JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA

"UNITIME - A UNIVERSITY TIMETABLE DECODER (B4 BATCH) DEVELOPED USING C"

A PROJECT REPORT SUBMITTED AS A PART OF THE MINI PROJECT OF SOFTWARE DEVELOPMENTS FUNDAMENTAL LAB - 1

SUBMITTED BY

AKASH SHARMA (NBTG13285)
MUKUND SARDA (NBTG13756)
KARANJOT SINGH (NBTG13346)

ABSTRACT OF PROJECT

We all know it's really difficult to keep a track of your classes and schedules in university with all the ongoing vivas, practical exams, fests. So, we came up with a Project called UniTime to help you to keep track of your classes so you don't need to always lookup for a whole timetable for every class.

UniTime is a Timetable decoder program developed using C language with the help of structures, functions and file handling. In Universities, we have a long timetable which contains the schedule for the whole week and it becomes really difficult to look for each class with all the different abbreviations you need to remember when you are running late for the class. According to a study by wired, an average student spends around 5 mins to look for a class on the timetable. So, with UniTime, we are also saving those 5 mins from your busy university schedule.

UniTime is focused on saving time using this project so whenever you run this program, it will sync with your device time and display a minimal output that tells you about which class you are having with which teacher and at what location. We also added one functionality in which if the user wants to manually check the class, he/she can enter the time and the program will display the class information.

TOPICS OF SDF-1 USED

- **Data File Handling - to import text files containing details about each class**
- **Control flow - includes all the basic functions**
- **Functions - used for printing timetables for different days, etc.**
- **Pointers - used in file handling and addressing of functions**
- **Structures**
- **Array**
- **Strings**

PROJECT DESIGN

So In this project, we ask user for input, if he/she wants to enter the time manually or wants to check the schedule for the current class.

 # If the user wants to check the current class

We have added a date-time structure through which it extracts the date and time from the user's device. Using switch case we are able to print the day and the timetable as per the variables 'day' & 'hours' which contains the text file of different classes according to the day and hour. We have created different functions for printing timetables for different days and in switch statements, we call those functions to print the timetable for that day according to the input provided.

 # if the user manually wants to enter time and check class

In this the user needs to input the day and time to check the class . In this we also use switch case to call those functions to print the timetable for that day and time.

IMPLEMENTATION DETAILS

SOURCE CODE

```
/* SOFTWARE DEVELOPMENT FUNDAMENTALS - I (15B17CI171)
MINI-PROJECT

UNITIME
(Batch-B4 Time Table Decoder)

By:
Akash Sharma NBTG13285
Mukund Sarda NBTG13756
Karanjot Singh NBTG13346 */


#include <stdio.h>

#include <stdlib.h>

#include <time.h>


/* Function for printing U in * */
void U(int i);

/* Function for printing N in * */
void N(int i);

/* Function for printing I in * */
void I(int i);

/* Function for printing T in * */
void T(int i);

/* Function for printing M in * */
void M(int i);

/* Function for printing E in * */
void E(int i);
```

```c
/* Function for finding day of the week */
int dayofweek(int d,int m,int y);

/* Function for reading contents of a text file */
int fileread(char filename[100]);

/* Function for printing time table for Monday */
int monday(int hours);

/* Function for printing time table for Tuesday */
int tuesday(int hours);

/* Function for printing time table for Wednesday */
int wednesday(int hours);

/* Function for printing time table for Thursday */
int thursday(int hours);

/* Function for printing time table for Friday */
int friday(int hours);

/* Function for printing time table for Saturday */
int saturday(int hours);

void U(int i)
{
    int j;

        for(j=1;j<=5;j++)
        {

            if(j==1||j==5||i==9)
                printf("* ");

            else
                printf("  ");

        }
}


void N(int i)
```

```
{
    int j;

        for(j=1;j<=9;j++)
        {

            if(j==1||j==9||i==j)
                printf("*");

            else
                printf(" ");

        }
}


void I(int i)
{
    int j;

        for(j=1;j<=5;j++)
        {

            if(j==3||i==1||i==9)
                printf("* ");

            else
                printf("  ");

        }
}


void T(int i)
{
    int j;

        for(j=1;j<=5;j++)
        {

            if(i==1||j==3)
                printf("* ");
```

```
            else
                printf("   ");

        }
}


void M(int i)
{
    int j;

        for(j=1;j<=9;j++)
        {

            if(j==1||j==9||(j<=5&&i==j)||(j>=5&&i+j==10))
                printf("*");

            else
                printf(" ");

        }
}


void E(int i)
{
    int j;

        for(j=1;j<=5;j++)
        {

            if(j==1||i==1||(i==5&&j<=4)||i==9)
                printf("* ");

            else
                printf("   ");

        }
}
```

```c
int dayofweek(int d,int m,int y)
{

    static int t[]={ 0, 3, 2, 5, 0, 3, 5, 1, 4, 6, 2, 4 };

    y -= m < 3;

    /* Formula for calculating day of the week */
    return ( y + y/4 - y/100 + y/400 + t[m-1] + d) % 7;

}


int fileread(char filename[100])
{

    /* Declaring pointer for reading the file */
    FILE *fptr;
    char c;

    /* Opening the file in read only mode */
    fptr = fopen(filename, "r");

    while (c != EOF)
    {
        printf ("%c", c);
        c = fgetc(fptr);
    }

    printf("\n");

    /* Closing the file */
    fclose(fptr);

    return 0;
}


int monday(int hours)
{
    switch(hours)
    {
```

```
        case 10: fileread("monday1011.txt");
        break;

        case 11: fileread("monday1112.txt");
        break;

        case 12: printf(" Lunch Break!\n");
        break;

        case 13: fileread("monday1314.txt");
        break;

        case 15: fileread("monday1517.txt");
        break;

        case 16: fileread("monday1517.txt");
        break;

        default: printf(" No Class!\n");
        break;

    }
}


int tuesday(int hours)
{
    switch(hours)
    {

        case 10: fileread("tuesday1011.txt");
        break;

        case 12: printf(" Lunch Break!\n");
        break;

        case 13: fileread("tuesday1315.txt");
        break;

        case 14: fileread("tuesday1315.txt");
        break;
```

```
        case 15: fileread("tuesday1516.txt");
        break;

        case 16: fileread("tuesday1617.txt");
        break;

        default: printf(" No Class!\n");
        break;

    }
}


int wednesday(int hours)
{
    switch(hours)
    {

        case 10: fileread("wednesday1011.txt");
        break;

        case 11: fileread("wednesday1112.txt");
        break;

        case 12: printf(" Lunch Break!\n");
        break;

        case 13: fileread("wednesday1314.txt");
        break;

        default: printf(" No Class!\n");
        break;

    }
}


int thursday(int hours)
{
    switch(hours)
    {
```

```
        case 9: fileread("thursday0912.txt");
        break;

        case 10: fileread("thursday0912.txt");
        break;

        case 11: fileread("thursday0912.txt");
        break;

        case 12: printf(" Lunch Break!\n");
        break;

        case 13: fileread("thursday1314.txt");
        break;

        case 14: fileread("thursday1415.txt");
        break;

        case 16: fileread("thursday1617.txt");
        break;

        default: printf(" No Class!\n");
        break;

    }
}


int friday(int hours)
{
    switch(hours)
    {

        case 10: fileread("friday1011.txt");
        break;

        case 11: fileread("friday1112.txt");
        break;

        case 12: printf(" Lunch Break!\n");
        break;
```

```
        case 13: fileread("friday1314.txt");
        break;

        case 15: fileread("friday1517.txt");
        break;

        case 16: fileread("friday1517.txt");
        break;

        default: printf(" No Class!\n");
        break;

    }
}


int saturday(int hours)
{
    switch(hours)
    {

        case 9: fileread("saturday0911.txt");
        break;

        case 10: fileread("saturday0911.txt");
        break;

        case 11: fileread("saturday1112.txt");
        break;

        case 12: printf(" Lunch Break!\n");
        break;

        default: printf(" No Class!\n");
        break;

    }
}


int main(void)
```

```
{

    printf("\n");

    /* Printing UNITIME using * */
    for(int a=1;a<=9;a++)
    {
        printf(" ");
        U(a);

        printf("   ");
        N(a);

        printf("   ");
        I(a);

        printf("   ");
        T(a);

        printf("   ");
        I(a);

        printf("   ");
        M(a);

        printf("   ");
        E(a);

        printf("   ");

        printf("\n");
    }

    printf(" \n ------------------------------");

    printf("\n | Batch-B4 Time Table Decoder |\n");

    printf(" ------------------------------\n\n");

    int z;

    /* Menu of the program */
```

```
    printf(" Options Available:\n\n");

    printf(" 1. Find time table as per current day, date and time\n\n");

    printf(" 2. Find time table by entering day and time manually\n\n");

    printf(" Enter your choice: ");

    scanf("%d",&z);

    /* Executing program to print time table as per current parameters
*/
    if (z==1)
    {
        time_t now;

        time(&now);

        int hours, minutes, seconds, date, month, year, day;

        /* Structure having local time and date information */
        struct tm *local = localtime(&now);

        /* Extracting current hours from the structure */
        hours = local->tm_hour;

        /* Extracting current minutes from the structure */
        minutes = local->tm_min;

        /* Extracting current seconds from the structure */
        seconds = local->tm_sec;

        /* Extracting current date from the structure */
        date = local->tm_mday;

        /* Extracting current month from the structure */
        month = local->tm_mon + 1;

        /* Extracting current year from the structure */
        year = local->tm_year + 1900;
```

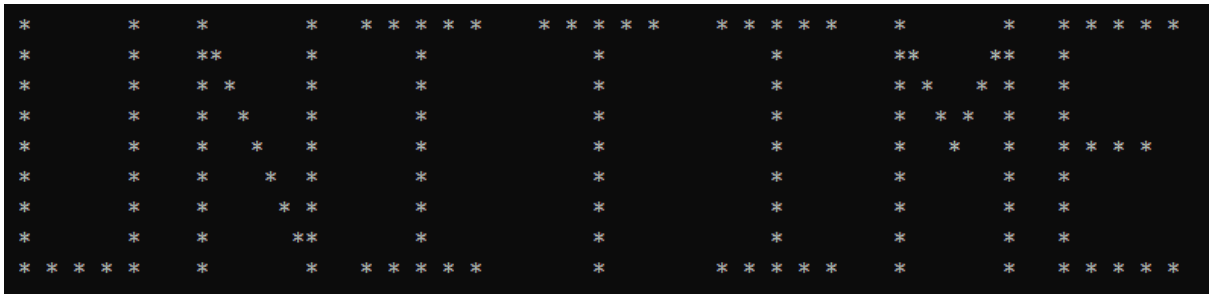
```
        /* Printing Current Date */
        printf("\n Date is: %d/%d/%d\n\n", date, month, year);

        /* Printing Current Time */
        printf(" Time is: %d:%d:%d\n\n", hours, minutes, seconds);

        /* Finding the day of the week as per the obtained values of
date, month, year */
        day = dayofweek(date,month,year);

        /* Printing day and the time table as per the value of variables
'day' & 'hours' */
        switch(day)
        {

            case 1: printf(" Day is: Monday\n\n");

            /* Calling function monday() for printing time table */
            monday(hours);
            break;

            case 2: printf(" Day is: Tuesday\n\n");

             /* Calling function tuesday() for printing time table */
            tuesday(hours);
            break;

            case 3: printf(" Day is: Wednesday\n\n");

             /* Calling function wednesday() for printing time table */
            wednesday(hours);
            break;

            case 4: printf(" Day is: Thursday\n\n");

             /* Calling function thursday() for printing time table */
            thursday(hours);
            break;

            case 5: printf(" Day is: Friday\n\n");

             /* Calling function friday() for printing time table */
```

```
            friday(hours);
            break;

            case 6: printf(" Day is: Saturday\n\n");

             /* Calling function saturday() for printing time table */
            saturday(hours);
            break;

            case 7: printf(" Day is: Sunday\n\n");

            printf(" No Class!\n\n");
            break;

        }

    }

    /* Executing program to print time table as per entered parameters
*/
    else if (z==2)
    {

        int hours, minutes, day;

        printf("\n Enter 1 for Monday\n");

        printf(" Enter 2 for Tuesday\n");

        printf(" Enter 3 for Wednesday\n");

        printf(" Enter 4 for Thursday\n");

        printf(" Enter 5 for Friday\n");

        printf(" Enter 6 for Saturday\n");

        printf(" Enter 7 for Sunday\n\n");

        /* Enter choice for day of the week */
        printf(" Enter your choice: ");
```

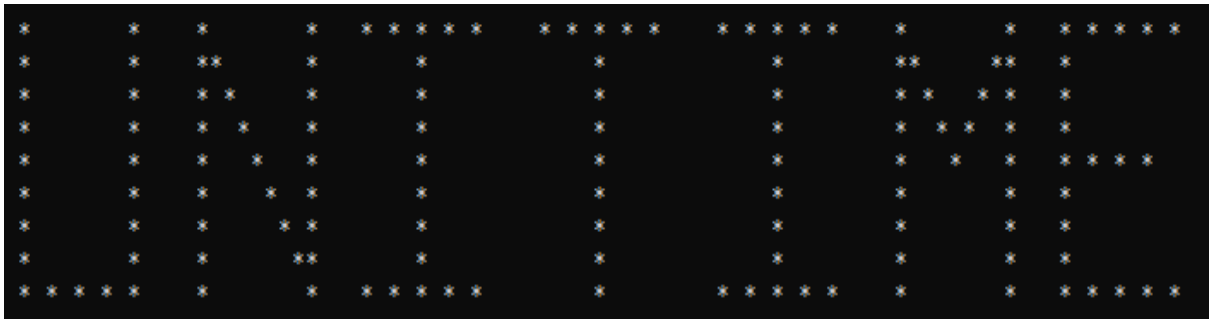
```
        scanf("%d",&day);

        printf("\n Enter hours in 24 hour format: ");

        scanf("%d",&hours);

        printf(" Enter minutes: ");

        scanf("%d",&minutes);

        printf("\n Time is %d:%d\n\n",hours,minutes);

        /* Printing day and the time table as per the value of variables
'day' & 'hours' */

        switch(day)
        {

            case 1: printf(" Day is: Monday\n\n");

            /* Calling function monday() for printing time table */
            monday(hours);
            break;

            case 2: printf(" Day is: Tuesday\n\n");

             /* Calling function tuesday() for printing time table */
            tuesday(hours);
            break;

            case 3: printf(" Day is: Wednesday\n\n");

             /* Calling function wednesday() for printing time table */
            wednesday(hours);
            break;

            case 4: printf(" Day is: Thursday\n\n");

             /* Calling function thursday() for printing time table */
            thursday(hours);
            break;
```

```
            case 5: printf(" Day is: Friday\n\n");

             /* Calling function friday() for printing time table */
            friday(hours);
            break;

            case 6: printf(" Day is: Saturday\n\n");

             /* Calling function saturday() for printing time table */
            saturday(hours);
            break;

            case 7: printf(" Day is: Sunday\n\n");

            printf(" No Class!\n\n");
            break;

        }
    }

    else
    {
        printf("\n Wrong Input!\n\n");
    }

    printf(" Program Executed Successfully!\n\n");

    printf(" ------------------------------\n");

    printf(" | Thanks For Using UNITIME ! |\n");

    printf(" ------------------------------\n");

    return 0;
}
```

```
------------------------------
| Batch-B4 Time Table Decoder |
------------------------------

Options Available:

1. Find time table as per current day, date and time

2. Find time table by entering day and time manually

Enter your choice:
```



```
------------------------------
| Batch-B4 Time Table Decoder |
------------------------------

Options Available:

1. Find time table as per current day, date and time

2. Find time table by entering day and time manually

Enter your choice: 1

Date is: 17/12/2021

Time is: 14:42:57

Day is: Friday

No Class!
Program Executed Successfully!

------------------------------
| Thanks For Using UNITIME ! |
------------------------------

Process returned 0 (0x0)   execution time : 105.923 s
Press any key to continue.
```



```
------------------------------
| Batch-B4 Time Table Decoder |
------------------------------

Options Available:

1. Find time table as per current day, date and time

2. Find time table by entering day and time manually

Enter your choice: 2

Enter 1 for Monday
Enter 2 for Tuesday
Enter 3 for Wednesday
Enter 4 for Thursday
Enter 5 for Friday
Enter 6 for Saturday
Enter 7 for Sunday

Enter your choice: 5

Enter hours in 24 hour format: 15
Enter minutes: 15

Time is 15:15

Day is: Friday

Subject: SDF Lab-B

Teacher: Dr. Raghu Vamsi and Dr. K. Rajalakshmi

Location: CL-04 (Near LRC, Basement, ABB-3)

Program Executed Successfully!

------------------------------
| Thanks For Using UNITIME ! |
------------------------------

Process returned 0 (0x0)   execution time : 12.290 s
Press any key to continue.
```

REFERENCES

- **Lecture PPT’s**
- **Let us C by Yashavant Kanetkar**
- **Stackoverflow.com**