**CALENDAR APPLICATION**

**Introduction:**

We all have Laptops, Smart phones where we have Calendar Application which gives daily update about date and day. In this project we are going to implement it with a simple look and feel. It is a Console based Application which means input and output is taken from command prompt, and it is developed using C Programming language. This application does not have any graphics or GUI, but it is simple to understand and use. A User can interact with this application using command prompt.

Calendar displays simple information like displaying dates, days and months of a year based on the input entered by user. We will be using Ubuntu C Terminal which is Linux Environment on Windows for development of this application.

**Features:**

Below are the features will be implemented as part of this Project.

1. Displays Calendar of entire year including Dates, Days and Months where year is taken as input from user through command prompt. Ex: When 2020 is given as input from user this application would display entire Calendar of 2020 year.
2. It also has a feature when a user gives input of Date, Month and Year then corresponding Day is given as output. Ex: When a user gives input like 26 Dec 2020 then it would display Saturday as output.

Font Colours for different parts of the Calendar will be implemented based upon time and feasibility.

**Technical Requirement :** Ubuntu C terminal

**End User of the Project :** Common man who has an idea on command prompt

**Brief Description about how to get day of week by entering dd-mm-yyyy**

Given year, month and day tell what day of week it is (Sunday, Monday etc.).

INPUT: Three numbers year, month and day representing valid date. year must be >0, months and days are numbered from 1. For example 3 January 2016 will be represented by year=2016,month=1 and day=3.

OUTPUT: A single number that represents day of week, we assume that 0 will represent Sunday, 1 Monday, 2 Tuesday, …, 6 Saturday.

Assume that you know what day of week is 31th December of previous year. Given day and month in current year tell what day of week it is.

Formula to solve this problem :

dayOfWeek = (dayOfWeek31DecPrevYear +

numberOfDaysSinceYearStart(month, day)) % 7

To test leap year, we can use the following formula

(year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)

Now we can write numberOfDaysSinceYearStart function:

int numberOfDaysSinceYearStart(int year, int month, int day) {

int[] days = { 0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334 }

int result = days[month-1] + day;

if (month > 2) {

result += isLeap(year) ? 1 : 0;

}

return result;

}

And finally we may implement dayOfWeek function:

int dayOfWeek(int dayOfWeek31DecPrevYear,int year, int month, int day)

{

return(dayOfWeek31DecPrevYear+numberOfDaysSinceYearStart(year,month,day)) % 7;

}

An article from Wikipedia tells us that 1 January 0001 is a Monday. But for us it will be more convenient to count days not from 1 January of given year but from 31 December of the previous year. It is interesting that there is no year 0000, year that precedes year 0001 or 1 AD is 1 BC (or in other words years are intervals not points). To sum up we start counting days from 31 December 1 BC.

We already have function that returns day of week if we know day of week of 31 December of the previous year. We can find out day of week of any date using formula:

dayOfWee31Dec1bc = 0; // Sunday

dayOfWeek31DecPrevYear = (dayOfWeek31Dec1bc+numberOfDaysInYears(year-1)) % 7;

dayOfWeek = dayOfWeek(dayOfWeek31DecPrevYear,year, month, day);

Let’s start by implementing numberOfDaysInYears(year) function that will return number of days since 31 December 1 BC up to 31 December of given year:

int numberOfDaysInYears(int year) {

int normalDays = 365 \* year;

int leapDays = year/4 - year/100 + year/400;

return normalDays+leapDays;

}

And finally we may implement our dayOfWeek routine:

int dayOfWeek(int year, int month, int day) {

int dayOfWeek31Dec1BC = 0; // Sunday int dayOfWeek31DecPrevYear = (dayOfWeek31Dec1BC+numberOfDaysInYears(year1)

) % 7;

int result = dayOfWeek(dayOfWeek31DecPrevYear,year, month, day);

return result;

}

**Project Algorithm/Flowchart :**

1. START
2. Take user input between these 2 choices 1.) Enter Date(dd/mm/yyyy) to find Day of a week 2.) Display calendar of desired Year(yyyy)
3. Read user choice
4. Function to Check whether entered input is correct format according to specific choice or not.
5. IF choice is 1 retrieve year and store it in variable called **currentYear**. ELSE IF Choice is 2 store the value in **currentYear** .Then Call the Function **dayOfWeek31DecPrevYear**.
6. IF Choice is 1 Call the Function **dayOfWeek(year,month,day)**
7. FUNCTION **dayOfWeek(year,month,day)**STARTS:
8. ASSIGN dayOfWeek31Dec1BC = 0 // Sunday
9. ASSIGN dayOfWeek31DecPrevYear = dayOfWeek31Dec1BC + numberOfDaysInYears(year-1) ) % 7

Here we are calling Function numberOfDaysInYears which goes to STEP 14

1. ASSIGN result = dayOfWeek( dayOfWeek31DecPrevYear, year, month, day)
2. RETURN result
3. FUNCTION **dayOfWeek** STOPS:
4. FUNCTION **numberOfDaysInYears(year)** STARTS:
5. ASSIGN Number of normalDays = 365 \* year
6. ASSIGN Number of leapDays = year/4 - year/100 + year/400;
7. RETURN normalDays+ leapDays, from here logic goes to STEP 9
8. FUNCTION **numberOfDaysInYears(year)** STOPS:
9. FUNCTION **dayOfWeek(** int **dayOfWeek31DecPrevYear,** int **year,** int **month,** int **day)** STARTS:
10. RETURN ( dayOfWeek31DecPrevYear + numberOfDaysSinceYearStart(year, month, day) ) % 7, Here numberOfDaysSinceYearStart calculates days since year start
11. FUNCTION **dayOfWeek** STOPS:
12. Function **numberOfDaysInYears(year)** STARTS:
13. ASSIGNdays = { 0, 31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334 }; this array consists of number of days in year before month first day.
14. ASSIGN result = days[month-1] + day
15. IF month > 2 THEN result = result+ isLeap(year) ? 1 : 0

Here, we are calling isLeap(year)function which is in STEP 26

1. RETURN result, Here it goes to Step 19
2. Function **numberOfDaysInYears** STOPS:
3. FUNCTION **isLeap**(**int** year)STARTS:
4. RETURN (year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)
5. FUNCTION **isLeap** STOPS:
6. Function **numberOfDaysInYears** STOPS:
7. From STEP 19 we get a Integer between 0-6 where Sunday represents 0 and Saturday represents 6 respectively
8. IF Choice ==2 Then Follow below lines of Code
9. DECLARE an array which consists all the months of the year

ASSIGN months[] = {"January","February","March","April","May","June","July","August","Septembe r","October","November","December"};

1. DECLARE an array which consists total number of days of each month

ASSIGN monthDay[]={31,28,31,30,31,30,31,31,30,31,30,31};

1. ASSIGN startingDay= **dayOfWeek(year,month,day)+1**
2. We use 3 FOR loops to get the Calendar and month in an order, where First For Loop loops through Months, Second For Loop goes through Days and then Third For Loop goes through days to PRINT The CALENDAR.
3. STOP