Title of the Mini-Project:- Calendar Application

Submitted in partial fulfillment of the requirements of the degree of
Bachelor of Engineering

By

Name of the Student:
1)Mayuresh Kulkarni - 26
2)Samaan Lambe - 29
3)Abhinav Mahadik - 31
4)Parag Mahajan -32

Supervisor:

Name of Supervisor:

Department of Computer Engineering

Bharati Vidyapeeth College of Engineering, Navi Mumbai

2019-20

Table of Contents

Project Synopsis Report Approval

This project report entitled **Calendar Application** by **Parag Mahajan, Abhinav Mahadik, Samaan Lambe, Mayuresh Kulkarni** is approved for the degree of **Bachelor of Engineering in Computer Engineering**.

Date: __________

__________________ __________________

Internal Examiner **External Examiner**

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

-- --

(Signature) (Signature)

PARAG MAHAJAN - 32 SAMAAN LAMBE - 29

-- --

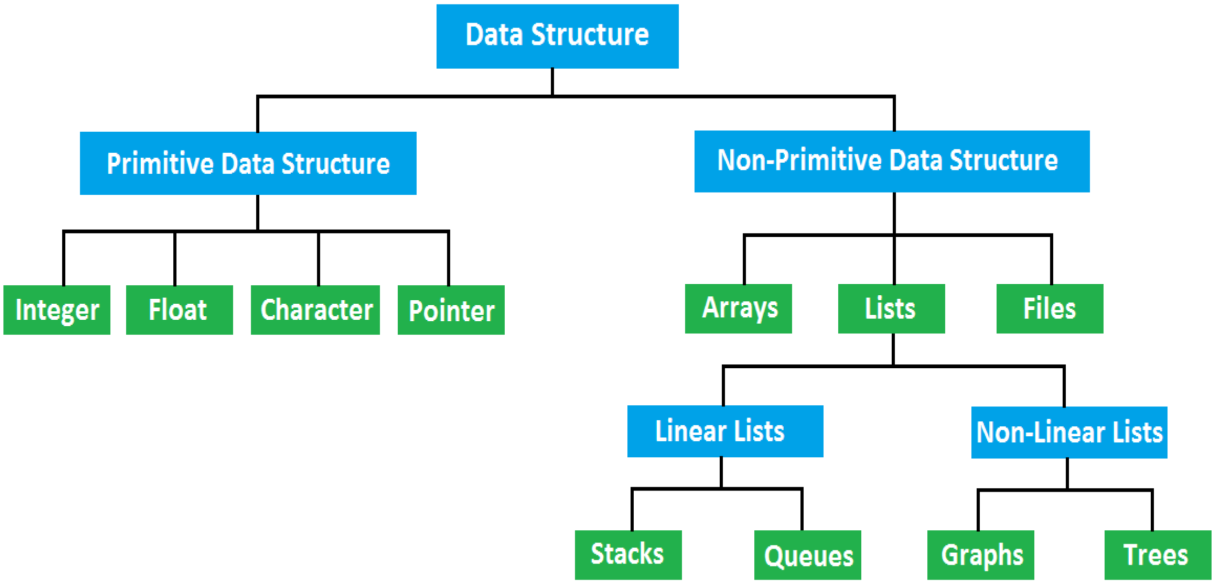
(Signature) (Signature)

MAYURESH KULKARNI - 26 ABHINAV MAHADIK - 31

Date:

Introduction:

It is believed that Data Structures are the building blocks of a program. According to Reema Thareja, 'A data structure is basically a group of data elements that are put together under one name, and which defines a particular way of storing and organizing data in a computer so that it can be used efficiently'. Data Structures has a very crucial role in computing. Data structures are used in almost every program or software system. Some common examples of data structures are arrays, linked lists, queues, stacks, binary trees, and hash tables.



The above diagram shows how we can classify Data Structures into various categories and sub-categories.

Now, what do we do with these data structures? This is where the various different operations come into picture. The following table shows the various possible data operations on the above-mentioned data structures:

NAME OF OPERATION	DESCRIPTION
Traversing	It means to access each data item exactly once so that it can be processed.

Searching	It is used to find the location of one or more data items that satisfy the given constraint
Inserting	It is used to add new data items to the given list of data items
Deleting	It means to remove (delete) a particular data item from the given collection of data items
Sorting	Data items can be arranged in some order like ascending order or descending order depending on the type of application
Merging	Lists of two sorted data items can be combined to form a single list of sorted data items

This mini project on **Calendar in C** programming language using Data Structures is a console application without graphics. This would provide a digital overview to the calendar. Where any part of month/year can be visible, just by executing code.

Algorithm followed is by using concepts of Data structures and File Handling. To make the calendar look colorful, many windows properties can be used further in this project (in order to make it attractive).

In this mini project, you can find out the day corresponding to a given date and view the days and dates corresponding to a particular month & year. The source code is not that long, below 200 lines. It is compiled in Turbo C compiler. This project is aimed to teach you "how to make Calendar using Data Structures in C".

Methodology:

Basically technique/method used to resolve a condition of code was achieved by using Data Structures; i.e. Linked lists and Arrays. A certain type of File Handling to store data in more Dynamic way. Therefore, entire module is approached on Structured programming.

The following are the key points tends to be covered

- Finding an error in a program is easily.
- Syntax and structure are very simple to understand and remember.
- Modification is easy.
- There is no re-usability of existing code as much as expected.

Operators, Operands and Constants constitute to form an expression which increases Simplicity. Three major operations can be done in this calendar application. To find out the day corresponding to a given date, the date, month and year are asked. You can list the days and dates of any month of any year. For example, entering 21 10 2019 (21st October) will give you an output as "MONDAY". The third feature of this C mini project on Calendar application utilizes file handling. With this feature, you can add important notes with corresponding dates. The functions used in the source code are simple and easy to understand. It provides very simple interface.

It's a Menu driven program, so list of options is provided where whatever user chooses to opt for.

1. *FIND THE DAY.*
2. *PRINT A CALENDAR.*
3. *ADD NOTES.*
4. *EXIT* (to source code).

Header files used are:

1. stdio.h
2. conio.h
3. stdlib.h

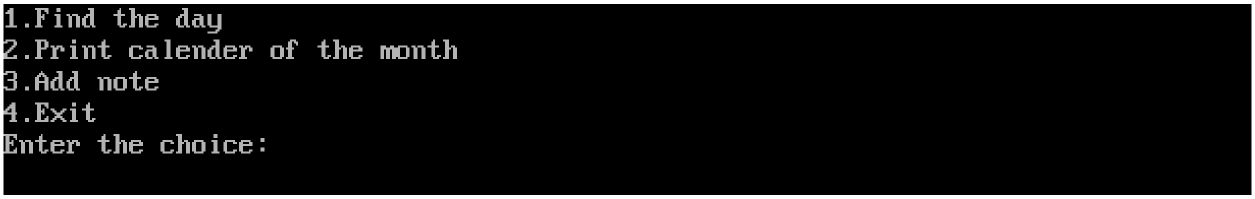
Source Code:

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>

int no(int d,int m,int y);

char *dayName(int day);

typedef struct
{
        int day;
        int month;
        int year;
        char note[255];
}Note;

int isLeapYear(int y);
int leapYears(int y);
int todayOf(int y,int m,int d);
long days(int y,int m,int d);
void calender(int m,int y,int d1,int m1,int y1,char *note);

void main()
{
        int year,month,day,ch;
        Note note1;
        clrscr();
        while(1)
        {
                printf("1.Find the day\n2.Print calender of the month\n3.Add
note\n4.Exit\nEnter the choice:");
                scanf("%d",&ch);
                switch(ch)
                {
                        case 1: printf("Enter the day,month,year:");
                                scanf("%d %d %d",&day,&month,&year);
                                printf("The day is:%s\n",dayName(no(day,month,year)));
                                break;

                        case 2: printf("Enter month and year:");
                                scanf("%d %d",&month,&year);
                                printf("Enter 'n'to see the notes\nEnter any key to
continue:");

        calender(month,year,note1.day,note1.month,note1.year,note1.note);
                                break;

                        case 3: printf("Enter day,month,year:");
                                scanf("%d %d
                                %d",&note1.day,&note1.month,&note1.year);
                                printf("Enter the note:");
                                scanf("%s",&note1.note);
                                printf("Note added successfully.\n");
                                break;

                        case 4: exit(0);
                                break;

                        default:printf("Not a valid entry\n");
                                break;
                }
        }

}

int no(int d,int m,int y)
{
        static int t[]={0,3,2,5,0,3,5,1,4,6,2,4};
        y-=m<3;
        return((y+y/4-y/100+y/400+t[m-1]+d)%7);
}

char *dayName(int day)
{
        switch(day)
        {
                case 0:return("Sunday");
                case 1:return("Monday");
                case 2:return("Tuesday");
                case 3:return("Wednesday");
                case 4:return("Thrusday");
                case 5:return("Friday");
                case 6:return("Saturday");
```

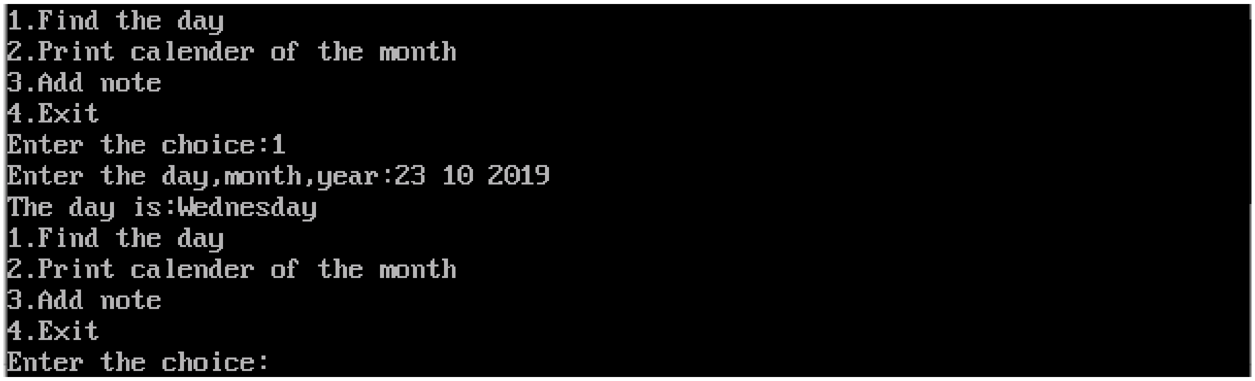
```
                default:return("Error:Invalid arguments passed...\n");
        }
}

int isLeapYear(int y)
{
        return((y%400==0)||((y%4==0)&&(y%100!=0)));
}

int leapYears(int y)
{
        return(y/4-y/100+y/400);
}

int todayOf(int y,int m,int d)
{
        static int dom[]={-1,0,31,59,90,120,151,181,212,243,273,304,334};
        return(dom[m]+d+((m>2&&isLeapYear(y))?1:0));
}

long days(int y,int m,int d)
{
        int lastYear;
        lastYear=y-1;
        return(365L*lastYear+leapYears(lastYear)+todayOf(y,m,d));
}

void calender(int m,int y,int d1,int m1,int y1,char *note)
{
        const char
*nom[]={NULL,"January","February","March","April","May","June","July","August","September","October","November","December"};
        char week[]="  Sun  Mon  Tue  Wed  Thu  Fri  Sat";
        char ch;
        int dom[]={-1,31,28,31,30,31,30,31,31,30,31,30,31};
        int w,i,day,hasNote;
        w=(days(y,m,1))%7;
        if(isLeapYear(y))
                dom[2]=29 ;
        printf("\n          %s  %d\n%s\n",nom[m],y,week);
        for(i=0;i<w;i++)
                printf("     ");
```

```
    for(i=w,day=1;day<=dom[m];i++,day++)
    {
        if((y1==y)&&(m1==m))
        {
            if(d1==day)
            {
                printf(" |%2d|",day);
            }

            else
            {
                printf("%5d",day);
            }
        }

        else
        {
            printf("%5d",day);
        }

if(i%7==6)
            printf("\n");
    }

    printf("\n");
    scanf("\n%c",&ch);
    if(ch=='n')
    {
        if(note!='\0')
        {
            printf("Here are list of notes for %d %d\n",m,y);
            printf("%d:",d1);

            while(*note!='\0')
            {
                printf("%c",*note);
                note++;
            }

            printf("\n");
        }

        else
        {
            printf("No notes to display!!!");
        }
    }

    else
        return;
}
```

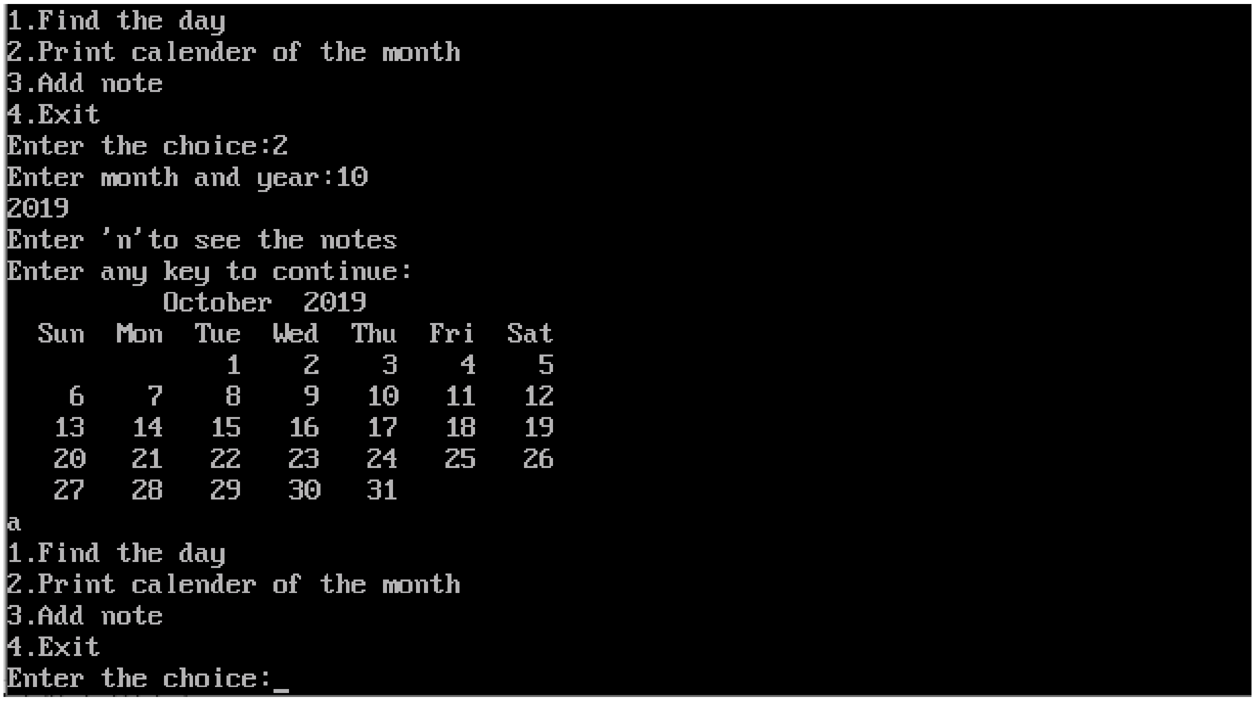
Results / Outputs :



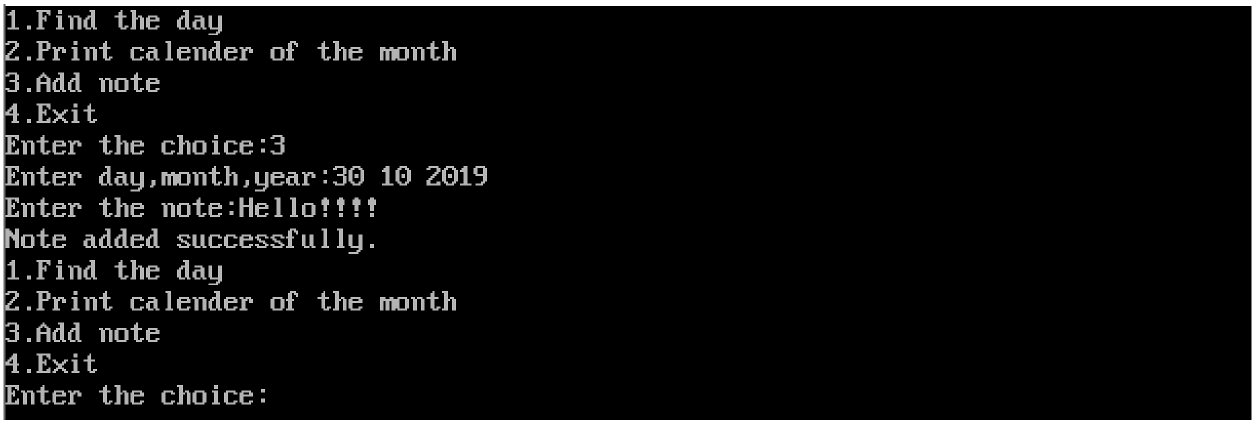
```
1.Find the day
2.Print calender of the month
3.Add note
4.Exit
Enter the choice:
```



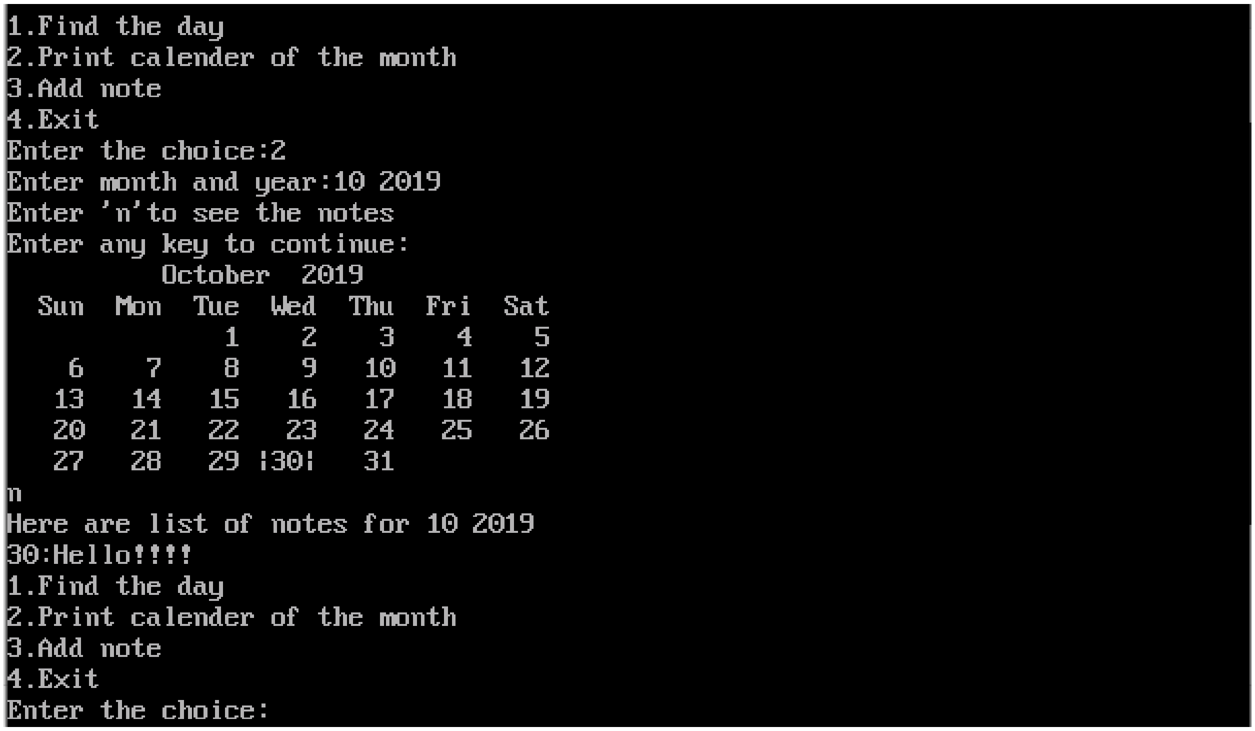
```
1.Find the day
2.Print calender of the month
3.Add note
4.Exit
Enter the choice:1
Enter the day,month,year:23 10 2019
The day is:Wednesday
1.Find the day
2.Print calender of the month
3.Add note
4.Exit
Enter the choice:
```



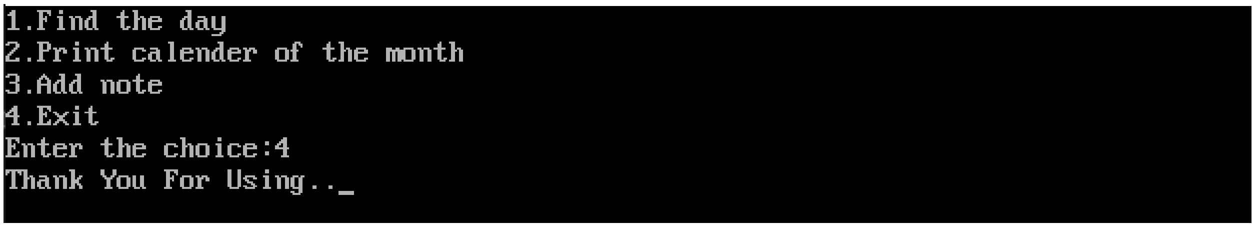
```
1.Find the day
2.Print calender of the month
3.Add note
4.Exit
Enter the choice:2
Enter month and year:10
2019
Enter 'n'to see the notes
Enter any key to continue:
          October  2019
  Sun  Mon  Tue  Wed  Thu  Fri  Sat
                 1    2    3    4    5
    6    7    8    9   10   11   12
   13   14   15   16   17   18   19
   20   21   22   23   24   25   26
   27   28   29   30   31
a
1.Find the day
2.Print calender of the month
3.Add note
4.Exit
Enter the choice:_
```



```
1.Find the day
2.Print calender of the month
3.Add note
4.Exit
Enter the choice:3
Enter day,month,year:30 10 2019
Enter the note:Hello!!!!
Note added successfully.
1.Find the day
2.Print calender of the month
3.Add note
4.Exit
Enter the choice:
```



```
1.Find the day
2.Print calender of the month
3.Add note
4.Exit
Enter the choice:2
Enter month and year:10 2019
Enter 'n'to see the notes
Enter any key to continue:
          October  2019
  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
n
Here are list of notes for 10 2019
30:Hello!!!!
1.Find the day
2.Print calender of the month
3.Add note
4.Exit
Enter the choice:
```



```
1.Find the day
2.Print calender of the month
3.Add note
4.Exit
Enter the choice:4
Thank You For Using..
```

Scope/Advantage:

1. Helping in Digitalization
2. Making people’s daily-life by allowing option of Notes
3. Computer Application helps to save paper
4. Paper-Less India, Digital India
5. Regular paper based calendars are very easy to loose. A calendar installed on your computer will never be lost (unless you format your pc).

Conclusion:

We are at the end of this book. You have probably read everything and also tested some pieces of code with your own hardware. Thus we’ve observed,that proper outputs are perceptible. Therefore, it can be confidently concluded that this code is completely executable; followed with desired outcomes, that were required.

The most important advantage of our project is how professionally it handles the time-space complexity. Since the program length is small, it gets compiled and executed quickly. Thus, it’s Time-Complexity is very less.

On the other hand, since the size of the code is not very big, it takes less space to be stored. Less consumption of space means lesser Space-Complexity.

Again, several aspects of the language have been fulfilled at this stage of the specifications. Hence further we can use more developing languages, and utilize multiple softwares.

References

Books:

1)Data Structures Using C
- Reema Thareja

2)Data Structures Through C
- Yashavant Kanetkar

Website (URL):

1. *http//:www.programming-techniques.com*

2. *http//:www.codewithc.com*

3. *http//:www.codeproject.com*