Timetable Generator

Project submitted to the

SRM University – AP, Andhra Pradesh

for the partial fulfillment of the requirements to award the degree of

Bachelor of Technology/Master of Technology

In

Computer Science and Engineering

School of Engineering and Sciences

Submitted by

Group Name: Silent Sparks

Under the Guidance of

(Dr. Krishna Prasad)

SRM University–AP

Neerukonda, Mangalagiri, Guntur

Andhra Pradesh – 522 240

[Month, Year]

Team Members:

1) Lokesh Chirumamilla

2) Lohith Marneni

3) Sahithi Mukkamala

Introduction

The Timetable Generator project aims to automate the creation and display of class-wise timetables for a given set of subjects. The primary goal is to efficiently organize and manage the schedule for a week, considering different classes, days, and subjects.

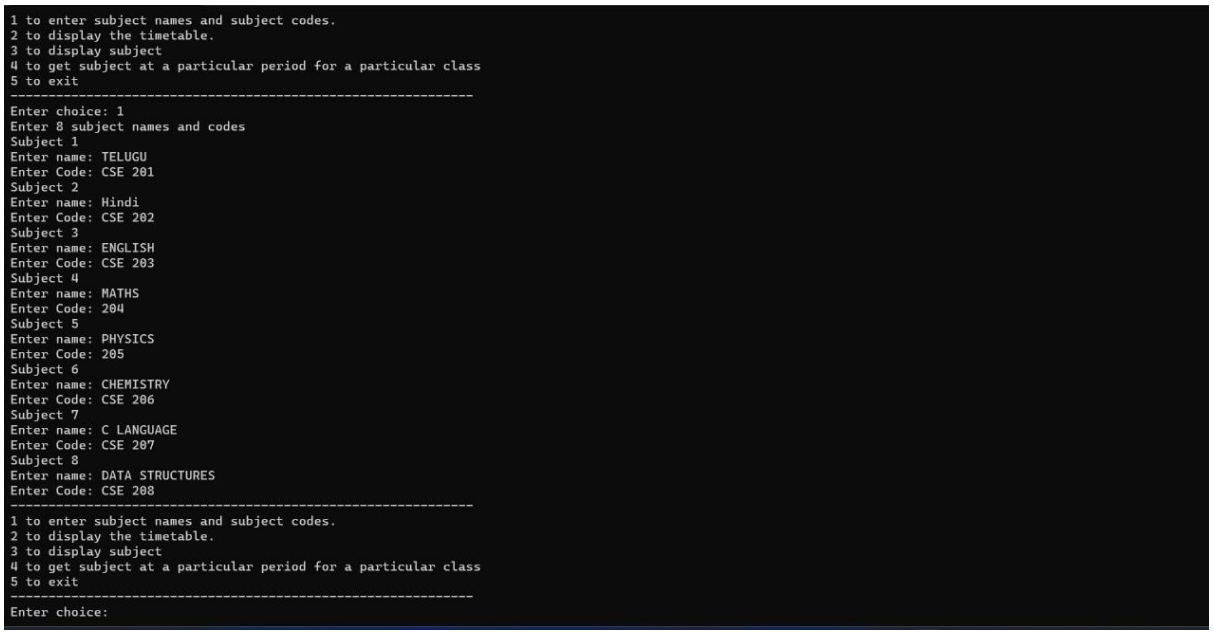
Problem Statement

Organizing a timetable for educational institutions can be a complex task. It involves assigning specific subjects to different classes and time periods, ensuring a balanced and effective schedule. Manual creation of timetables can be time-consuming and error-prone. The project addresses this problem by providing an automated solution for timetable generation.

Steps to Solve the Problem

1. Input Subject Information

The project starts by taking input for eight subjects, including their names and codes. This step ensures that the system is aware of the subjects to be scheduled.



2. Timetable Initialization

The timetable is initialized with a default value of "No class" for each cell, representing the absence of a scheduled class.

```
void initialize()
{
  for (int i = 0; i < no_of_days; i++)
  {
    for (int j = 0; j < no_of_class; j++)
    {
      for (int k = 0; k < no_of_subject; k++)
      {
        timetable[i][j][k] = "No class";
      }
    }
  }
}
```

3. Index Calculation

To efficiently distribute subjects in the timetable, an index calculation function is used. It employs a hashing technique to determine the position of a subject in the timetable.

```
int index(int dayno, int classno, int subject)
{
  return (dayno * 13 + classno * 17 + subject * 19) % no_of_subject;
}
```

4. Timetable Creation

The actual timetable is created by assigning subjects to different classes, days, and time periods. The indexing mechanism ensures a

fair distribution of subjects.

```
void create_tt()
{
  int m;
  for (int i = 0; i < no_of_days; i++)
  {
    for (int j = 0; j < no_of_class; j++)
    {
      for (int k = 0; k < no_of_subject; k++)
      {
        m = index(i, j, k);
        timetable[i][j][k] = sub_codes[m];
      }
    }
  }
}
```

5. Display Timetables

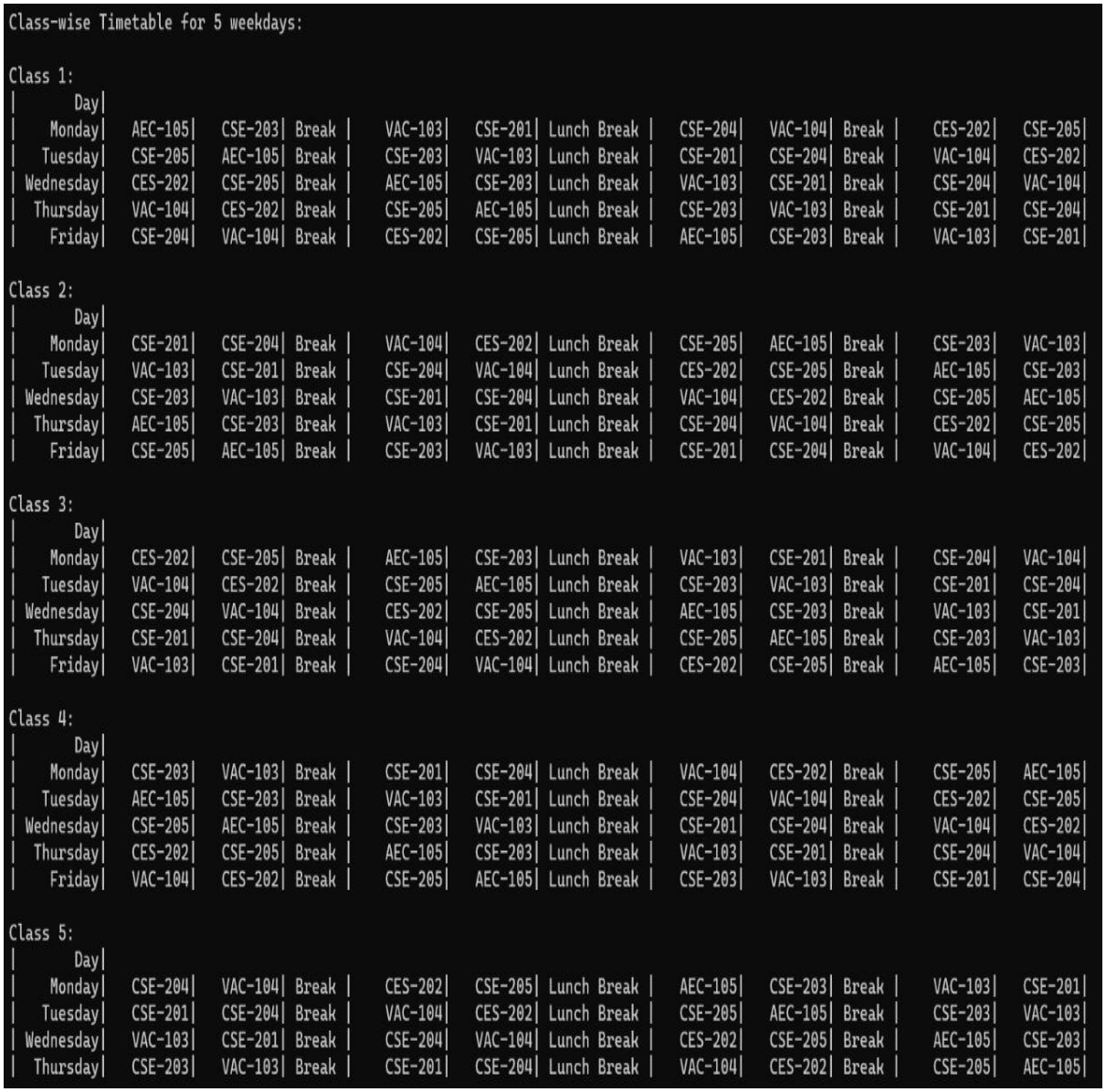
The system displays the generated timetables for each class and day, providing a clear visualization of the weekly schedule.

```
void display_timetables()
{
  const int column_width = 10;

  cout << "Class-wise Timetable for 5 weekdays:\n\n";

  for (int j = 0; j < no_of_class; j++)
  {
    cout << "Class " << (j + 1) << ":\n";
    cout << "|" << setw(column_width + 1) << " Day|";
    cout << "\n";
    for (int i = 0; i < no_of_days; i++)
    {
      cout << "|" << setw(column_width) << weekdays[i] << "|";
      for (int k = 0; k < no_of_subject; k++)
      {
        if (k == 4)
          cout << " Lunch Break |";
        if (k == 2 || k == 6)
          cout << " Break | ";
        cout << setw(column_width) << timetable[i][j][k] << "|";
      }
      cout << "\n";
    }

    cout << "\n";
  }
  return;
}
```



```
Class-wise Timetable for 5 weekdays:

Class 1:
|       Day|
|    Monday|   AEC-105|   CSE-203| Break |    VAC-103|   CSE-201| Lunch Break |   CSE-204|   VAC-104| Break |    CES-202|   CSE-205|
|   Tuesday|   CSE-205|   AEC-105| Break |    CSE-203|   VAC-103| Lunch Break |   CSE-201|   CSE-204| Break |    VAC-104|   CES-202|
| Wednesday|   CES-202|   CSE-205| Break |    AEC-105|   CSE-203| Lunch Break |   VAC-103|   CSE-201| Break |    CSE-204|   VAC-104|
|  Thursday|   VAC-104|   CES-202| Break |    CSE-205|   AEC-105| Lunch Break |   CSE-203|   VAC-103| Break |    CSE-201|   CSE-204|
|    Friday|   CSE-204|   VAC-104| Break |    CES-202|   CSE-205| Lunch Break |   AEC-105|   CSE-203| Break |    VAC-103|   CSE-201|

Class 2:
|       Day|
|    Monday|   CSE-201|   CSE-204| Break |    VAC-104|   CES-202| Lunch Break |   CSE-205|   AEC-105| Break |    CSE-203|   VAC-103|
|   Tuesday|   VAC-103|   CSE-201| Break |    CSE-204|   VAC-104| Lunch Break |   CES-202|   CSE-205| Break |    AEC-105|   CSE-203|
| Wednesday|   CSE-203|   VAC-103| Break |    CSE-201|   CSE-204| Lunch Break |   VAC-104|   CES-202| Break |    CSE-205|   AEC-105|
|  Thursday|   AEC-105|   CSE-203| Break |    VAC-103|   CSE-201| Lunch Break |   CSE-204|   VAC-104| Break |    CES-202|   CSE-205|
|    Friday|   CSE-205|   AEC-105| Break |    CSE-203|   VAC-103| Lunch Break |   CSE-201|   CSE-204| Break |    VAC-104|   CES-202|

Class 3:
|       Day|
|    Monday|   CES-202|   CSE-205| Break |    AEC-105|   CSE-203| Lunch Break |   VAC-103|   CSE-201| Break |    CSE-204|   VAC-104|
|   Tuesday|   VAC-104|   CES-202| Break |    CSE-205|   AEC-105| Lunch Break |   CSE-203|   VAC-103| Break |    CSE-201|   CSE-204|
| Wednesday|   CSE-204|   VAC-104| Break |    CES-202|   CSE-205| Lunch Break |   AEC-105|   CSE-203| Break |    VAC-103|   CSE-201|
|  Thursday|   CSE-201|   CSE-204| Break |    VAC-104|   CES-202| Lunch Break |   CSE-205|   AEC-105| Break |    CSE-203|   VAC-103|
|    Friday|   VAC-103|   CSE-201| Break |    CSE-204|   VAC-104| Lunch Break |   CES-202|   CSE-205| Break |    AEC-105|   CSE-203|

Class 4:
|       Day|
|    Monday|   CSE-203|   VAC-103| Break |    CSE-201|   CSE-204| Lunch Break |   VAC-104|   CES-202| Break |    CSE-205|   AEC-105|
|   Tuesday|   AEC-105|   CSE-203| Break |    VAC-103|   CSE-201| Lunch Break |   CSE-204|   VAC-104| Break |    CES-202|   CSE-205|
| Wednesday|   CSE-205|   AEC-105| Break |    CSE-203|   VAC-103| Lunch Break |   CSE-201|   CSE-204| Break |    VAC-104|   CES-202|
|  Thursday|   CES-202|   CSE-205| Break |    AEC-105|   CSE-203| Lunch Break |   VAC-103|   CSE-201| Break |    CSE-204|   VAC-104|
|    Friday|   VAC-104|   CES-202| Break |    CSE-205|   AEC-105| Lunch Break |   CSE-203|   VAC-103| Break |    CSE-201|   CSE-204|

Class 5:
|       Day|
|    Monday|   CSE-204|   VAC-104| Break |    CES-202|   CSE-205| Lunch Break |   AEC-105|   CSE-203| Break |    VAC-103|   CSE-201|
|   Tuesday|   CSE-201|   CSE-204| Break |    VAC-104|   CES-202| Lunch Break |   CSE-205|   AEC-105| Break |    CSE-203|   VAC-103|
| Wednesday|   VAC-103|   CSE-201| Break |    CSE-204|   VAC-104| Lunch Break |   CES-202|   CSE-205| Break |    AEC-105|   CSE-203|
|  Thursday|   CSE-203|   VAC-103| Break |    CSE-201|   CSE-204| Lunch Break |   VAC-104|   CES-202| Break |    CSE-205|   AEC-105|
```

6. Display Subjects

Users can view the list of subjects along with their corresponding codes, aiding in better understanding and reference.

```
------------------------------------------------------------
1 to enter subject names and subject codes.
2 to display the timetable.
3 to display subject
4 to get subject at a particular period for a particular class
5 to exit
------------------------------------------------------------
Enter choice: 3

SUBJECT CODES:
AEC-105-->Analytical Skills for Engineers
CSE-201-->Coding Skill-1
CES-202-->Oops with cpp
CSE-203-->Data structures-2
CSE-204-->Digital Electronics
CSE-205-->Python
VAC-103-->Co-circular activities
VAC-104-->comunity services and social responsibility
------------------------------------------------------------
```

7. Get Subject Information

The project allows users to retrieve information about a specific subject at a particular period for a given class.

```
1 to enter subject names and subject codes.
2 to display the timetable.
3 to display subject
4 to get subject at a particular period for a particular class
5 to exit
------------------------------------------------------------
Enter choice: 4
Enter class number (1-5): 2
Enter day number (1-5): 2
Enter period number (1-8): 2
Subject at Class 2, Day Tuesday, Period 2: CSE-201
------------------------------------------------------------
```

Data Structures Used

T In this code, the data structure used for maintaining the timetable information is a 3D array named timetable. The array is defined as follows:

```
string timetable[no_of_days][no_of_class]
[no_of_subject];
```

This 3D array is essentially a timetable where each element represents a slot in the schedule. The dimensions correspond to days, classes, and subjects. The array is initialized and manipulated throughout the code to store and retrieve subject codes for each class, day, and period. Additionally, the index function is used to calculate the index in the sub_codes array based on the day, class, and subject. This function employs a hashing-like approach to generate indices:

```
int index(int dayno, int classno, int
subject)
{
  return (dayno * 13 + classno * 17 +
subject * 19) % no_of_subject;
}
```

Here, the modulo operation is applied to ensure that the calculated index remains within the bounds of the sub_codes array. In summary, the primary data structure used in this code is a 3D array for managing the timetable, and the index function incorporates a simple hashing technique to determine the index for accessing subject codes.

Source Code Explanation

The source code is structured as a C++ program using object-oriented principles. It defines a timetable class with member functions for input, initialization, index calculation, timetable creation, and display. The main function serves as the entry point, providing a user interface for interacting with the timetable functionalities.

Input and Output Explanation

1. **Input**: The system takes user input for subject names and codes.

2. **Output**: The generated timetables are displayed class-wise, allowing users to visualize the weekly schedule. Subjects and their codes are also displayed, providing a comprehensive overview.

Conclusion

The Timetable Generator project offers a practical solution to the complex task of timetable creation. By automating the process, it enhances efficiency and reduces the likelihood of scheduling errors.

Future Enhancements

Potential future enhancements include adding features for manual adjustments, considering constraints such as teacher availability and classroom allocation.