N.B.K.R. Institute of Science and Technology

Vidyanagar-Tirupati District

Title: SIMPLE ALARM CLOCK (CONSOLE-BASED)

Course: Data Structure

Branch: Computer science

Section: D

Year: I

Semester: II

Submitted to: Ashok Selva Kumar

Submitted by:

* Harsha vardhan – 24KB1A05CX
* Deekshith Reddy – 24KB1A05EN
* Mukesh - 24KB1A05CS

**1. ACKNOWLEDGMENT**

I would like to express my sincere gratitude to our faculty members and the Department of Computer Science and Engineering for giving me the opportunity to work on this mini project titled “Simple Alarm Clock”. I am especially thankful to my project guide and instructors for their continuous support and valuable guidance throughout the development of this project.

**2. OBJECTIVES**

- To implement a simple alarm clock system using the C programming language.

- To demonstrate the use of control structures, functions, and time-based operations using time.h.

- To apply Data Structures and basic programming logic in a real-world application.

**3. SYSTEM REQUIREMENTS**

- Hardware Requirements:

- Processor: Intel Core i3 or higher

- RAM: Minimum 4 GB

- Storage: Minimum 100 MB free space

- Software Requirements:

- Operating System: Windows/Linux

- Compiler: GCC (for C)

- IDE: Code::Blocks / Dev C++ / VS Code

**4. METHODOLOGY**

- The user is prompted to set an alarm time (hour, minute, second).

- The system continuously checks the current system time using the time.h library.

- A loop runs until the alarm time matches the system time.

- When the set time is reached, the alarm message is printed on the console.

**5. PROJECT DESCRIPTION**

This project is a console-based application developed in C that allows users to set an alarm time. The application uses the standard C library time.h to access and compare system time. It demonstrates the use of loops, conditional statements, and standard input/output functions in a real-time scenario. The project is simple yet effective in demonstrating basic concepts in C programming and time-based event triggering.

**6. ALGORITHM**

1. Start

2. Ask the user to input alarm time (hours, minutes, seconds)

3. Store the input

4. Use a loop to continuously check the current system time

5. Compare the current time with the set alarm time

6. If current time equals alarm time, display "Alarm! Time reached."

7. Exit

**7. PROGRAM CODE**

#include <stdio.h>

#include <time.h>

#include <stdlib.h>

int main() {

int hour, minute, second;

printf("Set Alarm Time (HH MM SS): ");

scanf("%d %d %d", &hour, &minute, &second);

printf("Alarm is set for %02d:%02d:%02d\n", hour, minute, second);

while (1) {

time\_t now = time(NULL);

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

if (current->tm\_hour == hour &&

current->tm\_min == minute &&

current->tm\_sec == second) {

printf("⏰ Alarm! Time reached: %02d:%02d:%02d\n", hour, minute, second);

break;

}

}

return 0;

}

**8. TESTING**

| Test Case | Input Time (HH:MM:SS) | Expected Output | Result |

|-----------|-----------------------------------|------------------------------------|-----------------|

| TC1 | 15:30:00 | Alarm triggers at 15:30:00 | Passed |

| TC2 | Invalid input (25:00:00) | Error or incorrect behavior | Not handled |

| TC3 | 00:00:00 | Alarm triggers at midnight | Passed |

**9. LIMITATIONS**

- Works only when the program is running; it does not run in the background.

- No sound or advanced notification—only a text-based alert.

- No error handling for invalid time formats.

**10. CONCLUSION**

This mini project helped in understanding how real-time applications like an alarm clock can be implemented using C programming. It strengthened the concepts of loops, condition checking, and working with the system clock through the time.h library. The project is a simple yet effective demonstration of applying data structures and programming fundamentals.