



## TOPIC:-

University Timetable Management System

Course: Programming In C  
(Course Code:- CSEG1041\_5)

Submitted by  
Daksh Goyal  
(Sap\_Id:- 590023668)

Submitted To  
Mr. Mohsin  
Furkh Dar

Date of Submission:- 01/12/2025

## Problem Statement:-

In many educational institutions, creating a schedule by hand is a time-consuming and prone to error. It becomes difficult to effectively manage subject distribution, faculty availability, and student batches. The project's objective is to develop and implement a C program that arranges and methodically automates the faculty management, student record management, and schedule creation processes.

## Objective:-

- To develop a C-based system for automatic timetable generation.
- To manage student records including name, ID, date of birth, and batch.
- To manage faculty records with subject allocation.
- To generate a conflict-free timetable using randomization.
- To allow searching of student and faculty timetables.

## Software / Tools Used:-

Operating System:- Windows 11

Compiler:- GCC (MinGW) Compiler

# Algorithm of Project:-

1. **START**
2. **INPUT:** Load student data, faculty data, and timetable from files.
3. **INITIALIZE:** Start the random number generator using current time.
4. **DISPLAY:** Show the main menu with the following options:
  - Add Student
  - Add Faculty
  - Generate Timetable
  - Search Student
  - Search Faculty
  - Exit
5. **INPUT:** Read the user's choice.
6. **PROCESS:**
  - If choice = 1, accept and store student details.
  - If choice = 2, accept and store faculty details.
  - If choice = 3, generate the timetable automatically.
  - If choice = 4, search and display student timetable.
  - If choice = 5, search and display faculty timetable.
7. **REPEAT:** Return to the main menu until the user selects Exit.
8. **SAVE:** Store all updated student, faculty, and timetable data into files.
9. **OUTPUT:** Display program exit message.
10. **STOP**

# Pseudocode of Project:-

```
1  BEGIN TimetableSystem
2
3  INPUT current_time
4  SET random_seed = current_time
5
6  CALL LoadStudents
7  CALL LoadFaculty
8  CALL LoadTimetable
9
10 REPEAT
11     DISPLAY "1. Add Students"
12     DISPLAY "2. Add Faculty"
13     DISPLAY "3. Generate Timetable"
14     DISPLAY "4. Search Student"
15     DISPLAY "5. Search Faculty"
16     DISPLAY "6. Exit"
17
18     INPUT choice
19
20     IF choice == 1 THEN
21         CALL AddStudents
22     END IF
23
24     IF choice == 2 THEN
25         CALL AddFaculty
26     END IF
27
28     IF choice == 3 THEN
29         CALL GenerateTimetable
30     END IF
31
32     IF choice == 4 THEN
33         CALL SearchStudent
34     END IF
35
36     IF choice == 5 THEN
37         CALL SearchFaculty
38     END IF
39
40 UNTIL choice == 6
41
42 CALL SaveStudents
43 CALL SaveFaculty
44 CALL SaveTimetable
45
46 OUTPUT "Program Ended Successfully"
47
48 END TimetableSystem
```

# Output Of Program (Screenshots:-)

## 1:- Adding Student Details:-

```
PS D:\C Language\C PROJECT> gcc src/*.c -I include -o timetable_app
PS D:\C Language\C PROJECT> ./timetable_app.exe

=== Timetable System ===
1. Add Students
2. Add Faculty
3. Generate Timetable
4. Search Student
5. Search Faculty
6. Exit
Choose: 1
How many students? 3
Enter name: Ujjawal Chaurasia
DOB (DD-MM-YYYY): 23-09-2007

Enter name: Aarav Rana
DOB (DD-MM-YYYY): 15-06-2007

Enter name: Jaideep Singh
DOB (DD-MM-YYYY): 12-09-2007
```

## 2:- Adding Faculty Data:-

```
=== Timetable System ===
1. Add Students
2. Add Faculty
3. Generate Timetable
4. Search Student
5. Search Faculty
6. Exit
Choose: 2
How many faculty? 3
Enter name: Mr. Ashish Bhatnagar
Choose subject:

1. Advanced Engineering Mathematics
2. Programming in C
3. Linux Lab
4. Problem Solving
5. E.V.S

Enter Subject Number the Faculty teaches: 2

Enter name: Mr. Sunil Pujari
Choose subject:

1. Advanced Engineering Mathematics
2. Programming in C
3. Linux Lab
4. Problem Solving
5. E.V.S

Enter Subject Number the Faculty teaches: 3

Enter name: Ms. Damanpreet Kaur
Choose subject:

1. Advanced Engineering Mathematics
2. Programming in C
3. Linux Lab
4. Problem Solving
5. E.V.S

Enter Subject Number the Faculty teaches: 4
```

## 3:- Generating Timetable:-

```
=== Timetable System ===
1. Add Students
2. Add Faculty
3. Generate Timetable
4. Search Student
5. Search Faculty
6. Exit
Choose: 3
Timetable generated.
```

## 4:- Searching Student Timetable:-

```
=== Timetable System ===
1. Add Students
2. Add Faculty
3. Generate Timetable
4. Search Student
5. Search Faculty
6. Exit
Choose: 4
Enter student name: Aarav Rana

Student: Aarav Rana(Id: 1013), Batch 1

Monday:
14:00-15:00 Linux Lab (Mr. Sunil Pujari)
17:00-18:00 Problem Solving (Mr. Baldivya Mitra)

Tuesday:
10:00-11:00 Programming in C (Mr. Bhuvam Bam)
11:00-12:00 Linux Lab (Mr. Sunil Pujari)
14:00-15:00 Advanced Engineering Mathematics (Ms. Rishu Gandhi)
15:00-16:00 Advanced Engineering Mathematics (Ms. Rishu Gandhi)

Wednesday:
08:00-09:00 Programming in C (Mr. Ashish Bhatnagar)
10:00-11:00 Linux Lab (Dr. Supreet Singh)
16:00-17:00 Programming in C (Ms. Roshni Singh)
17:00-18:00 Programming in C (Mr. Mohsin Furkh Dar)

Thursday:
08:00-09:00 Programming in C (Mr. Ashish Bhatnagar)
11:00-12:00 Problem Solving (Ms. Damanpreet Kaur)
15:00-16:00 Problem Solving (Ms. Madhuri Mishra)
17:00-18:00 E.V.S (Mr. Vaibhav Saini)

Friday:
09:00-10:00 Linux Lab (Mr. Jitendra Yadav)
12:00-13:00 Linux Lab (Mr. Jitendra Yadav)
15:00-16:00 Programming in C (Mr. Bhuvam Bam)

Saturday:
08:00-09:00 E.V.S (Mr. Vaibhav Saini)
09:00-10:00 Problem Solving (3)
12:00-13:00 Linux Lab (Dr. Supreet Singh)
```

## 5:- Searching Faculty Timetable:-

```
=== Timetable System ===
1. Add Students
2. Add Faculty
3. Generate Timetable
4. Search Student
5. Search Faculty
6. Exit
Choose: 5
Enter faculty name: Mr. Sunil Pujari

Faculty: Mr. Sunil Pujari (Subject: Linux Lab)
Monday:
  08:00-09:00 Linux Lab (Batches: 13)
  12:00-13:00 Linux Lab (Batches: 5, 8)
  14:00-15:00 Linux Lab (Batches: 1)
  17:00-18:00 Linux Lab (Batches: 4, 20)

Tuesday:
  08:00-09:00 Linux Lab (Batches: 5)
  11:00-12:00 Linux Lab (Batches: 1, 13)
  15:00-16:00 Linux Lab (Batches: 9)
  16:00-17:00 Linux Lab (Batches: 8)

Wednesday:
  12:00-13:00 Linux Lab (Batches: 12)
  15:00-16:00 Linux Lab (Batches: 12)

Thursday:
  08:00-09:00 Linux Lab (Batches: 3)
  10:00-11:00 Linux Lab (Batches: 6)
  11:00-12:00 Linux Lab (Batches: 19)
  12:00-13:00 Linux Lab (Batches: 9)
  16:00-17:00 Linux Lab (Batches: 10)

Friday:
  10:00-11:00 Linux Lab (Batches: 8)
  12:00-13:00 Linux Lab (Batches: 21)
  13:00-14:00 Linux Lab (Batches: 19)
  16:00-17:00 Linux Lab (Batches: 13)

Saturday:
  09:00-10:00 Linux Lab (Batches: 6)
  15:00-16:00 Linux Lab (Batches: 15)
```

## 6:- Saving all changes and Exiting the program

```
=== Timetable System ===
1. Add Students
2. Add Faculty
3. Generate Timetable
4. Search Student
5. Search Faculty
6. Exit
Choose: 6
Saved and exiting.
PS D:\C Language\C PROJECT>
```

# Source Code (All .c files):-

## 1:- main.c

```
src > C main.c > _
1 // Including all standard library and user_defined library.
2
3 #include <stdio.h>
4 #include <stdlib.h>
5 #include <time.h>
6
7 #include "student.h"
8 #include "faculty.h"
9 #include "timetable.h"
10
11 int main()
12 {
13     srand((unsigned)time(NULL)); // Define in stdlib.h that makes your timetable different every time by randomizing subjects and slots.
14
15     load_students(); // Read data from student, faculty and timetable binary data.
16     load_faculty();
17     load_timetable();
18
19     int ch;
20     while (1)
21     {
22         printf("\n--- Timetable System ---\n");
23         printf("1. Add Student\n");
24         printf("2. Add Faculty\n");
25         printf("3. Generate Timetable\n");
26         printf("4. Search Student\n");
27         printf("5. Search Faculty\n");
28         printf("6. Exit\n");
29         printf("Choose : ");
30         scanf("%d", &ch);
31
32         if (ch == 1) add_students();
33         else if (ch == 2) add_faculty();
34         else if (ch == 3) generate_timetable();
35         else if (ch == 4) search_student();
36         else if (ch == 5) search_faculty();
37         else if (ch == 6)
38         {
39             save_students(); // Save all changes made in student, faculty and timetable binary file.
40             save_faculty();
41             save_timetable();
42             printf("Saved and exiting.\n");
43             break;
44         }
45         else
46         {
47             printf("Invalid Choice! Enter Again\n\n");
48         }
49     }
50     return 0;
51 }
```

## 2:- student.c

```
src > C student.c > _
1 #include <stdio.h>
2 #include <string.h>
3 #include "student.h"
4 #include "timetable.h"
5
6 Student students[MAX_STUDENTS];
7 int student_count = 0;
8 int next_student_id = 1000;
9
10 static void flush_stdin() { // It clears leftover input from the keyboard buffer so the next input works correctly.
11     int c;
12     while ((c = getchar()) != '\n' && c != EOF) {}
13 }
14
15 static int ci_equal(const char *a, const char *b) { // Compare strings and convert uppercase to lowercase.
16     while (*a && *b)
17     {
18         char x = *a, y = *b;
19         if (x >= 'A' && x <= 'Z') x += 32;
20         if (y >= 'A' && y <= 'Z') y += 32;
21         if (x != y) return 0;
22         a++; b++;
23     }
24     return *a == 0 && *b == 0;
25 }
26
27 void save_students()
28 {
29     FILE *f = fopen("students.dat", "wb");
30     if (!f) return;
31     fwrite(&student_count, sizeof(int), 1, f);
32     fwrite(&next_student_id, sizeof(int), 1, f);
33     fwrite(students, sizeof(Student), student_count, f);
34     fclose(f);
35 }
36
37 void load_students()
38 {
39     FILE *f = fopen("students.dat", "rb");
40     if (!f) return;
41     fread(&student_count, sizeof(int), 1, f);
42     fread(&next_student_id, sizeof(int), 1, f);
43     fread(students, sizeof(Student), student_count, f);
44     fclose(f);
45 }
```

```

src > C student.c > ...
43
44 void add_students(){
45     int n;
46     printf("How many students? ");
47     if (scanf("%d", &n) != 1){
48         flush_stdin();
49         return;
50     }
51     flush_stdin();
52
53     for (int i=0; i<n; i++){
54         if (student_count >= MAX_STUDENTS){
55             printf("Limit reached.\n");
56             break;
57         }
58         Student S;
59         S.student_id = next_student_id++;
60
61         printf("Enter name: ");
62         fgets(S.name, MAX_NAME, stdin);
63         S.name[strcspn(S.name, "\n")] = 0; // strcspn() removes extra \n for clear result.
64
65         printf("DOB (DD-MM-YYYY): ");
66         scanf("%d-%d-%d", &S.dob.day, &S.dob.month, &S.dob.year);
67         flush_stdin();
68
69         S.batch = (student_count / MAX_STUDENTS_PER_BATCH) + 1;
70         students[student_count++] = S;
71
72         printf("\n");
73     }
74
75     save_students();
76 }
77
78 void search_student(){
79     char student_name[MAX_NAME];
80     printf("Enter student name: ");
81     flush_stdin();
82     fgets(student_name, MAX_NAME, stdin);
83     student_name[strcspn(student_name, "\n")] = 0;
84
85     for (int i=0; i<student_count; i++){
86         if (strcmp(student_name, students[i].name) == 0){
87             printf("Student: %s (Id: %d), Batch %d\n", students[i].name, students[i].student_id, students[i].batch);
88             printf("\n");
89             print_batch_timetable(students[i].batch);
90             return;
91         }
92     }
93
94     printf("Not found.\n");
95 }

```

### 3:- faculty.c

```

src > C faculty.c > add_faculty()
1 #include <stdio.h>
2 #include <string.h>
3 #include <stdlib.h>
4 #include "faculty.h"
5 #include "timetable.h"
6
7 Faculty fac[MAX_FACULTY];
8 int faculty_count = 0;
9 int next_faculty_id = 1;
10
11 static void flush_stdin() { // It clears leftover input from the keyboard buffer so the next input works correctly.
12     int c;
13     while ((c = getchar()) != '\n' && c != EOF) {}
14 }
15
16 static int randint(int a, int b){
17     return a + rand() % (b - a + 1);
18 }
19
20 static int ci_equal(const char *a, const char *b) { // // Compare strings and convert uppercase to lowercase.
21     while (*a && *b){
22         char x = *a, y = *b;
23         if (x >= 'A' && x <= 'Z') x += 32;
24         if (y >= 'A' && y <= 'Z') y += 32;
25         if (x != y) return 0;
26         a++; b++;
27     }
28     return *a == 0 && *b == 0;
29 }
30
31 void save_faculty(){
32     FILE *f = fopen("faculty0.dat", "wb");
33     if (!f) return;
34     fwrite(&faculty_count, sizeof(int), 1, f);
35     fwrite(&next_faculty_id, sizeof(int), 1, f);
36     fwrite(fac, sizeof(Faculty), faculty_count, f);
37     fclose(f);
38 }
39
40 void load_faculty(){
41     FILE *f = fopen("faculty0.dat", "rb");
42     if (!f) return;
43     fread(&faculty_count, sizeof(int), 1, f);
44     fread(&next_faculty_id, sizeof(int), 1, f);
45     fread(fac, sizeof(Faculty), faculty_count, f);
46     fclose(f);
47 }
48

```



src > C faculty.c > add\_faculty()

```
48
49 void add_faculty(){
50     int n;
51     printf("How many faculty? ");
52     scanf("%d", &n);
53     flush_stdin();
54
55     for (int i=0;i<n;i++){
56         if (faculty_count >= MAX_FACULTY){
57             printf("Limit reached.\n");
58             break;
59         }
60
61         Faculty F;
62         F.faculty_id = next_faculty_id++;
63
64         printf("Enter name: ");
65         fgets(F.name, MAX_NAME, stdin);
66         F.name[strcspn(F.name, "\n")] = 0;
67
68         printf("Choose subject:\n\n");
69         for (int s=0;s<SUBJECT_COUNT;s++){
70             printf("%d. %s\n", s+1, subjects[s]);
71         }
72         printf("\n");
73         int c;
74         printf("Enter Subject Number the Faculty teaches: ");
75         scanf("%d", &c);
76
77         if (c > SUBJECT_COUNT) {
78             printf("Invalid Option. Please Select Again\n");
79
80             printf("Enter Subject Number the Faculty teaches: ");
81             scanf("%d", &c);
82         }
83         flush_stdin();
84         F.subject_id = c - 1;
85
86         fac[faculty_count++] = F;
87
88         printf("\n");
89     }
90
91     save_faculty();
92 }
93
```

src > C faculty.c > ...

```
93
94 int random_faculty_for_subject(int subj){
95     int list[MAX_FACULTY], count = 0;
96     for (int i=0;i<faculty_count;i++){
97         if (fac[i].subject_id == subj)
98             list[count++] = fac[i].faculty_id;
99     }
100     if (count == 0) return 0;
101
102     return list[rand() % count];
103 }
104
105 void search_faculty(){
106     char faculty_name[MAX_NAME];
107     printf("Enter faculty name: ");
108     flush_stdin();
109     fgets(faculty_name, MAX_NAME, stdin);
110     faculty_name[strcspn(faculty_name, "\n")] = 0;
111
112     printf("\n");
113
114     for (int i=0;i<faculty_count;i++){
115         if (ci_equal(faculty_name, fac[i].name)){
116             print_faculty_timetable_grouped(fac[i].faculty_id);
117             return;
118         }
119     }
120     printf("Not found.\n");
121 }
```

## 4:- timetable.c

```
src > C timetable.c > ...
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  #include <time.h>
5
6  #include "timetable.h"
7  #include "faculty.h"
8  #include "student.h"
9  #include "common.h"
10
11  Cell timetable[MAX_BATCHES][DAYS][SLOTS];
12
13  /* Subject names */
14  const char *subjects[SUBJECT_COUNT] = {
15      | "Advanced Engineering Mathematics", "Programming in C", "Linux Lab", "Problem Solving", "E.V.S"
16  };
17
18  /* Day names */
19  const char *daynames[DAYS] = {
20      | "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"
21  };
22
23  int randint(int a, int b){
24      | return a + rand() % (b - a + 1);
25  }
26
27  /* Count how many batches a faculty teaches at the same time */
28  int faculty_slot(int faculty_id, int day, int slot){
29      | int count = 0;
30
31      | for (int b = 0; b < MAX_BATCHES; b++){
32      |     | if (timetable[b][day][slot].faculty_id == faculty_id){
33      |         | count++;
34      |     | }
35      | }
36      | return count; /* Maximum allowed = 2 */
37  }
38
39  void save_timetable(){
40      | FILE *f = fopen("timetable.dat", "wb");
41      | if (!f) return;
42      | fwrite(timetable, sizeof(Cell), MAX_BATCHES * DAYS * SLOTS, f);
43      | fclose(f);
44  }
45
46  void load_timetable(){
47      | FILE *f = fopen("timetable.dat", "rb");
48      | if (!f){
49      |     /* Initialize empty timetable */
50      |     | for (int b=0; b<MAX_BATCHES; b++){
51      |         | for (int d=0; d<DAYS; d++){
52      |             | for (int s=0; s<SLOTS; s++){
53      |                 | timetable[b][d][s].subject_id = -1;
54      |                 | timetable[b][d][s].faculty_id = 0;
55      |                 | timetable[b][d][s].hour = 0 + s;
56      |             | }
57      |         | }
58      |     | }
59      | }
60
61      | fread(timetable, sizeof(Cell), MAX_BATCHES * DAYS * SLOTS, f);
62      | fclose(f);
63  }
64
```

```
src > C timetable.c > ...
64
65  void generate_timetable(){
66
67      /* Clear timetable first */
68      | for (int b=0; b<MAX_BATCHES; b++){
69      |     | for (int d=0; d<DAYS; d++){
70      |         | for (int s=0; s<SLOTS; s++){
71      |             | timetable[b][d][s].subject_id = -1;
72      |             | timetable[b][d][s].faculty_id = 0;
73      |             | timetable[b][d][s].hour = 0 + s;
74      |         | }
75      |     | }
76      | }
77
78      /* generate timetable */
79      | for (int b=0; b<MAX_BATCHES; b++){
80      |     | for (int d=0; d<DAYS; d++){
81      |         | int periods = randint(MIN_PERIODS, MAX_PERIODS);
82      |         | int used[SLOTS] = {0};
83      |         | int done = 0;
84
85      |         | while (done < periods){
86      |             | int slot = randint(0, SLOTS-1);
87      |             | if (used[slot] continue;
88      |             | used[slot] = 1;
89
90      |             | int subj = randint(0, SUBJECT_COUNT-1);
91      |             | int assigned_faculty = 0;
92
93      |             | /* Try to find a faculty who is free (max 2 loads allowed) */
94      |             | for (int try = 0; try < 20; try++){
95      |                 | int fid = random_faculty_for_subject(subj);
96      |                 | if (fid == 0) break;
97
98      |                 | if (faculty_slot(fid, d, slot) < 2){
99      |                     | assigned_faculty = fid;
100      |                     | break;
101      |                 | }
102
103      |                 | timetable[b][d][slot].subject_id = subj;
104      |                 | timetable[b][d][slot].faculty_id = assigned_faculty;
105
106      |                 | done++;
107      |             | }
108      |         | }
109      |     | }
110
111      | save_timetable();
112      | printf("Timetable generated.\n");
113  }

```

```

src > C timetable.c > ...
115 void print_batch_timetable(int batch){
116     if (batch < 1 || batch > MAX_BATCHES){
117         printf("Invalid batch.\n");
118         return;
119     }
120
121     int b = batch - 1;
122
123     for (int d=0; d<DAYS; d++){
124         printf("%s:\n", daynames[d]);
125
126         int any = 0;
127
128         for (int s=0; s<SLOTS; s++){
129             if (timetable[b][d][s].subject_id != -1){
130                 any = 1;
131
132                 int hour = timetable[b][d][s].hour;
133                 int subj = timetable[b][d][s].subject_id;
134                 int fid = timetable[b][d][s].faculty_id;
135
136                 char fname[MAX_NAME] = "TBA";
137
138                 if (fid != 0){
139                     for (int i=0; i<faculty_count; i++){
140                         if (fac[i].faculty_id == fid){
141                             strcpy(fname, fac[i].name);
142                             break;
143                         }
144                     }
145                 }
146
147                 printf(" %02d:00-%02d:00 %s (%s)\n",
148                     hour, hour+1, subjects[subj], fname);
149             }
150         }
151         printf("\n");
152
153         if (!any) printf(" No classes\n");
154     }
155 }
156

```

```

src > C timetable.c > ...
156
157 void print_faculty_timetable_grouped(int faculty_id){
158
159     int fidx = -1;
160     for (int i=0; i<faculty_count; i++){
161         if (fac[i].faculty_id == faculty_id){
162             fidx = i;
163             break;
164         }
165     }
166
167     if (fidx == -1){
168         printf("Faculty not found.\n");
169         return;
170     }
171
172     printf("Faculty: %s (Subject: %s)\n",
173         fac[fidx].name,
174         subjects[fac[fidx].subject_id]);
175
176     for (int d=0; d<DAYS; d++){
177         printf("%s:\n", daynames[d]);
178
179         int any = 0;
180
181         for (int s=0; s<SLOTS; s++){
182             int batches[3];
183             int count = 0;
184
185             for (int b=0; b<MAX_BATCHES; b++){
186                 if (timetable[b][d][s].faculty_id == faculty_id){
187                     if (count < 2){ /* show max 2 only */
188                         batches[count] = b + 1;
189                     }
190                     count++;
191                 }
192             }
193
194             if (count > 0){
195                 any = 1;
196                 int hour = 8 + s;
197
198                 printf(" %02d:00-%02d:00 %s (Batches: ",
199                     hour, hour+1, subjects[fac[fidx].subject_id]);
200
201                 int show = count;
202                 if (show > 2) show = 2; /* print max 2 */
203
204                 for (int k=0; k<show; k++){
205                     printf("%d", batches[k]);
206                     if (k+1 < show) printf(", ");
207                 }
208
209                 printf(")\n");
210             }
211         }
212         printf("\n");
213
214         if (!any) printf(" No classes\n");
215     }
216 }

```

## Conclusion:-

The Timetable Management System was successfully designed and implemented using the C programming language. The program efficiently manages student and faculty records and generates an automatic schedule with suitable subject and faculty allocation. This project helped me understand file handling, structures, arrays, and modular programming in C. The system successfully accomplishes the project's original objectives.

## Future Enhancements:-

- Adding a option to search timetable by entering batch number.
- Adding a option to show list of students in a batch number entered by user.
- Adding a option to show list of faculties by the subject they teach (subject was entered by user).