MICRO PROJECT REPORT

Title: Calendar Application Using C Language

---

1. Introduction

The calendar is an essential tool for organizing daily, monthly, and yearly schedules. This project presents a calendar program implemented in C, which can display the calendar for any given month and year. The program incorporates functionality to navigate between months, view specific dates, and calculate leap years, making it an efficient and interactive tool for users.

---

2. Problem Statement

The primary aim of the project is to create a user-friendly calendar application that can:

1. Display the calendar of any given month and year.

2. Calculate and display leap years.

3. Allow navigation between months and years.

4. Provide real-time date integration.

---

3. Algorithm

The program uses the following steps:

1. Input and Initialization: Initialize the current month and year using system time.

2. Leap Year Calculation: Implement a function to determine if the given year is a leap year.

3. Days in a Month: Create a function to calculate the number of days in a given month.

4. First Day of the Month: Use a formula to determine the weekday of the 1st day of a month.

5. Display Calendar: Print the calendar layout for the selected month and year.

6. User Navigation: Provide options to navigate to the previous month, next month, or enter a custom date.

7. Exit Condition: Allow users to exit the program gracefully.

---

4. Flowchart

```
Start
          |
   Get current date
          |
   Display calendar
          |
     User options:
  1. Previous Month
  2. Next Month
  3. Custom Month/Year
  4. Exit
          |
 Update month/year
          |
       Repeat
          |
         End
```

---

5. Methodology (Used Concepts)

1. Control Structures: Loops and conditional statements for program flow.

2. Functions: Modular approach using functions for leap year calculation, day calculation, and calendar display.

3. Date-Time Library: Utilized time.h to fetch the current date.

4. Data Structures: Arrays to store the number of days in each month.

5. Mathematical Formulas: Applied Zeller's Congruence for determining the first day of the month.

---

6. Results (Output)

The program successfully displays a well-formatted calendar for any selected month and year.

It accurately accounts for leap years and adjusts February's days accordingly.

Users can navigate seamlessly between months or jump directly to a specific month and year.

Sample Output:

```
December 2024
 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
```

---

7. Future Scope

1. Event Scheduling: Integrating event reminders for specific dates.

2. Graphical Interface: Transitioning from console-based to GUI-based calendars.

3. Multilingual Support: Adding support for different languages and regional formats.

4. Integration: Syncing with online calendars like Google Calendar.

5. Year View: Displaying the entire year in a single view.

---

8. Conclusion

The calendar project successfully demonstrates the capability of C programming to solve real-world problems. It showcases the use of modular programming, logical problem-solving, and effective use of data structures. The application is interactive, efficient, and has the potential for future enhancements.

---

9. References

1. Books:

"The C Programming Language" by Brian Kernighan and Dennis Ritchie.

2. Websites:

GeeksforGeeks: www.geeksforgeeks.org

TutorialsPoint: www.tutorialspoint.com

3. Tools:

GCC Compiler

Code::Blocks IDE

---

10. Implemented Code

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

// Function to check if a year is a leap year
int isLeapYear(int year) {
    if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {
        return 1;
    }
    return 0;
}

// Function to get the number of days in a month
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];
}

// Function to calculate the day of the week for the 1st day of a month
int getFirstDayOfMonth(int month, int year) {
    int day = 1;
    int y = year - (14 - month) / 12;
    int m = month + 12 * ((14 - month) / 12) - 2;
    return (day + y + y / 4 - y / 100 + y / 400 + (31 * m) / 12) % 7;
}

// Function to display the calendar for a month
void displayCalendar(int month, int year) {
    char *months[] = {
        "January", "February", "March", "April", "May", "June",
        "July", "August", "September", "October", "November", "December"
    };
    int days = getDaysInMonth(month, year);
    int startDay = getFirstDayOfMonth(month, year);

    // Declare i and day outside of the loop
    int i, day;

    printf("\n\n  %s %d\n", months[month - 1], year);
    printf("  Sun  Mon  Tue  Wed  Thu  Fri  Sat\n");

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

    for (day = 1; day <= days; day++) {
        printf("%5d", day);
        if ((day + startDay) % 7 == 0) {
            printf("\n");
        }
    }
    printf("\n");
}

// Main function
int main() {
    int month, year, choice;
    time_t t = time(NULL);
    struct tm *currentTime = localtime(&t);

    // Default to the current month and year
    month = currentTime->tm_mon + 1;
    year = currentTime->tm_year + 1900;

    while (1) {
```

```
printf("\n================ CALENDAR ================\n");
displayCalendar(month, year);
printf("\nOptions:\n");
printf("1. Previous Month\n");
printf("2. Next Month\n");
printf("3. Enter Month and Year\n");
printf("4. Exit\n");
printf("Enter your choice: ");
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:
        printf("Enter month (1-12): ");
        scanf("%d", &month);
        printf("Enter year: ");
        scanf("%d", &year);
        if (month < 1 || month > 12) {
            printf("Invalid month! Please enter a value between 1 and 12.\n");
            month = currentTime->tm_mon + 1;
            year = currentTime->tm_year + 1900;
        }
        break;

    case 4:
        printf("Exiting... Goodbye!\n");
        exit(0);

    default:
            printf("Invalid choice! Please try again.\n");
        }
    }

    return 0;
}
```

```
================ CALENDAR ================


  December 2024
  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

Options:
1. Previous Month
2. Next Month
3. Enter Month and Year
4. Exit
Enter your choice:
1

================ CALENDAR ================


  November 2024
  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


Options:
1. Previous Month
2. Next Month
3. Enter Month and Year
4. Exit
Enter your choice: 2
```

```
================ CALENDAR ================


  December 2024
  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

Options:
1. Previous Month
2. Next Month
3. Enter Month and Year:3
4. Exit
Enter your choice: 3
Enter month (1-12): 6
Enter year: 2035

================ CALENDAR ================


  June 2035
  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


Options:
1. Previous Month
2. Next Month
3. Enter Month and Year
4. Exit
Enter your choice:
```