Yearly Calendar Generator

Project Overview

This C program generates and displays a calendar for any given year. It calculates the number of days in each month, determines whether the year is a leap year, and computes the day of the week for the first day of each month. The program provides a neatly formatted output, with the calendar for each month presented in a grid format showing the days of the week.

Features

- Input any year to generate a full calendar.
- Automatically accounts for leap years.
- Displays each month in a grid with proper alignment.
- Utilizes Zeller's Congruence algorithm to compute the first day of each month.

Functions

`int isLeapYear(int year)`

This function checks if the provided year is a leap year.

- **Input**: `int year` - The year to check.
- **Output**: Returns `1` if the year is a leap year, otherwise returns `0`.

Logic:

- A year is a leap year if it is divisible by 4, but not divisible by 100, unless it is divisible by 400.

`int getDaysInMonth(int month, int year)`

This function returns the number of days in a given month, considering leap years.

- **Input**:
  - `int month` - The month (1 for January, 2 for February, etc.).
  - `int year` - The year, used to account for leap years in February.
- **Output**: The number of days in the specified month.

Month-Day Mapping:

- January: 31 days
- February: 28 days (29 if leap year)
- March: 31 days
- April: 30 days
- May: 31 days
- June: 30 days
- July: 31 days
- August: 31 days
- September: 30 days
- October: 31 days
- November: 30 days
- December: 31 days

```
int getFirstDayOfMonth(int month, int year)
```

This function calculates the day of the week for the first day of the specified month and year using Zeller's Congruence algorithm.

- **Input**:
    - `int month` - The month.
    - `int year` - The year.
- **Output**: The day of the week for the first day of the month (1 for Monday, 7 for Sunday).

Zeller's Congruence Algorithm:

- If the month is January or February, adjust the month by adding 12 and subtract 1 from the year.
- The algorithm calculates the day of the week using modular arithmetic and other parameters derived from the date.

```
void displayMonthCalendar(int month, int year)
```

This function displays a single month in a calendar format with the day of the week aligned correctly.

- **Input**:
    - `int month` - The month to display.
    - `int year` - The year of the month.
- **Output**: Prints the calendar for the given month and year.

```
void displayYearCalendar(int year)
```

This function generates the full calendar for the entire year by calling `displayMonthCalendar()` for all 12 months.

- **Input**:
    - `int year` - The year to display.
- **Output**: Displays the calendar for all months of the specified year.

```
int main()
```

The main function prompts the user to input a year and then displays the entire calendar for that year.

- **Input**: The user provides the year.
- **Output**: The full calendar of the specified year is printed.

Example Output

```sql
Copy code
Enter the 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
 . . .
```

Algorithm Details

Leap Year Calculation

The program checks whether a year is a leap year using the following rules:

- If the year is divisible by 4, it may be a leap year.
- If the year is also divisible by 100, it is not a leap year unless it is also divisible by 400.

Day of the Week Calculation (Zeller's Congruence)

This algorithm is used to find the day of the week for the first day of a month. The program adjusts for January and February by treating them as months 13 and 14 of the previous year.

Example Use Cases

- Use this program to easily generate a calendar for any year, accounting for leap years.
- The program can be extended to add more features such as highlighting holidays or adding custom events.

Code Customization

- The program can be extended to include features such as marking public holidays or special events.
- You could allow for multi-language support by translating the month names.

Conclusion

This project provides a robust method for generating calendars for any year, accounting for leap years and correctly calculating the day of the week using Zeller's Congruence. It demonstrates the integration of multiple concepts, including modular arithmetic, leap year logic, and formatted output handling in C.