**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Chapter No.** | **Title** | **Page No.** |
| 1 | Problem Statement | 3 |
| 2 | Methodology / Procedure | 4 |
| 3 | Coding (C or Python) | 5 - 11 |
| 4 | Results | 12 -14 |
| 5 | Conclusion | 14 |

# Problem Statement

The given code is made to get a print of a calender.

The user will enter any year according to his choice and the after the code is run it will print out a calender in a particular format.

# PROCEDURE

To begin, we'll create two arrays: one with the number of days in each month, and another with all of the month names. Note that the first position in both arrays is intentionally left empty; we want to keep things simple by using 1 to 12.

The user input is obtained via the first function inputyear(). The user is asked to enter a year. Note that no input validation or error handling is done in order to keep things basic.

The following method, determinedaycode(), is used to get the day number of the first day of that year, allowing us to display the date in the proper location. (As a result, it's just utilized for output.)

The determineleapyear() method is used to see whether the user's input is a leap year. If this is the case, the number of days in February is increased to 29.

Each month is printed on the screen using the final function calendar(). To loop across all months, use the first for loop. The month's name and all of the days of the week are then printed. The daycode is then used to

place the prompt under the correct weekday. Then we print a month's worth of dates. The final step is to place the prompt in the proper weekday position.

# CODING

#include<stdio.h> #define TRUE 1

#define FALSE 0

int days\_in\_month[]={0,31,28,31,30,31,30,31,31,30,31,30,31};

char \*months[]=

{

" ",

"\n\n\nJanuary", "\n\n\nFebruary", "\n\n\nMarch", "\n\n\nApril", "\n\n\nMay", "\n\n\nJune", "\n\n\nJuly",

"\n\n\nAugust", "\n\n\nSeptember", "\n\n\nOctober", "\n\n\nNovember", "\n\n\nDecember"

};

int inputyear(void)

{

int year;

printf("Please enter a year (example: 1999) : "); scanf("%d", &year);

return year;

}

int determinedaycode(int year)

{

int daycode; int d1, d2, d3;

d1 = (year - 1.)/ 4.0;

d2 = (year - 1.)/ 100.;

d3 = (year - 1.)/ 400.;

daycode = (year + d1 - d2 + d3) %7; return daycode;

}

int determineleapyear(int year)

{

if(year% 4 == FALSE && year%100 != FALSE || year%400 == FALSE)

{

}

else

{

days\_in\_month[2] = 29; return TRUE;

days\_in\_month[2] = 28; return FALSE;

}

}

void calendar(int year, int daycode)

{

int month, day;

for ( month = 1; month <= 12; month++ )

{

Sat\n" );

printf("%s", months[month]); printf("\n\nSun Mon Tue Wed Thu Fri

// Correct the position for the first date

for ( day = 1; day <= 1 + daycode \* 5; day++ )

{

printf(" ");

}

// Print all the dates for one month

day++ )

for ( day = 1; day <= days\_in\_month[month];

{

printf("%2d", day );

// Is day before Sat? Else start next line Sun. if ( ( day + daycode ) % 7 > 0 )

printf(" " );

else

}

printf("\n " );

// Set position for next month

daycode = ( daycode + days\_in\_month[month] ) % 7;

}

}

int main(void)

{

int year, daycode, leapyear;

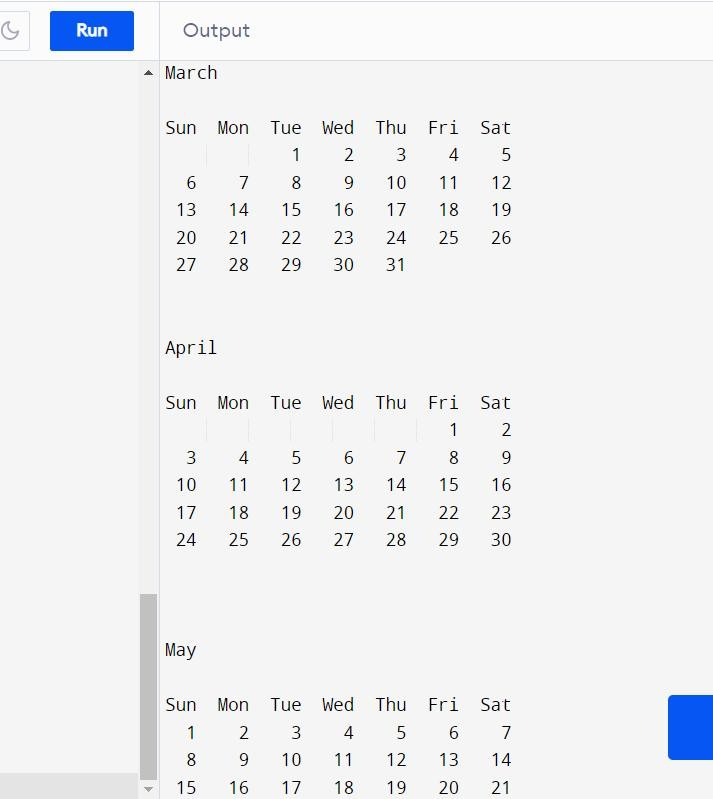
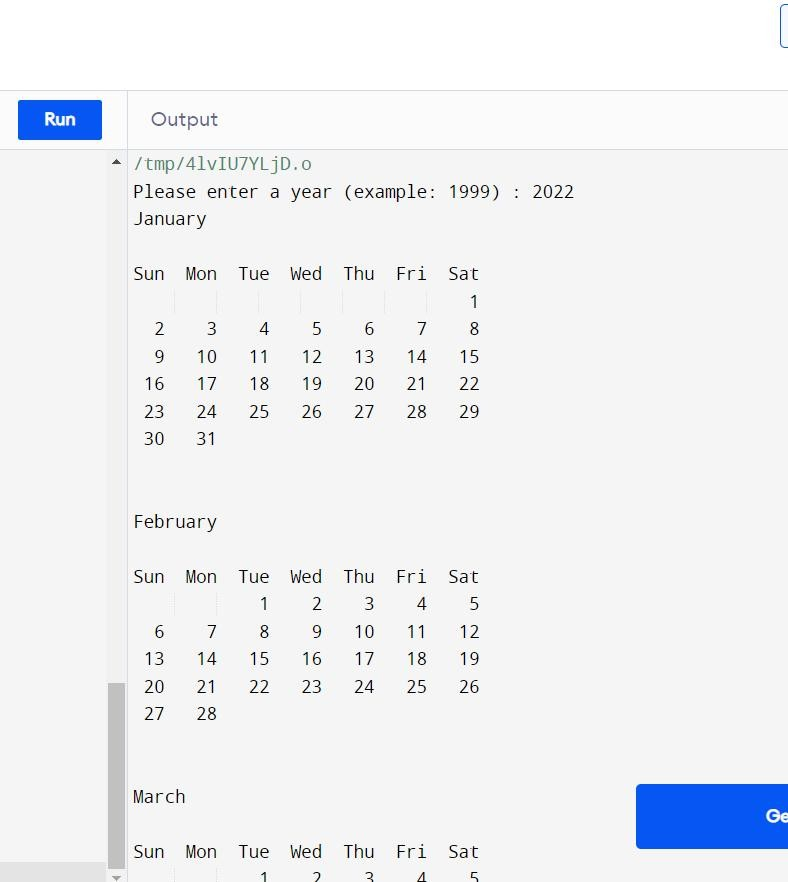
year = inputyear();

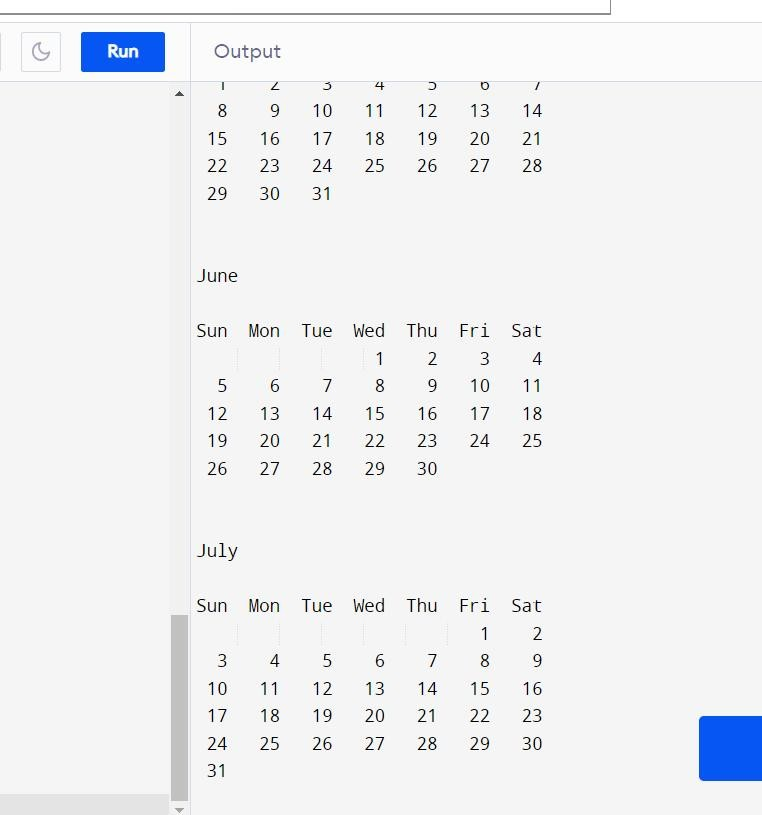
daycode = determinedaycode(year); determineleapyear(year); calendar(year, daycode);

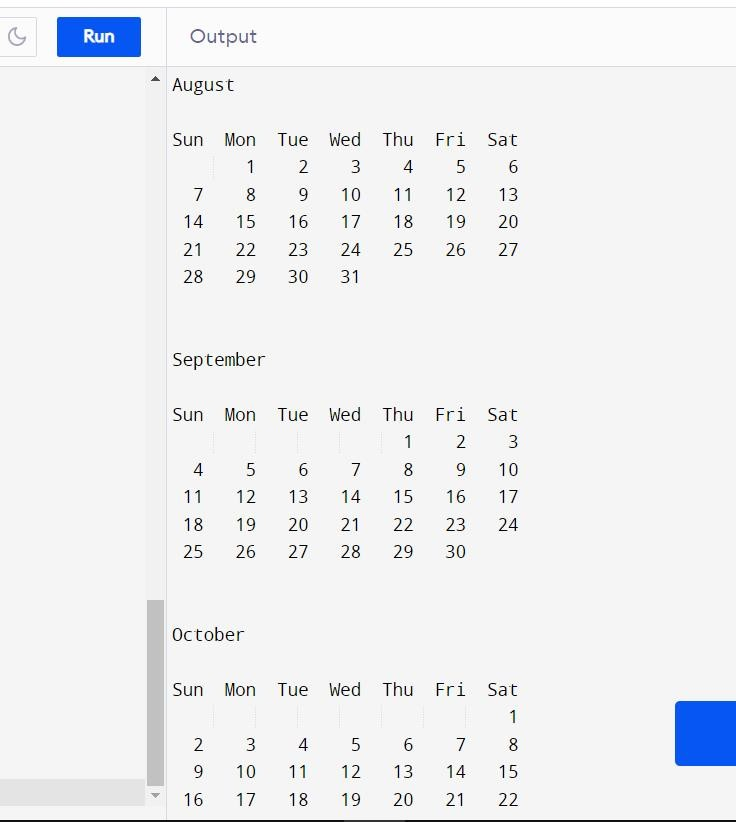
printf("\n");

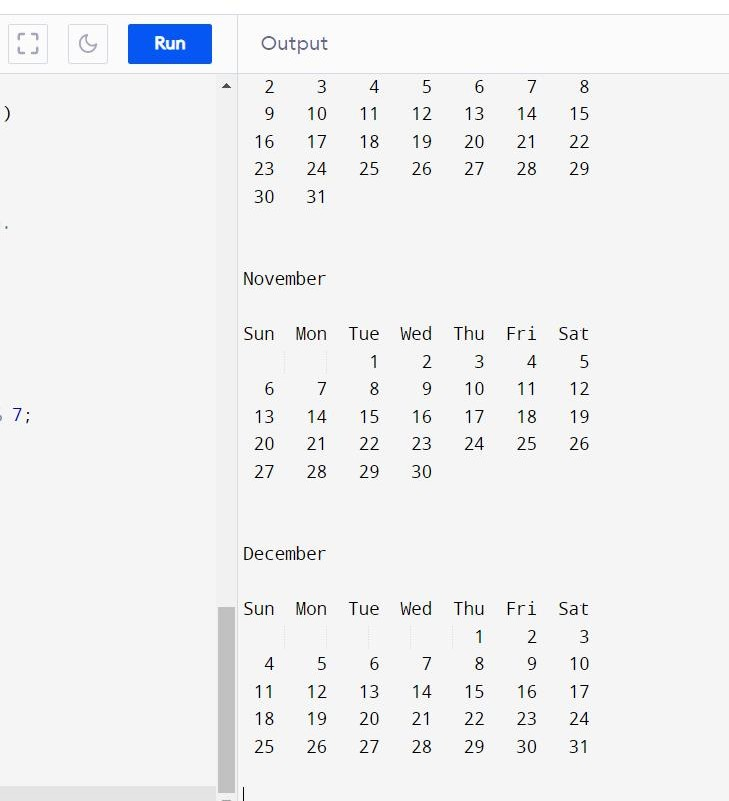
}

# OUTPUT









CONCLUSION

The above program has been successfully executed. We made this program using C language and used its various functions.

This was an interesting project that helped us understand the language more and made it more interesting for us.