

Digital Clock

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

```
#include <time.h>
```

```
#include <unistd.h>
```

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

```
int hour, minute, second;
```

```
int month, year, date;
```

```
int main()
```

```
{
```

```
    printf("Set Your Date & Time:\n");
```

```
    printf("Enter Year: ");
```

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

```
    if (year < 0)
    {
        printf("Invalid year please ReEnter :");
        scanf("%d", &year);
    }
```

```
printf("Enter Month: ");
scanf("%d", &month);
if (month > 12 || month < 0)
{
    printf("Invalid month please ReEnter :");
    scanf("%d", &month);
}
```

```
printf("Enter Date: ");
scanf("%d", &date);
if (date > 31 || date < 0)
{
    printf("Invalid date please ReEnter :");
```

```
scanf("%d", &date);  
}
```

```
printf("Enter Hour: ");  
scanf("%d", &hour);  
if (hour > 24 || hour < 0)  
{  
    printf("Invalid hour please ReEnter :");  
    scanf("%d", &hour);  
}
```

```
printf("Enter Minute's: ");  
scanf("%d", &minute);  
if (minute > 60 || minute < 0)  
{  
    printf("Invalid minute please ReEnter :");  
    scanf("%d", &minute);  
}
```

```
printf("Enter Second: ");
```

```
scanf("%d", &second);  
if (second > 60 || second < 0)  
{  
    printf("Invalid second please ReEnter :");  
    scanf("%d", &second);  
}
```

```
while (1)  
{  
    if (hour < 12)  
    {  
        printf("\t\t\t\t\tDATE MONTH YEAR\n",  
hour, minute, second);  
        printf("\t\t\t\t\t%02d : %02d : %02d \n",  
date, month, year);  
        printf("\t\t\t\t\t%02d:%02d:%02d AM \n",  
hour, minute, second);  
    }  
    else  
    {
```

```
    printf("\t\t\t\tDATE MONTH YEAR\n",
hour, minute, second);

    printf("\t\t\t\t%02d : %02d : %02d \n",
date, month, year);

    printf("\t\t\t\t%02d:%02d:%02d PM\n",
hour, minute, second);
}

sleep(1);
second++;
if (second == 60)
{
    minute++;
    second = 0;
}
if (minute == 60)
{
    hour++;
    minute = 0;
}
if (hour == 24)
```

```
{  
  
    hour = 0;  
  
}  
  
system("cls");  
  
}  
  
return 0;  
}
```



Code

main

Go to file

DigitalClock.c

DigitalClock.exe

README.md

Time-Clock-Project / README.md

Preview Code Blame 19 lines (12 loc) · 629 Bytes

Raw Copy Download Edit Menu

Description

This program allows you to set the date and time, and then displays the current date and time in a continuous loop. The time is displayed in a 12-hour format with AM or PM indication.

Usage

1. Compile the program using a C compiler.
2. Run the compiled program.
3. Follow the prompts to set the date and time.
4. The program will display the current date and time continuously, updating every second.

Contributing

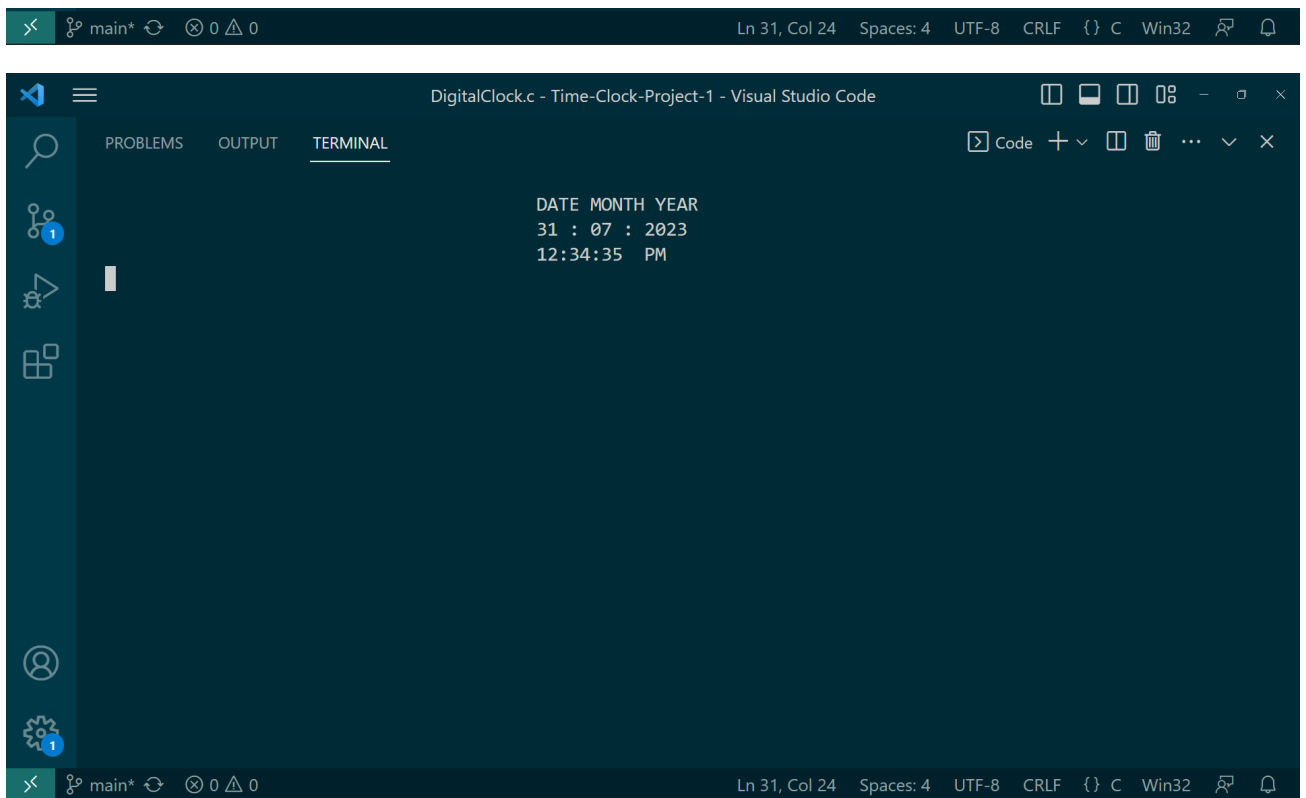
Contributions are welcome! If you have any suggestions or improvements, feel free to open an issue or submit a pull request.

Documentation • Share feedback

OUTPUT

```
PS C:\Users\OM\OneDrive - iiit-bh.ac.in\Notebooks\Desktop\Digital Clock\Time-Clock-Project-1> cd "c:\Users\OM\OneDrive - iiit-bh.ac.in\Notebooks\Desktop\Digital Clock\Time-Clock-Project-1\" ; if ($?) { gcc DigitalClock.c -o DigitalClock } ; if ($?) { .\DigitalClock }
Set Your Date & Time:
Enter Year: 2023
Enter Month: 7
Enter Date: 31
Enter Hour: 12
Enter Minute's: 34
Enter Second: 35
```

```
DATE MONTH YEAR
31 : 07 : 2023
12:34:31 PM
```



```
DATE MONTH YEAR
31 : 07 : 2023
12:34:35 PM
```



The code includes necessary header files:

`<stdio.h>` for input/output functions, `<time.h>` for time-related functions, `<unistd.h>` for sleep function, and `<stdlib.h>` for system function.

Global variables are declared to store the user-entered values for year, month, date, hour, minute, and second.

The main function starts the execution of the program. It prompts the user to set the date and time:

a. The user is asked to input the year, and if the value is negative, it prompts again for a valid year.

- b. The user is asked to input the month, and if the value is not within the valid range (1 to 12), it prompts again for a valid month.**
- c. The user is asked to input the date, and if the value is not within the valid range (1 to 31), it prompts again for a valid date.**
- d. The user is asked to input the hour, and if the value is not within the valid range (0 to 23), it prompts again for a valid hour.**
- e. The user is asked to input the minute, and if the value is not within the valid range (0 to 59), it prompts again for a valid minute.**
- f. The user is asked to input the second, and if the value is not within the valid range (0 to 59), it prompts again for a valid second.**

After successfully setting the date and time, the program enters an infinite loop (while(1)). Within this loop, the clock is continuously updated and displayed.

The clock display shows the date, month, and year, followed by the time in the format HH:MM:SS AM/PM.

The sleep(1) function is used to pause the execution for one second, creating a one-second delay in the loop.

The variables second, minute, and hour are updated to reflect the new time after each second. If the seconds reach 60, the minutes are incremented, and seconds are reset to 0. The same happens for minutes and hours, respectively.

When the hours reach 24, they are reset to 0, representing a new day.

The system("cls") function is used to clear the screen before displaying the updated time. Note that this line of code assumes a Windows environment; on other platforms, you may need to use a different command to clear the screen.

The program keeps running indefinitely, displaying the updated clock in real-time until it is terminated manually.





DIGITAL CLOCK

By :

REETAMKUMAR NAYAK (B122090)

SASWAT PARASAR BEHERA (B122103)

DIVYAKUMAR PATEL (B122079)