**Composite Design on Programming**

**Lab Report**

|  |  |  |
| --- | --- | --- |
| **Version** | **Authors, Student ID, email,** | **Date** |
| V1.0 |  | 2019-6-14 |
|  |  |  |

1. **Deadline: 2019-6-14, 12pm**
2. **You have to complete the whole project and hand in a Lab report, all source code, and snapshots of program output or results before the deadline. If you miss the deadline, just get 0 mark for the composite design on programming.**
3. **If anybody copies any code from another student, both of them get 0 mark.**
4. **Please store your source code (e.g. .h .cpp) in one folder. And put snapshots of running result or description into the report below.**

**After you finish the whole project, all of your report doc(with snapshot and description), source code should be packed into a zip file with the following format.**

**CompositeProgramming \_ InternetStudent \_StudentID\_FullName.zip**

**E.g.** **CompositeProgramming \_InternetStudent\_20150103301\_David.zip**

1. **Please send emails with zip above to 1530456693@qq.com, and the file should be named as “CompositeProgramming+InternetStudentID+Name”.**

**Task: Calendar Project**

**Requirements:**

Write a program which can be used as a calendar. It must provide the following functions:

1. When the program starts, it prints a calendar for current year/month on the screen as follows:

April 2014

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

SU MO TU WE TH FR SA

01 02 03 04 05

06 07 08 09 10 11 12

13 14 15 16 17 18 19

20 21 22 23 24 25 26

27 28 29 30

2. On this interface, you can do one of the following operations:

Press 'N': Show the calendar for the next month;

Press 'P': Show the calendar for the previous month;

Press 'C': Input year and month, the program will show the calendar as you inputted;

Press 'A': Input year/month/day, then input a description, it will add an event to the calendar. Press 'L': Show all events you have created, each one with an ID;

Press 'D': Input event ID, delete the event with this ID;

Press 'S': Input a file name, save the events you have created in this calendar program to a file on the disk;

Press 'R': Input a file name, load the events you have saved;

Press 'Q': Exit the program.

**Implementation**

**Step 1**: Create the program skeleton

You need to create a class named Calendar, which have public member function called void Calendar::Run(). Create an object of this Calendar class and call Run() in your main function.

Calendar class should have other member functions, for example:

void Calendar::Draw() – Used for displaying the calendar for the current month;

void Calendar::Run() – the Calendar driver, accept user input and perform corresponding actions;

Also create member functions for each operation, for example Calendar::NextMonth(), Calendar::PreviousMonth(), Calendar::CreateEvent(), etc.

In this step, you may just print a string for each operation. The actual program logic will be added to the member function in Step 3.

**Step 2**: Create the data structures

You need a Date class, used for storing the date information (Year, Month, Day)

You need an Event class, used for storing the event information (Date, Description)

So, in your Calendar class, you need to have one Date object for storing the current date and a vector of Event class to store the Events you created.

For the Date class, you should write a constructor to initialize the object to the current date. You can use the following sample code to get the current date:

#include <ctime>

int year, month, day;

time\_t t = time(0);

struct tm \* now = localtime(&t);

year = now->tm\_year + 1900;

month = now->tm\_mon + 1;

day = now->tm\_mday;

You may also need to create some member functions for accessing the private member variables, eg: Date::GetYear(), Date::GetMonth(), etc.

**Step 3:** Implement Calendar::Draw()

You may need the following information to help you:

(1) How to determine the day of the week

Use the following formulation:

// Calculate the week for the first day in this month

    int y, c, m;

    if (month == 1 || month == 2)

    {

        y = (year - 1) % 100;

        c = (year - 1) / 100;;

        m = month + 12;

    }

    else

    {

        y = year % 100;

        c = year / 100;

        m = month;

    }

    int week = ((y + y / 4 + c / 4 - 2 \* c +

                26 \* (m + 1) / 10) % 7 + 7) % 7;

if the calculated week equals to 0, it is Sunday; if it equals to 1, it is Monday, and so on.

(2) How to determine the number of days in February

Use the following code sample:

bool leap = false;

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

leap = true;

If it is a leap year, the number of days in February should be 29.

**Step 4:** Implement other date related member functions in the Calendar class.

These functions are relatively easy to implement. Just call the member functions of the date object to change the value of the date object inside the Calendar object.

**Step 5:** Implement the event related functions in the Calendar class

For Create Event: Prompt the user to input related information (date and description for the event), create a new Event object and push it to the events vector in Calendar class;

For List Event: Traverse the events vector and print the events information.

For Delete Event: Prompt the user to input an event ID (which is the subscript of the event in the events vector plus one), remove the related event in the events vector.

**Step 6:** Save and Load functions

Use fstream to save and load text files, you can use ifstream and ofstream to make your life easier.

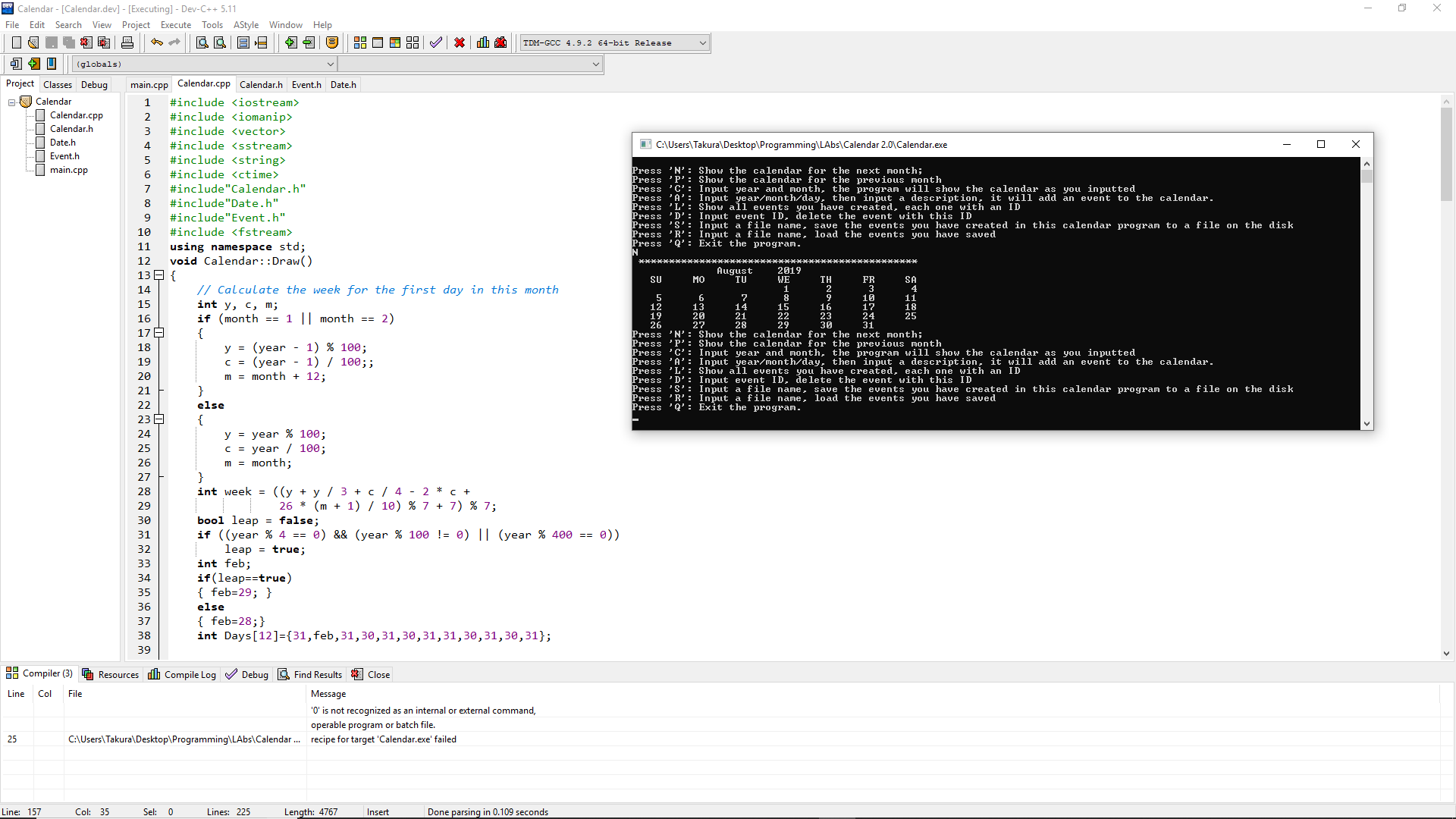
In the file, save the year, month, day and description of the event line by line. When loading the file, use a loop to read back all these information and create Event objects, push the Event objects to the events vector.

Don't forge to close the file stream objects after using them.

**Final Step:** Optimize your program

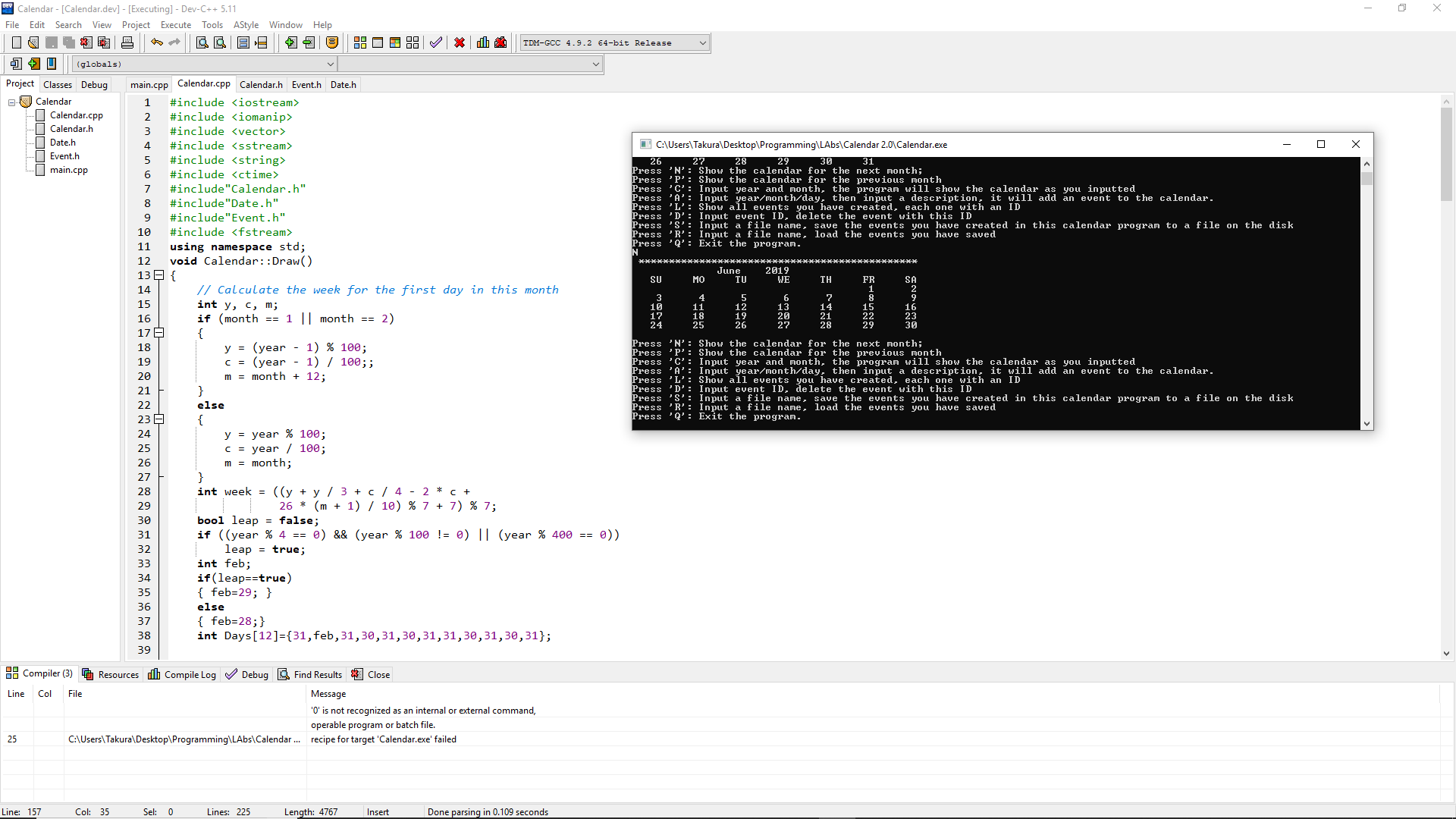
You may find that your code becomes ugly after finishing these steps. For example, you may find some of the function/variable names are not easy for understanding; or you have too much duplicated code which should be encapsulate as functions or classes. Try your best to make your program better.

Step 1



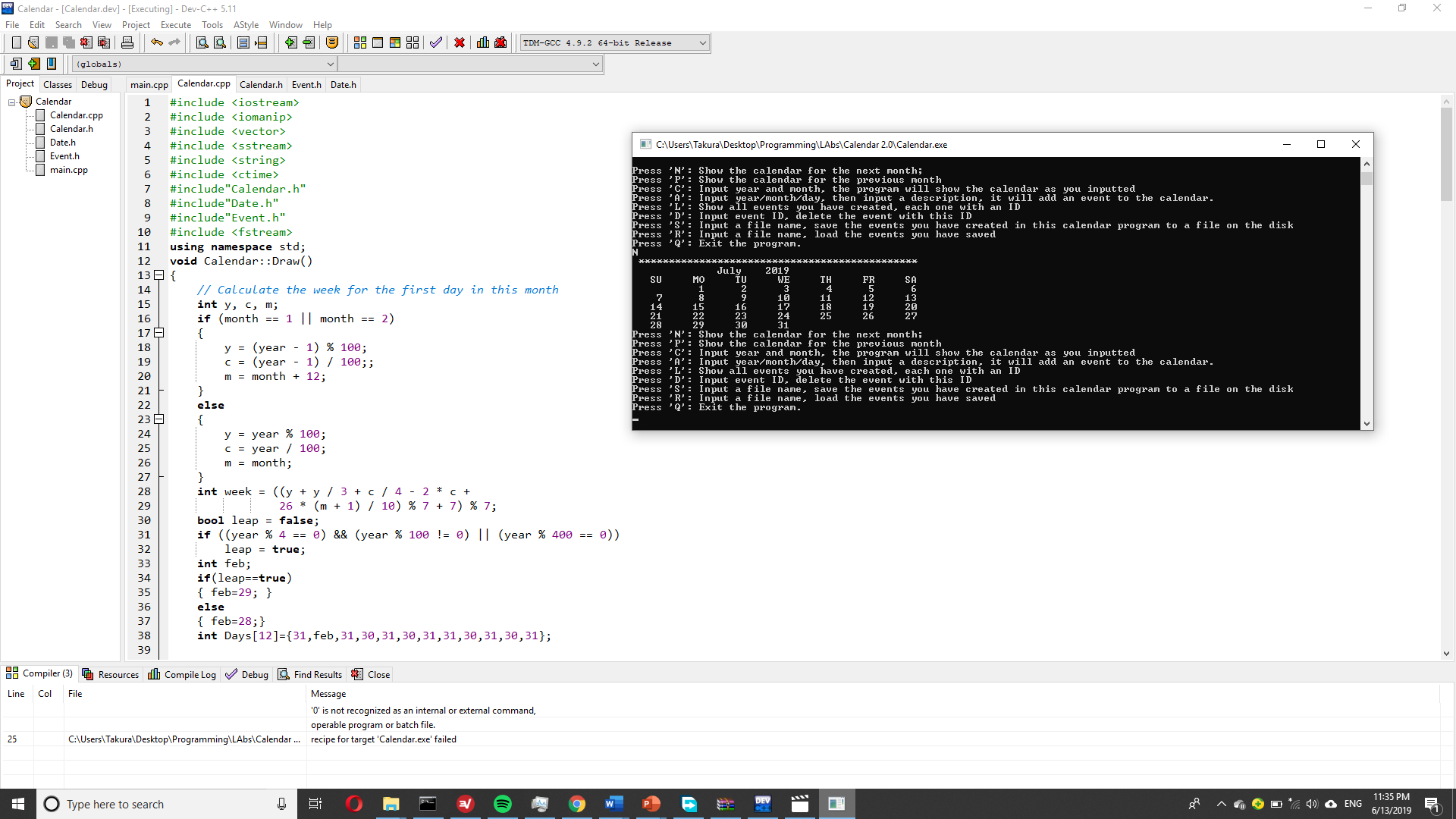
**The calendar program managed to print the calendar for the next month of August**

**Step 2**



**It was able to print the calendar for the previous month June**

**Step 3**



**Printed the calendar of the current month July**

**Step 4**

**Here the user had to input a specific date and the program had to output the calendar for that month that year**

