;
125     }
126     return (year + year/4 - year/100 + year/400 + t[month-1] + day) % 7;
127 }
```

### 2.5.5 Printing Calendar for Month and Year

- The printMonth function prints the calendar for a specific month and year.



- The printYear function prints the calendar for all months each year by calling printMonth.

```

129 // Function to print the calendar for a given month and year
130 void printMonth(int month, int year) {
131     char *months[] = { "January", "February", "March", "April", "May", "June",
132                       "July", "August", "September", "October", "November", "December" };
133     int days = getDaysInMonth(month, year);
134     int startDay = getDayOfWeek(1, month, year);
135
136     printf("\n -----%s-----\n", months[month - 1]);
137     printf(" Sun Mon Tue Wed Thu Fri Sat\n");
138
139     // Print leading spaces for the first day of the month
140     for (int i = 0; i < startDay; i++) {
141         printf("   ");
142     }
143
144     // Print the days of the month
145     for (int day = 1; day <= days; day++) {
146         printf("%5d", day);
147         if ((startDay + day) % 7 == 0) {
148             printf("\n");
149         }
150     }
151     printf("\n");
152 }

```

#### 2.5.6 Date Operations

Function to print the calendar for a given year

```

154 // Function to print the calendar for a given year
155 void printYear(int year) {
156     for (int month = 1; month <= 12; month++) {
157         printMonth(month, year);
158     }
159 }

```

Function to print the name of the day of the week for a given date

```

161 // Function to print the name of the day of the week for a given date
162 void printDayOfWeek(int day, int month, int year) {
163     char *daysOfWeek[] = { "Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday" };
164     int dayOfWeek = getDayOfWeek(day, month, year);
165     printf("The day of the week for %d-%d-%d is: %s\n", day, month, year, daysOfWeek[dayOfWeek]);
166 }

```

Function to get the day number within the year for a given date

```

168 // Function to get the day number within the year for a given date
169 int dayOfYear(int day, int month, int year) {
170     int days = day;
171     for (int i = 1; i < month; i++) {
172         days += getDaysInMonth(i, year);
173     }
174     return days;
175 }

```

Function to get the day number within the year for a given date

```

177 // Function to calculate the difference between two dates
178 void dateDifference(int day1, int month1, int year1, int day2, int month2, int year2) {
179     int n1 = year1 * 365 + day1;
180     for (int i = 0; i < month1 - 1; i++) {
181         n1 += getDaysInMonth(i + 1, year1);
182     }
183     n1 += year1 / 4 - year1 / 100 + year1 / 400;
184
185     int n2 = year2 * 365 + day2;
186     for (int i = 0; i < month2 - 1; i++) {
187         n2 += getDaysInMonth(i + 1, year2);
188     }
189     n2 += year2 / 4 - year2 / 100 + year2 / 400;
190
191     printf("Difference: %d days\n", abs(n2 - n1));
192 }

```

Function to add days to a date and print the new date

```

194 // Function to add days to a date and print the new date
195 void addDays(int day, int month, int year, int daysToAdd) {
196     while (daysToAdd > 0) {
197         int daysInMonth = getDaysInMonth(month, year);
198         if (day + daysToAdd <= daysInMonth) {
199             day += daysToAdd;
200             daysToAdd = 0;
201         } else {
202             daysToAdd -= (daysInMonth - day + 1);
203             day = 1;
204             if (++month > 12) {
205                 month = 1;
206                 year++;
207             }
208         }
209     }
210     printf("New date: %d-%d-%d\n", day, month, year);
211 }

```

### 3. Performance Evaluation

#### 3.1 Test Result / Output

Here's an example of what the interaction might look like:

Menu:

1. Print calendar for a year
2. Find day of the week for a specific date
3. Find day number within the year
4. Calculate difference between two dates
5. Add days to a date
0. Exit

Enter your choice: 1

Enter year: 2024

-----January-----

Sun	Mon	Tue	Wed	Thu	Fri	Sat
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

-----February-----

Sun	Mon	Tue	Wed	Thu	Fri	Sat
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29		

-----March-----

Sun	Mon	Tue	Wed	Thu	Fri	Sat
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

-----April-----

Sun	Mon	Tue	Wed	Thu	Fri	Sat
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

-----May-----

Sun	Mon	Tue	Wed	Thu	Fri	Sat
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

Menu:

1. Print calendar for a year
2. Find day of the week for a specific date
3. Find day number within the year
4. Calculate difference between two dates
5. Add days to a date
0. Exit

Enter your choice: 2

Enter day: 8

Enter month: 1

Enter year: 2024

The day of the week for 8-1-2024 is: Monday

Menu:

1. Print calendar for a year
2. Find day of the week for a specific date
3. Find day number within the year
4. Calculate difference between two dates
5. Add days to a date
0. Exit

Enter your choice: 3

Enter day: 1

Enter month: 8

Enter year: 2024

Day of year: 214

Menu:

1. Print calendar for a year
2. Find day of the week for a specific date
3. Find day number within the year
4. Calculate difference between two dates
5. Add days to a date
0. Exit

Enter your choice: 4

Enter first date (day month year): 23  
8

2003

Enter second date (day month year): 11  
6

2024

Difference: 7599 days

```
Menu:
1. Print calendar for a year
2. Find day of the week for a specific date
3. Find day number within the year
4. Calculate difference between two dates
5. Add days to a date
0. Exit
Enter your choice: 5
Enter day: 1
Enter month: 1
Enter year: 2024
Enter number of days to add: 5
New date: 6-1-2024
```

```
Menu:
1. Print calendar for a year
2. Find day of the week for a specific date
3. Find day number within the year
4. Calculate difference between two dates
5. Add days to a date
0. Exit
Enter your choice: 0
Exiting program...

Process returned 0 (0x0)   execution time : 696.220 s
Press any key to continue.
```

### 3.2 Achievements

It features an efficient, user-friendly, menu-driven interface, employing robust algorithms like Sakamoto's method for quick and precise results. The modular design and basic error handling enhance readability, maintainability, and robustness, making it an excellent educational tool for learning C programming concepts.

### 3.3 Challenges Faced

Several challenges were encountered during development, particularly in implementing efficient algorithms like Sakamoto's method and maintaining a modular code structure for seamless functionality posed significant challenges.

## 4. Future Work

### 4.1 Additional Features

Additional features could include displaying holidays for the year and providing a graphical interface for easier interaction. Another enhancement could be adding support for different calendar systems, such as lunar or Julian calendars.

## 5. Conclusion

The Calendar Manager project successfully implements a range of date-related functionalities with accurate calculations and a user-friendly interface. It serves as an effective educational tool for learning C programming concepts and can be further enhanced with additional features and improvements.

## Discussion

It features a menu-driven interface for user interaction and employs efficient algorithms for accurate date computations. The project aims to offer a practical tool for date-related tasks while serving as an educational resource for learning C programming concepts.

## References

### 1. Textbook:

- Teach Yourself C – By Herbert Schildt; Publisher: McGraw-Hill Osborne Media; 3rd Edition (April 1, 1997)

### 2. Online Tutorials or Documentation:

- [how to find the week-day for any date?](#)

### 3. Lecture Notes or Slides:

- Md Saklain Morshed, Lecture Slides on Structure Programming Lab, Green University of Bangladesh.