

Shenyang Aerospace University

Project Design Report

Project Category: C Programming Project

Project Title: The Perpetual Calendar

College: School of Computer Science

Major: Computer Science and Technology


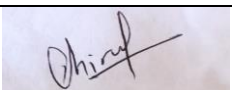
Class: 1751520107

Student ID: 175152010702

Student Name: 丁荣

Supervisor: Zhanglu

Date: 2021/06/16

Shenyang Aerospace University							
Task Specification for Project							
Course Name	C Project			Major	Computer Science and Technology		
Student Name	丁荣	Class No.	1751520107	Student ID	175152010702		
Subject Title	The Perpetual calendar						
Date	2021. 06. 16		To	2021. 06. 30			
Content and Requirement							
<p>Content:</p> <p>Design a program to realize a perpetual calendar:</p> <ol style="list-style-type: none"> 1.Input year、 month, and a calendar of the given month can be displayed; 2.Input year, the whole calendar of the given year can be displayed, and it can turn to next page when one screen is not sufficient; 3.Input year、 month and day, it can display what day it is in the given year; 4.When the input is unreasonable, the program should judge and give warning information; 5.The system should have a menu as follow: <p>*****</p> <ol style="list-style-type: none"> 1.Year and month calendar display 2. Year calendar display 3. Day query 0. Exit <p>*****</p> <p>Requirement:</p> <ol style="list-style-type: none"> 1. Student should complete the design, program and debug himself or herself, and accept the final check by advisor; 2. Complete the project task with C programming language; 3. Compose the project design report in accordance with curriculum design specification. 							
Scrutiny Result: <input checked="" type="checkbox"/> Approved <input type="checkbox"/> Rejected Director(Signature):							
Advisor(Signature)			2021	Y	06	M	1 D
Student(Signature)			2021	Y	06	M	30 D

Contents

Chapter 1 Requirement Analysis.....	1
1.1 Content and requirement of project design	1
1.2 Project analysis	1
Chapter 2 Sketch Design.....	3
2.1 The design of data structure	3
2.2 General structure.....	3
Chapter 3 Detailed Design	4
3.1 The description of data structure.....	4
3.2 Calendar module	4
3.3 Leap Year module	5
Chapter 4 Debug and analysis.....	6
Chapter 5 Test and Execution	8
5.1 Test Plan and TestCases	8
5.2 Sample Execution	8
5.3 Analysis and Conclusions	11
References.....	12
Feedback	12
Appendix: Source Code	13

Chapter 1 Requirement Analysis

1.1 Content and requirement of project design

Design a program to realize a perpetual calendar:

1. Input year、 month, and a calendar of the given month can be displayed;
2. Input year, the whole calendar of the given year can be displayed, and it can turn to next page when one screen is not sufficient;
3. Input year、 month and day, it can display what day it is in the given year;
4. When the input is unreasonable, the program should judge and give warning information;
5. The system should have a menu as follow:

1. Year and month calendar display
2. Year calendar display
3. Day query
0. Exit

1.2 Project analysis

As I analyze the questions which requires obtain the month and year from the user. Then we keep checking day = 1, 2 and so on. As we check a day, we print it to the console. A switch statement is started on the day of the week. For each day, the column is adjusted so that all dates of Sunday fall under the first column, and likewise for others. If the day is a Saturday, the printing is shifted to the next row.

Interactive program that asks a user a calendar year to be displayed. It is a Gregorian type calendar. The program has all its functions be encapsulated in a class (calling it Calendar) that receives the year from the user and have a member function display the calendar.

The calendar has a header for the year given, display the weekdays and the months and the appropriate dates.

For Leap year, a year that has 366 days (29 days in February). There is a leap year every year whose number is perfectly divisible by four- except for years which are both divisible by 100 and not divisible by 400. For example: the century years 1600 and 2000 are leap years, but the century years 1700, 1800 are not. This means that three times out of every for hundred years there are years between leap years. So, to be clear, leap year needs to satisfy either of the two conditions, first condition is if a $\text{year} \% 4 == 0$ meaning if year is divisible by 4 it should not generate any reminder and also $\&\&\text{year} \% 100! = 0$ it should held some reminder. And the other condition is $\|\text{year} \% 400$ meaning when the year is divisible with 400 it should not hold any reminder.

To input year, month and day, which displays day it is in the given year, a string str which represents a date formatted as YYYY-MM-DD, the task is to find the day for the current year. This program makes use of single and multidimensional arrays, for loop and case function. For example, Suppose the user enters the date 1977-11-02 the output will be “The day is: Wednesday” and so on.

And finally, if the user puts the input unreasonable, the program will judge and give the warning function. For example: if the user inputs the year “12” or any unknown year, month or days the system will output “Invalid input... Press any key to continue” and the user will return to the homepage. And in the homepage if the user inputs wrong choice, then the user will get the warning information and will have to try again.

Chapter 2 Sketch Design

2.1 The design of data structure

In my data structure I used char and integer. it's appropriate for the type of program I want to write, such as the questions which requires to find the Days, Date, year and Month, such programs data structure can be char and integer. Switch case are also used checking variable and checking for each case from the menu.

2.2 General structure

The program has been divided into four modules: main menu, yearly calendar, monthly calendar, and day query, and there are three small modules for judging whether it is a leap year, the number of days in the month, and the first day of the month as the day of the week. It can be called by the yearly and monthly calendars, the annual calendar and the monthly calendar. It can be called from the main menu. Finally, we can consider minor issues such as interface optimization and input errors.

For the Main menu the user enters options, and uses a switch statement to perform the next task, and one of the options is used to exit. At the end of the program, use the goto statement to achieve the purpose of returning to the beginning of the function.

For the Monthly calendar, Receive the year and month from the keyboard, and use of small function to get the number of days of the month and the day of the week on the first day of the month, so as to output the monthly calendar on the screen. Change lines when it comes to the weekend.

For yearly calendar, the general idea is similar to the monthly calendar, except that a for loop is used to make the month from January to December. It receives the whole months of the year from the input and use of small function to get the days, months and the day of the week, so as to output the yearly calendar on the screen.

And finally, To Determine the day of the given year input the year, month, and day, and output the information of the day of the week. When the user is done viewing the calendar the user can press 0 and exit the program.

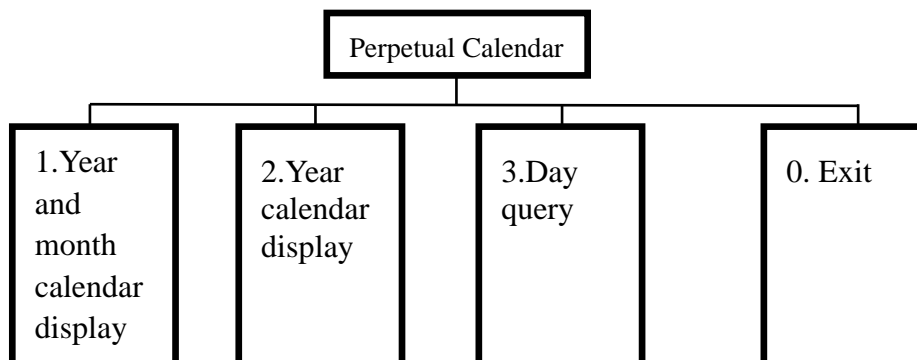


Fig. 2.1 General structure of Perpetual Calendar

Chapter 3 Detailed Design

3.1 The description of data structure

A function is a group of statements that together perform a task. Every C program has at least one function, which is `int` and all the most trivial programs can define additional functions. We can divide up our code into separate functions. How we divide up our code among different functions is up to us, but logically the division is such that each function performs a specific task.

3.2 Calendar module

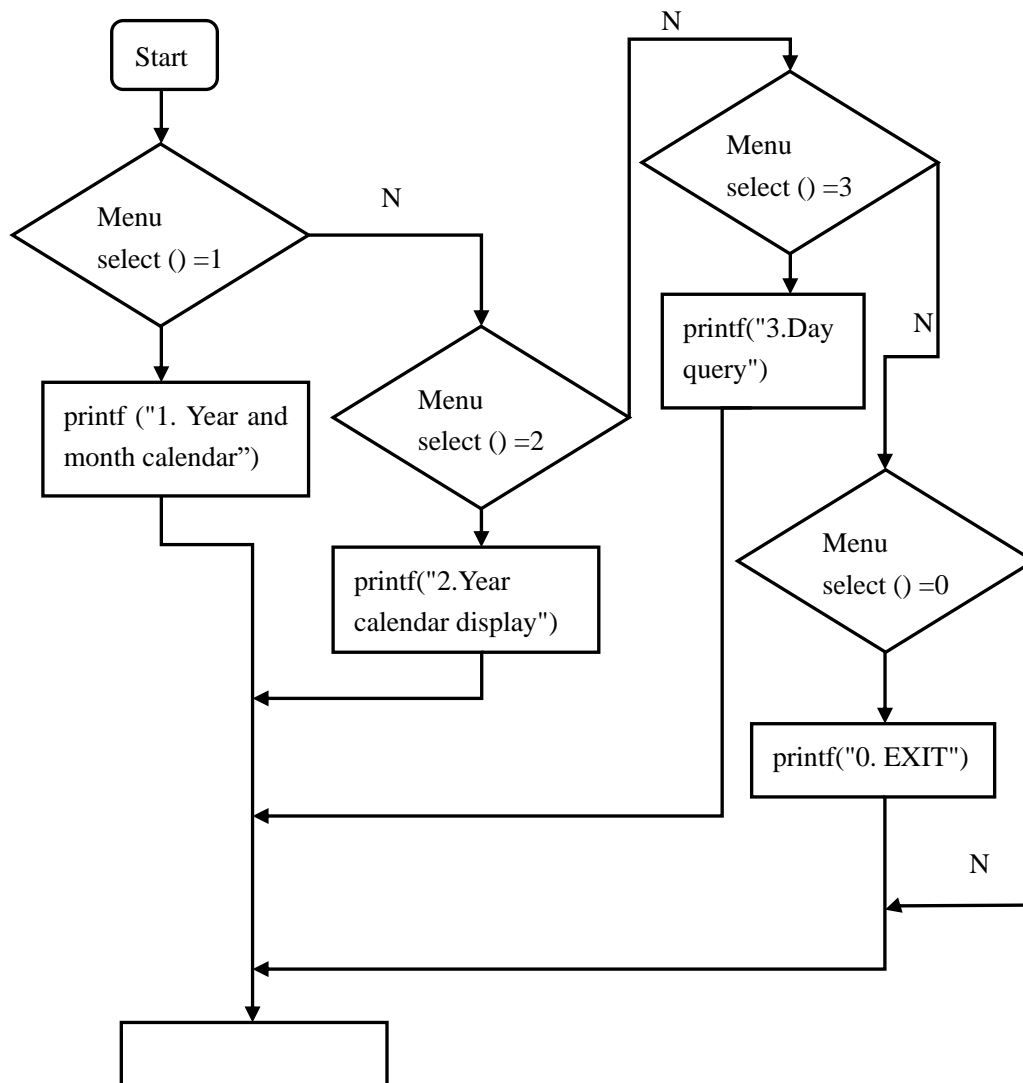


Fig.3.1 Flow chart Perpetual Calendar

- (1) Menu 1 to display year and month of calendar.
- (2) Menu 2 to display year of calendar.
- (3) Menu 3 to query the day.
- (4) Menu 0 to exit.

3.3 Leap Year module

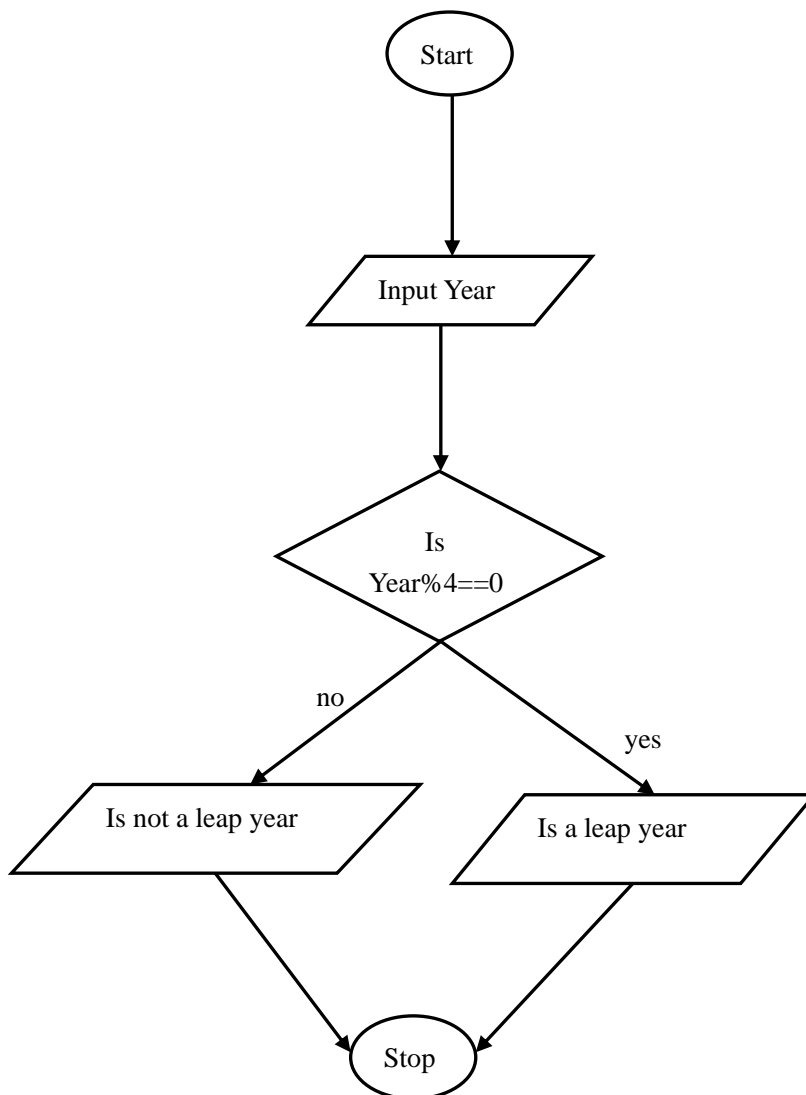


Fig.3.1 Flow chart Leap Year

- (1) If year is divisible by 4 then it is a leap year.
- (2) If year is not divisible by 4 then it is not a leap year.

Chapter 4 Debug and analysis

These are the problems faced when running and after debugging the codes, which are mostly common in writing codes at most as follows;

(1) Problem:

Pause the Output Console until a key is pressed:

Solution:

```
getch();
```

(2) Problem:

Program control out of the loop:

Solution:

```
break;
```

(3) Problem:

Using loop for the embrace blanket was missing instead of putting semicolons

Solution:

```
for (day=1; day<=daysInMonth; day++) {
```

(4) Problem:

Not getting Correct Leap year.

Solution:

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

```
monthDay[1]=29;
```

(5) Problem:

Trouble viewing Month Names.

Solution:

```
printf("\n %s \n",months[month]);
```

(6) Problem:

Unable to get 12 months

Solution:

```
if(!(mm>=1 && mm<=12)){  
    return("Invalid input for month...");
```

(7) Problem:

Not Getting date

Solution:

```
if(!(dd>=1 && dd<=getNumberOfDays(mm,yy)){  
    return("Invalid input for Date");
```

(8) Problem:

System was printing invalid year

Solution:

```
if(yy>=1600, yy<=2030){  
    day = getDayNumber(dd,mm,yy);  
    day = day%7;  
    return(getName(day));  
}else{  
    return("Invalid input for year...");
```

(9) Problem:

Having trouble switching checking variable and checking for each case

Solution:

```
case 1 :  
case 2 :  
case 3 :  
case 0 :
```

(10) Problem:

forgot to close the code by semicolons

Solution:

```
Return 0;}
```

Chapter 5 Test and Execution

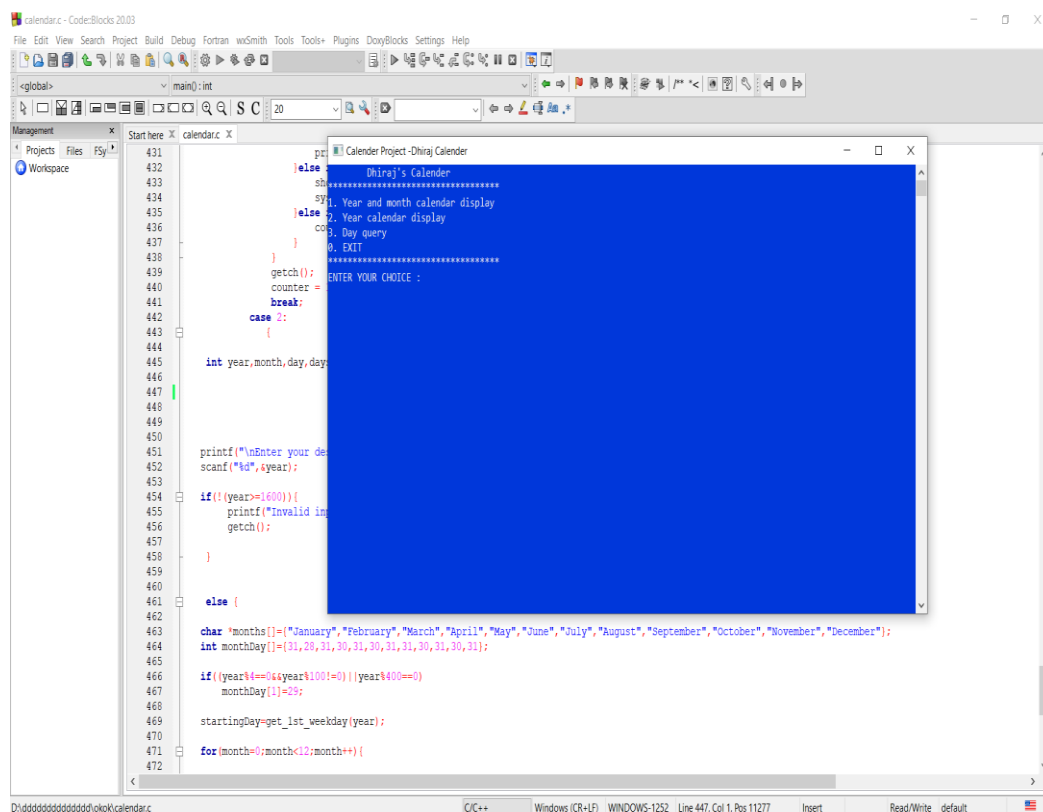
5.1 Test Plan and Test Cases

According to input given, which is viewing the calendar with different requirements, this program output is expected to print the calendar according to year, month and day.

Table 5. 1 A Sample of test case

Test Case Number	Input Values	Expected output
1	(MM YY) 1/2001	Year and Month of 1/2001
2	Year: 2002	The whole calendar of the year 2002.
3	(DD MM YY) 1/1/1990	Day is: Monday
4	Enter Choice: 0	Exit

5.2 Sample Execution



```

431 pr
432 else
433 sh
434 sy1. Year and month calendar display
435 else 2. Year calendar display
436 co 3. Day query
437 }
438 }
439 getch();
440 counter =
441 break;
442 case 0:
443 {
444
445 int year, month, day, day;
446
447
448
449
450 printf("\nEnter your de
451 scanf("%d", &year);
452
453
454 if (! (year >= 1600)) {
455     printf("Invalid im
456     getch();
457 }
458 }
459
460 else {
461
462 char *months[] = {"January", "February", "March", "April", "May", "June", "July", "August", "September", "October", "November", "December"};
463 int monthDay[] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
464
465 if ((year % 4 == 0 && year % 100 != 0) || year % 400 == 0)
466     monthDay[1] = 29;
467
468 startingDay = get_1st_weekday(year);
469
470 for (month = 0; month < 12; month++) {
471
472

```

Fig5.1 Menu

SAU project design report

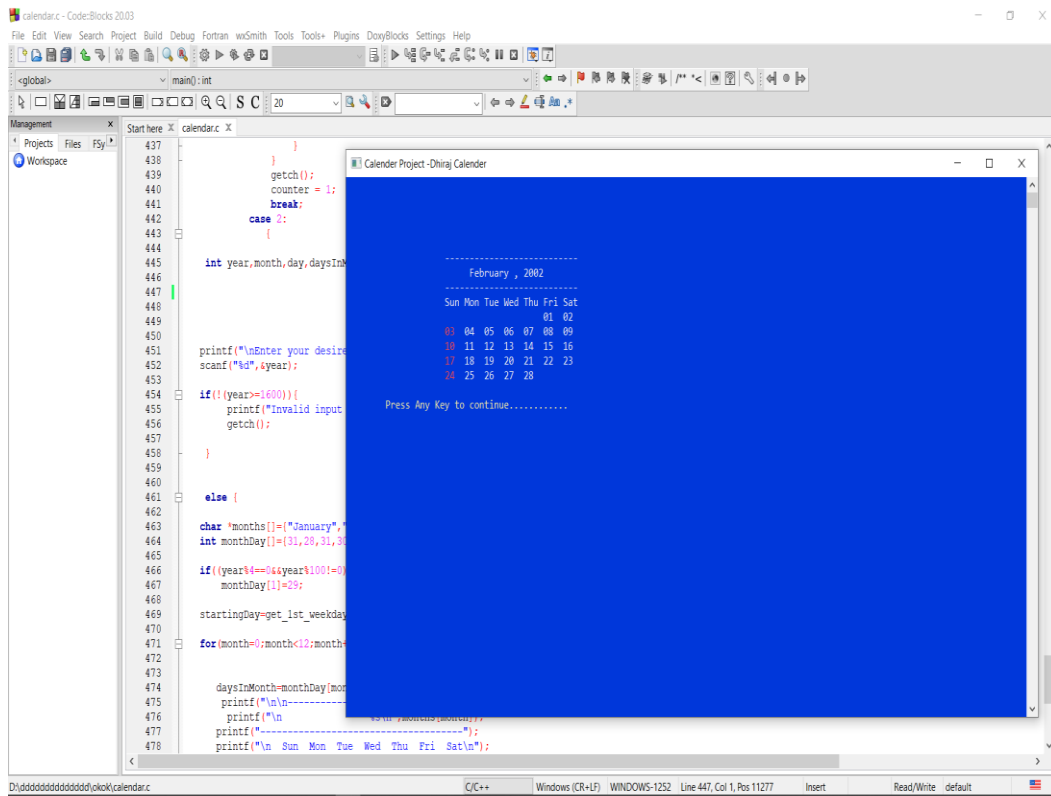


Fig5.2 Year and Month Calendar

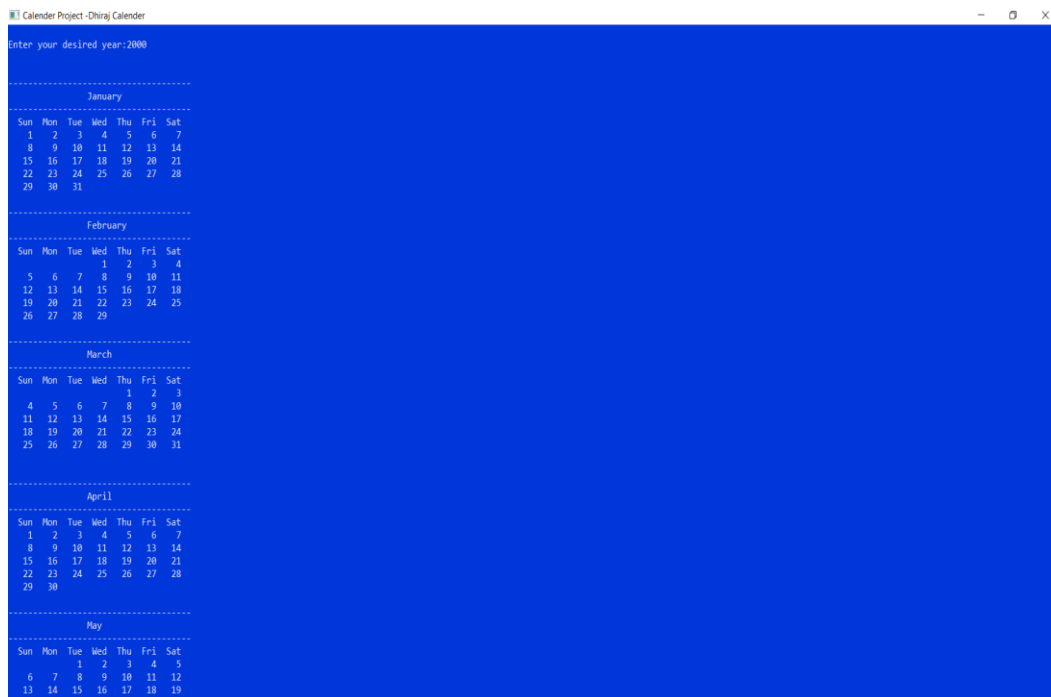


Fig5.3 Year Calendar

SAU project design report

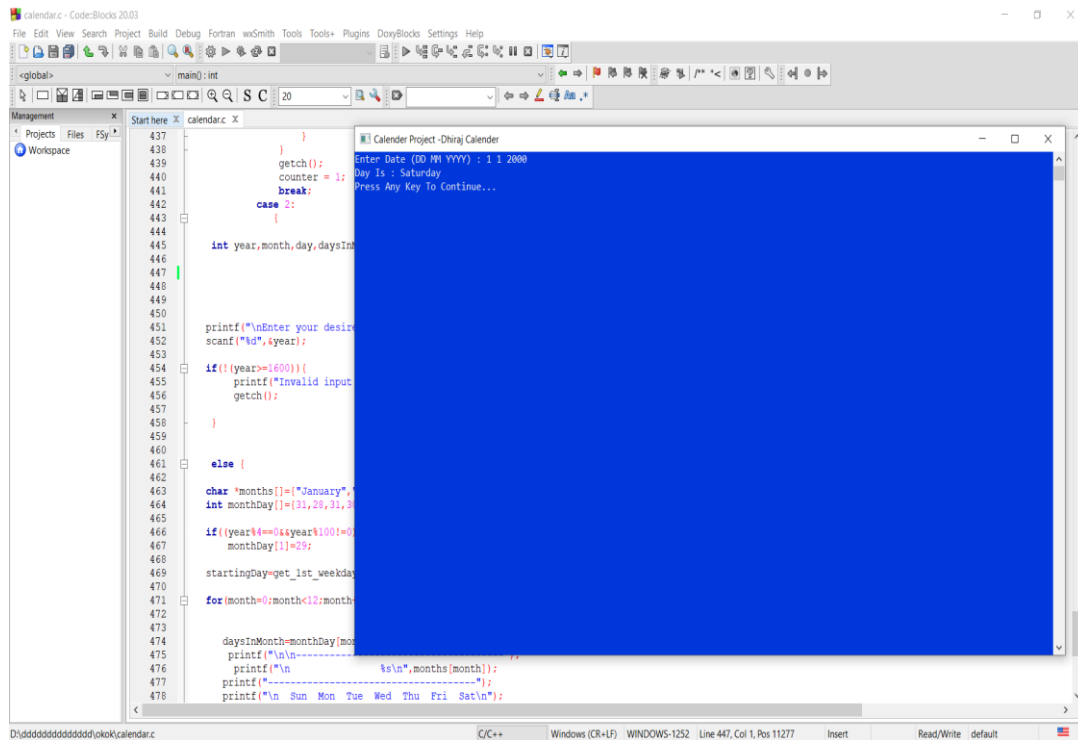


Fig5.4 Day Query

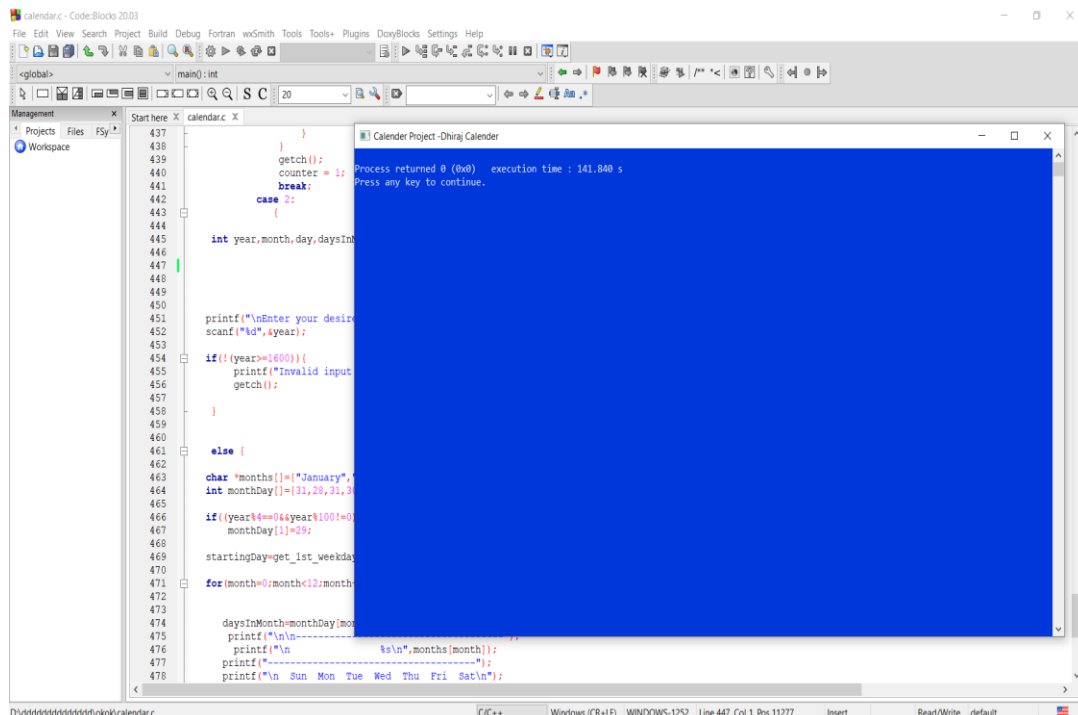


Fig5.5 Exit

5.3 Analysis and Conclusions

- What were the goals or objectives of the project?
 - The goals of the project were to create a Perpetual Calendar that allows the User to view year, month and day of the calendar.
- Did you achieve these goals? Give specifics on how each was accomplished.
 - Yes, the goals were achieved. The goal of entering the menu and viewing the perpetual calendar was done by putting the value displayed in the output. For example: 1 to view the month of calendar and 0 to exit and so on...
- Looking back on the project what would you do differently in the design phase and in the coding and testing phases?
 - In the design phase, I would have added a “Leap year” feature to allow user to check whether the given year is a leap year or not. In the coding, I would have incorporated switch case feature where there were multiple if-else in a single module or part of a feature.
- What additions would you suggest to improve the working of your program?
 - I would suggest to add the reminder and note to the program so that the user can put the event of the day they want, so that if the user forgets the program on the day, they have the system will notify the user of the given event on that day.

References

- [1] E Balagurusamy, Programming in ANSCI C, 5th Edition, 2011/10/01.
- [2]<https://stackoverflow.com/questions/1442116/how-to-get-the-date-and-time-values-in-a-c-program>
- [3]<https://stackoverflow.com/questions/37082316/return-number-of-days-in-a-given-month-of-a-given-year>
- [4]Ward, Terry A. (August 1983). "Annotated C/A Bibliography of the C Language". Byte. p. 268. Retrieved 31 January 2015.
- [5]Ritchie, Dennis M. (1993). "The Development of the C Language". History of Programming Languages, 2nd Edition. Retrieved 2018-11-11.
- [6]Kernighan, Brian; Ritchie, Dennis M. (March 1988). The C Programming Language (2nd ed.).
- [7] Kernighan, Brian W.; Ritchie, Dennis M. (February 1978). The C Programming Language (1st ed.).
- [8] https://www.tutorialspoint.com/cprogramming/switch_statement_in_c.htm

Feedback

Was it too difficult, or too easy?

- Difficulty level was average.

Was the assignment fun or challenging?

- It was both fun and challenging.

Was there something that was unclear?

- The project was crystal clear.

Was the project too long for the given amount of time?

- For me I completed the given work in a week, So I think the amount of time given was ok.

Appendix: Source Code

```
#include<stdlib.h>
#include<stdio.h>
#include<conio.h>
#include<windows.h>
struct Date{
    int dd;
    int mm;
    int yy;
};
struct Date date;

struct Remainder{
    int dd;
    int mm;
    char note[50];
};
struct Remainder R;
int check_leapYear(int year){ //checks whether the year passed is leap year or not//
    if(year % 400 == 0 || (year % 100!=0 && year % 4 ==0))
        return 1;
    return 0;
}

void increase_month(int *mm,  int *yy){ //increase the month by one
    ++*mm;
    if(*mm > 12){
        ++*yy;
        *mm = *mm - 12;
    }
}

void decrease_month(int *mm,  int *yy){ //decrease the month by one
    --*mm;
    if(*mm < 1){
        --*yy;
        if(*yy<1600){
            printf("No Record Available");
        }
    }
}
```



```
        return;
    }
    *mm = *mm + 12;
}
}
```

```
int getNumberOfDays(int month,int year){ //returns the number of days in given
month
```

```
    switch(month){                                     //and year
        case 1 : return(31);
        case 2 : if(check_leapYear(year)==1)
            return(29);
            else
            return(28);
        case 3 : return(31);
        case 4 : return(30);
        case 5 : return(31);
        case 6 : return(30);
        case 7 : return(31);
        case 8 : return(31);
        case 9 : return(30);
        case 10: return(31);
        case 11: return(30);
        case 12: return(31);
        default: return(-1);
    }
}
```

```
char *getName(int day){ //returns the name of the day//
```

```
    switch(day){
        case 0 :return("Sunday");
        case 1 :return("Monday");
        case 2 :return("Tuesday");
        case 3 :return("Wednesday");
        case 4 :return("Thursday");
        case 5 :return("Friday");
        case 6 :return("Saturday");
        default:return("Error in getName() module.Invalid argument passed");
    }
}
```

```
void print_date(int mm, int yy){ //prints the name of month and year//
```

```
printf("-----\n");
gotoxy(25,6);
switch(mm){
    case 1: printf("January"); break;
    case 2: printf("February"); break;
    case 3: printf("March"); break;
    case 4: printf("April"); break;
    case 5: printf("May"); break;
    case 6: printf("June"); break;
    case 7: printf("July"); break;
    case 8: printf("August"); break;
    case 9: printf("September"); break;
    case 10: printf("October"); break;
    case 11: printf("November"); break;
    case 12: printf("December"); break;
}
printf(" , %d", yy);
gotoxy(20,7);
printf("-----");
}

int getDayNumber(int day,int mon,int year){ //returns the day number
    int res = 0, t1, t2, y = year;
    year = year - 1600;
    while(year >= 100){
        res = res + 5;
        year = year - 100;
    }
    res = (res % 7);
    t1 = ((year - 1) / 4);
    t2 = (year-1)-t1;
    t1 = (t1*2)+t2;
    t1 = (t1%7);
    res = res + t1;
    res = res%7;
    t2 = 0;
    for(t1 = 1;t1 < mon; t1++){
        t2 += getNumberOfDays(t1,y);
    }
    t2 = t2 + day;
    t2 = t2 % 7;
    res = res + t2;
    res = res % 7;
    if(y > 2000)
```

```
        res = res + 1;
    res = res % 7;
    return res;
}

char *getDay(int dd,int mm,int yy){
    int day;
    if(!(mm>=1 && mm<=12)){
        return("Invalid input for month...");
    }
    if(!(dd>=1 && dd<=getNumberOfDays(mm,yy))){
        return("Invalid input for Date");
    }
    if(yy>=1600){
        day = getDayNumber(dd,mm,yy);
        day = day%7;
        return(getName(day));
    }else{
        return("Invalid input for year...");
    }
}

int checkNote(int dd, int mm){
    FILE *fp;
    fp = fopen("note.dat","rb");
    if(fp == NULL){
        printf("Error in Opening the File");
    }
    while(fread(&R,sizeof(R),1,fp) == 1){
        if(R.dd == dd && R.mm == mm){
            fclose(fp);
            return 1;
        }
    }
    fclose(fp);
    return 0;
}

void printMonth(int mon,int year,int x,int y){ //prints the month with all days
    int nod, day, cnt, d = 1, x1 = x, y1 = y, isNote = 0;
    if(!(mon>=1 && mon<=12)){
        printf("Invalid input for month...\nPress Any Key To Continue");
        getch();
    }
```

```
        return;
    }
    if(!(year>=1600)){
        printf("Invalid input for year...\nPress Any Key To Continue");
        getch();
        return;
    }
    gotoxy(20,y);
    print_date(mon,year);
    y += 3;
    gotoxy(x,y);
    printf("Sun Mon Tue Wed Thu Fri Sat    ");
    y++;
    nod = getNumberOfDays(mon,year);
    day = getDayNumber(d,mon,year);
    switch(day){ //locates the starting day in calender
        case 0 :
            x=x;
            cnt=1;
            break;
        case 1 :
            x=x+4;
            cnt=2;
            break;
        case 2 :
            x=x+8;
            cnt=3;
            break;
        case 3 :
            x=x+12;
            cnt=4;
            break;
        case 4 :
            x=x+16;
            cnt=5;
            break;
        case 5 :
            x=x+20;
            cnt=6;
            break;
        case 6 :
            x=x+24;
            cnt=7;
```

```
        break;
    default :
        printf("INVALID DATA FROM THE getOddNumber()MODULE");
        return;
}
gotoxy(x,y);
if(cnt == 1){
    SetColor(12);
}
if(checkNote(d,mon)==1){
    SetColorAndBackground(15,12);
}
printf("%02d",d);
SetColorAndBackground(15,1);
for(d=2;d<=nod;d++){
    if(cnt%7==0){
        y++;
        cnt=0;
        x=x1-4;
    }
    x = x+4;
    cnt++;
    gotoxy(x,y);
    if(cnt==1){
        SetColor(12);
    }else{
        ClearColor();
    }
    if(checkNote(d,mon)==1){
        SetColorAndBackground(15,12);
    }
    printf("%02d",d);
    SetColorAndBackground(15,1);
}
gotoxy(8, y+2);
SetColor(14);
printf("Press Any Key to continue.....");
gotoxy(8,y+3);

ClearColor();
}
```

```
int get_1st_weekday(int year){
    int d;
    d = (((year - 1) * 365) + ((year - 1) / 4) - ((year - 1) / 100) + ((year) / 400) + 1) % 7;
    return d;}

void AddNote(){
    FILE *fp;
    fp = fopen("note.dat","ab+");
    system("cls");
    gotoxy(5,7);
    printf("Enter the date(DD/MM): ");
    scanf("%d%d",&R.dd, &R.mm);
    gotoxy(5,8);
    printf("Enter the Note(50 character max): ");
    fflush(stdin);
    scanf("%[^\n]",R.note);
    if(fwrite(&R,sizeof(R),1,fp)){
        gotoxy(5,12);
        puts("Note is Saved Successfully");
        fclose(fp);
    }else{
        gotoxy(5,12);
        SetColor(12);
        puts("\aFail to Save!!\a");
        ClearColor();
    }
    gotoxy(5,15);
    printf("Press Any Key.....");
    getch();
    fclose(fp);
}

void showNote(int mm){
    FILE *fp;
    int i = 0, isFound = 0;
    system("cls");
    fp = fopen("note.dat","rb");
    if(fp == NULL){
        printf("Error in Opening the File");
    }
    while(fread(&R,sizeof(R),1,fp) == 1){
        if(R.mm == mm){
            gotoxy(10,5+i);
```

```
        printf("Note %d Day = %d: %s", i+1, R.dd, R.note);
        isFound = 1;
        i++;
    }
}
if(isFound == 0){
    gotoxy(10,5);
    printf("This Month Contains No Note");
}
gotoxy(10,7+i);
printf("Press any key to back.....");
getch();
}
//Main function started//
int main(){
    ClearConsoleToColors(15, 1);
    SetConsoleTitle("Calender Project -Dhiraj Calender");
    int choice;
    char ch = 'a';
    int counter = 1;
    while( counter == 1){
        system("cls");
        printf("\tDhiraj's Calender\n");
        printf("*****\n");

        printf("1. Year and month calendar display\n");
        printf("2. Year calendar display\n");
        printf("3. Day query\n");
        printf("0. EXIT\n");
        printf("*****\n");
        printf("ENTER YOUR CHOICE : ");
        scanf("%d",&choice);
        system("cls");
        switch(choice){
            default:
                printf("WARNING: Your choice is invalid- Please enter a valid
choice from the menu \n");
                printf("\n Press Any Key To Continue... \n");
                getch();
                break;

            case 3://Find the day which can u search//
                printf("Enter Date (DD MM YYYY) : ");
```

```
scanf("%d %d %d",&date.dd,&date.mm,&date.yy);
printf("Day Is : %s",getDay(date.dd,date.mm,date.yy));
printf("\nPress Any Key To Continue...");
getch();
break;
case 1 :

    printf("Enter Month And Year (MM YYYY) : ");
    scanf("%d %d",&date.mm,&date.yy);
    system("cls");
    int counter1 = 1;

    while(counter1 == 1){
        printMonth(date.mm,date.yy,20,5);
        ch = getch();
        if(ch == 'n'){
            increase_month(&date.mm,&date.yy);
            system("cls");
            printMonth(date.mm,date.yy,20,5);
        }else if(ch == 'p'){
            decrease_month(&date.mm,&date.yy);
            system("cls");
            printMonth(date.mm,date.yy,20,5);
        }else if(ch == 's'){
            showNote(date.mm);
            system("cls");
        }else if(ch == 'q'){
            counter1 = 0;
        }
    }
    getch();
    counter = 1;
    break;
case 2:
{

    int year,month,day,daysInMonth,weekDay=0,startingDay;
    printf("\nEnter your desired year:");
    scanf("%d",&year);

    if(!(year>=1600)){
        printf("Invalid input for year...\nPress Any Key To Continue....");
        getch();
    }
}
```



```
    }

    else {
    char
*months[]={ "January", "February", "March", "April", "May", "June", "July", "August", "Se
ptember", "October", "November", "December" };
    int monthDay[]={ 31,28,31,30,31,30,31,31,30,31,30,31 };

    if((year%4==0&&year%100!=0)||year%400==0)
        monthDay[1]=29;

    startingDay=get_1st_weekday(year);

    for(month=0;month<12;month++){

        daysInMonth=monthDay[month];
        printf("\n\n-----");
        printf("\n                %s\n",months[month]);
        printf("-----");
        printf("\n  Sun  Mon  Tue  Wed  Thu  Fri  Sat\n");

        for(weekDay=0;weekDay<startingDay;weekDay++)
            printf("      ");

        for(day=1;day<=daysInMonth;day++){
            printf("%5d",day);

            if(++weekDay>6){
                printf("\n");
                weekDay=0;
            }
            startingDay=weekDay;
        }
    }
}

getch();

        break;
    case 0 ://In this case we can exit
        exit(0);

    }    }
}
```

```
    return 0;  
} //end of main
```