FACULTY OF ENGINEERING

جامعة مصر للمعلوماتية
EGYPT UNIVERSITY
OF INFORMATICS

Prepared By:

Malak Sharaf	23-101169	(Team Leader)
Sondos Aly	23-101189	
Nour ElBakry	23-101084	
Farah Ezzat	23-101224	
Areej Taher	23-101030	

FACULTY OF
ENGINEERING

جامعة مصر للمعلوماتية
EGYPT UNIVERSITY
OF INFORMATICS

Under Supervision of:

Dr. Hala Zayed

Eng Rameez Barakat

Table of Contents

Project Overview

The program receives input from the user about different courses and their dates and student IDs with their course codes then it sorts all the data into arrays which is used when the user asks about any function which tell the number of students, halls or the student which attend in specific hall or date and many other function which will be described in details.

Implementation

Functions

This section should include the functions in your project with a description of each function's objectives, parameters and expected output.

Input

This is a **sample of the input** we take:

CoursesExams=[101,28/4/2016,A;201,3/5/2016,A;110,5/5/2016,C;103,5/5/2016,A;120,6/5/2016,D;132,7/5/2016,B]
Students=[191001,201,101,110;191002,201,110,103;191003,110,120,132;191006,103,132;191000,102,105]

We got this input in 2-char arrays, then we sperate the input in 3 arrays:

Students: a 2D integer array contains students IDs and courses they attend.

courses_exam: a 2D integer array contains courses, day, month and year.

Halls: a 1D char array contains the halls of the courses with the same index as the courses.

Firstly:

We had 3 for nested loops for each array

One to shift rows **/** second to shift columns **/** third to shift between characters

```
for(int x = 0 ; check2 [i-1] != ']' ; x++){                    // the outer loop to shift rows
        for(int y =0 ; ; y++){                                  // the middle loop to shift columns
           for(; check2 [i] != ',' && check2 [i] != ';' ; i++)
```

If the value of the array is **‘,’** it will shift column but if it was **‘;’** it will shift row to the next course in the first array and to the next student in the second one.

To change from char to int we subtract the ASCII code of the zero ‘0’ from the array value at that index to change to int and we use int (count) to change to a whole number through using:

Int count = 0;

count = count * 10 + (check[n] - '0');

For example, If we want to read 102, but we read 1 0 2 separately. First, we changed to to int by subtracting ‘0’ then we multiply count by 10. So,

At first try count = (0*10) + 1=1

Second one: count = (1*10) +0= 10

Third one: count= (10*10) +2 = 102

Then count got initialized in each iteration to get another value. By this way we can change any char to integers

We made specific condition to but the halls in different array:

if(check[n] >= 'A' && check[n] <= 'Z')

Check is the array and **n** is the int for the index. We made condition if the value of the array (ASCII code) is between A and Z to be put in different char array but with the same index.

The condition of the loop

So, at the end we changed the 2-char array to 3 different arrays.

We made another array (command) to be the array scanning for the desired function from the user in an infinite loop where it does not stop until calling function Quit then it returns zero and ends the loop and the main function.

Function 1:

The 1st function counts the number of students. This function requires an array (the Students array) as a parameter. By using the fact that when you initialize an array with elements less than its size, the rest are initialized to 0. So, I created a loop that increments a variable with each row in the Students array until an element is equal to 0. Then the function prints the variable and breaks the loop.

Function 2:

The 2nd function counts the number of halls without counting the same hall more than once if it appears more than once in the Halls array. This function requires an array (the Halls array) as a parameter. I first copy the Halls array into another array that I called Halls_2 so the original array won’t be affected when I sort the halls. I then sort the Halls_2 array alphabetically. I create a variable and call it sum, I iterate on all elements of the array and only increment sum when the value of the element (ASCII code of the character) is not equal to the one after it. When the loop reads the delimiter of the function, it stops. Then the function prints sum.

Function 3:

The 3rd Function (Student_ID_Min) takes the student array (with the size [50] [6]) as a parameter. In the function, I initialize an integer called min_id to the first element in the array.

Then using a for loop, I pass by every Student ID and if it is less than the assigned number to min_id, the integer is redefined by the new ID.

After the loop has passed all IDs, it then prints the minimum ID.

Function 4:

The 4th Function (Students_Dropped_ID) also takes the student array (with the size [50] [6]) as a parameter. And for this function we have 3 variables 2 integers (min_id and max_id) both assigned to the first element in the array and a Boolean (a).

The function first uses the same loop used in Function 3 with the difference of also assigning the greatest ID to max_id

Then I use a nested for loop where the outer loop adds 1 to the min_id as long as it's less than the max_id

The inner loop then passes by all the IDs in the array to check if one of them equals the new variable. If the variable is found, the Boolean (a) changes to True and the whole loop breaks

However, if the Id is not in the array and the (a) is still false the ID will be printed, and the a will be reassigned as false to ensure that the loop continues correctly

Function 5:

-The required output from this function is to print out all days between the start and end day of students' exams including start and end days.

-the user will input Exams_Period_InDays and the output will be as we said before

-I put the variables "int max_month= INT_MIN "to make sure that any number I will put will be at the end greater than the initial value and all of them are the same concept, days_count are the days between the end, start days including them.

-1st 'for loop' to find the maximum date, first 'if' we compare years with each other, 'else if' if there are more than one year with the same date, we will go to the 2nd nested 'if' to compare months if the months also are the same them we will go to the last nested if in this loop to compare days and store this date in max_year, max_month, max_day.

-2nd 'for loop' is used to do the same as the first one but instead to calculate the minimum date.

-After that, I made an 'if loop' to calculate days if the start, and end years are the same year.

-Inside this 'if loop' there is 'nested for' to iterate months between the minimum and maximum months within the same year.

-Then 'else if' to know how to count days if the max_year > min_year.

-The 3rd ‘for loop’ is used to iterate months from min_month till December of the minimum year, because if we iterated again, it would make month 13 and we do not need that we need month 1.

-4th ‘for loop’ used to iterate months from month 1 to the max_month in the maximum year.

-The 5th for loop is used to iterate years between min_year and max_year, and the inner for loop is used to iterate months within each year when calculating days for whole years.

-After finishing the Exams_Period_InDays function I made a function called daysinmonth “its parameters are int month, int year” by a switch that will return for me the number of days in each month

-The last function I used is called leap year “Its parameter is int year” used to know if this year is leaping year or not.

Function 6:

- The function Course_Students requires the user to input a course code and the output will be the students who have taken this course.

- x is the course code that the user will input

-The program will iterate row by row to know if the student has taken this course, then print this student ID.

-That is all about this function.

Function 7:

This function iterates through each course in the CoursesExams array counts the number of students enrolled in each course by comparing course codes in the student array, and outputs the course codes with more than n students enrolled in.

The outer loop iterates over each row in the coursesExams array (representing each course)

The middle loop iterates over each row in of the student's array (representing each student)

The inner loop iterates through course code (Columns in the student array) for current student.

Enrolled students in each course are counted by checking if the current student is enrolled in the current course by comparing the course codes if yes it increments count of number of students enrolled in that course by one.

The output: check if the count of students enrolled in the current course is greater than n if yes it prints the course id.

Function 8:

Prints student IDs of the students attending more than n courses. It iterates over the row (representing the students) and checks the number of courses attended by each student. If the number of courses enrolled by the student is less than n it prints it out.

The outer loop Iterates through the row each row of the 2D array representing students

The inner loop iterates through each column skipping the first column which represents courses taken by each student.

The function counts the number of courses taken by each student by initializing a counter that keeps track of the number of courses taken by each student

Check if the value in the current cell is greater than 0 (meaning the student is attending the course) if yes it increments it by 1.

Check if the value of attended courses is less than n.

Function 9:

The 9th function lists the IDs of students in a certain hall at a certain date. This function requires 3 arrays (the CoursesExams array (2D containing the courses codes and their exam dates), the Students array (2D containing the students' IDs and the courses each takes), and the Hall array (1D char array that contains the halls names)) as a parameter The user calls for the function using the command List_Hall_Students, this command calls the function which then **1)** uses 2 scanf to take the Hall in the SHall array and the date in the Date array. **2)**The function then turns the Date elements into integers using 3 for loops, each one for each section of the date. The loop runs until the element is '/', which means the section is done, in each run the date updates to a new digit (Day = Day*10) and adds the new element to the least digit in an integer form (Date[i] - '0'), we subtract the '0' as ASCII code to get to the integer of the element. When the 3 loops are done, the function **3)** starts getting the code of the course which has the same exam date and hall name. This is done in a for loop that compares the hall name and the date given with the ones in the CoursesExams array, when the code that applies to these requirements is found, it is kept in a variable called SCode. **4)** Then in a nested for loop that iterates on the 2D array of students checking if they had that course, if they do **5)** the function prints the student's ID and it skips the rest od the courses using continue.

Function 10:

This function is called (List_Hall_Students_InAnyday) where we get from the user a hall name for example 'A' and we print all the IDs attending in this hall at any day without repeating.

First, we scan the desired hall then we search for all courses in this hall and put them in an integer array (temparray).

```
for(int i=0; i<20;i++){              // loop to check courses in this hall
    if(hallname[i] == hall){
```

```
                    temparray[num]=courseCode[i][0];   // put the desired courses in a one D
array

        num++;

         }}                              // end of the loop
```

After that we put 3 nested for loops to check in the students array if the the first course of the first student is any of these courses in the temparray. If this is true it prints the ID and break from the inner and middle loop to prevent repeating the same ID. If this is false it keeps checking the rest of IDs:

```
for (int i=0; Course_Students [i][0]! = 0; i++) {              // outer loop to shift rows

    for (int j = 1; j<6; j++){                            // middle loop to shift columns

        int a =0;                                  // initialize int to the index of the array

      for (;temparray[a]!= 0; a++){                    // inner loop to check if the course is one of any courses in the array

        if (Course_Students [i][j]==temparray[a] ){

          printf("%d\n",Course_Students[i][0]);

          break ;          } }   // break to prevent searching if the student has at least one of the courses

                    if (Course_Students [i][j]==temparray[a])      // break the middle loop and shift row to prevent
repeating the same ID

                { break; }

        }  }                                        // end of the outer lo
```

Function Quit:

The last function is quit, it ends the program. It takes no parameters and returns a variable (0) when called. Return Quit(); breaks the main function and ends the whole program.

Used Libraries/Built-in Functions

This section should include a description of any libraries that you used and any built-in methods (functions) that you use in your project.

#include <stdio.h>

Functions used: scanf, printf

#include <stdlib.h>

INT_MIN, INT_MAX

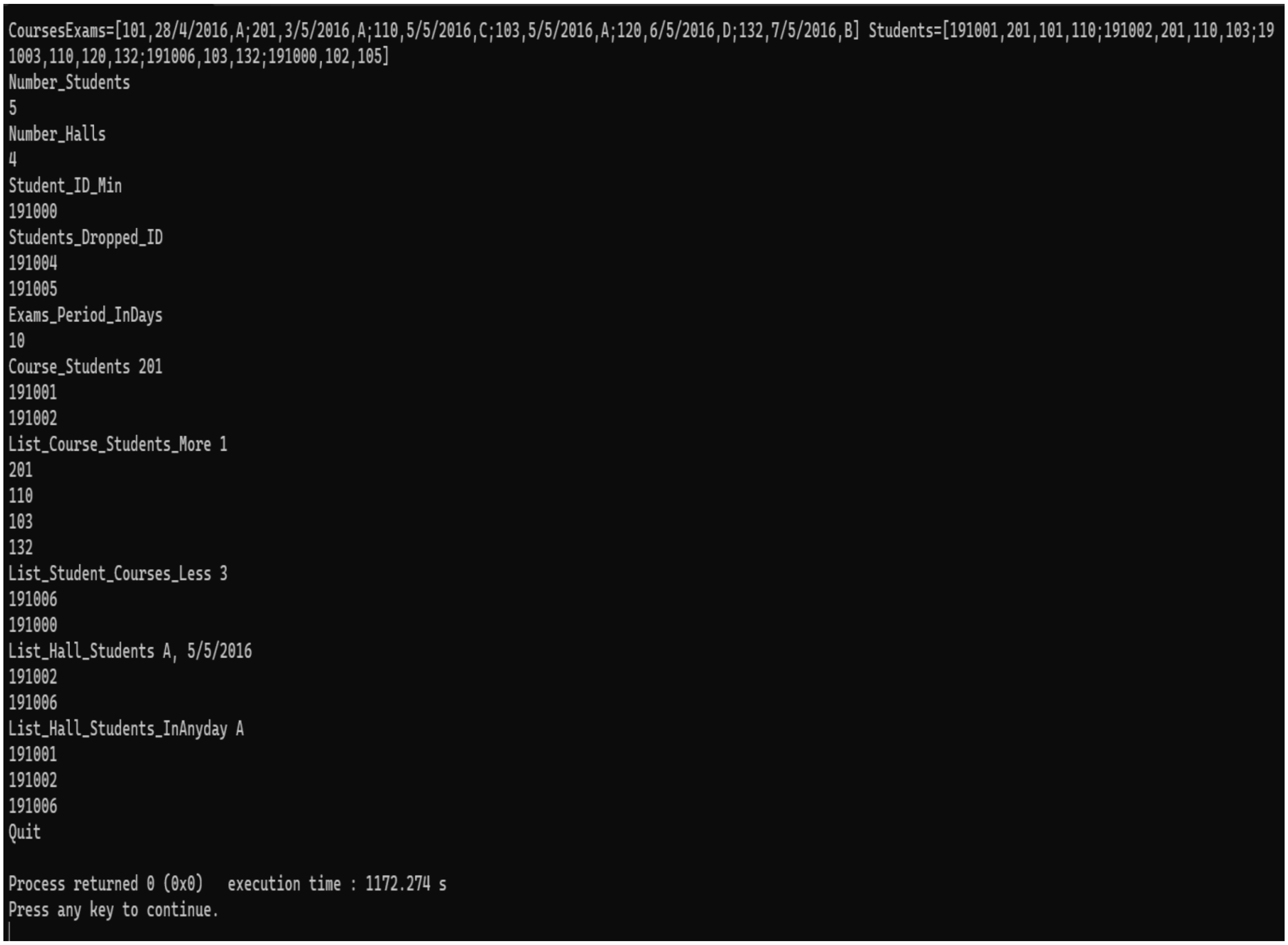
#include <string.h>

Functions used: strcmp()

#include <stdbool.h>

Declaration and use of the bool variable

Input/Output Samples



```
CoursesExams=[101,28/4/2016,A;201,3/5/2016,A;110,5/5/2016,C;103,5/5/2016,A;120,6/5/2016,D;132,7/5/2016,B] Students=[191001,201,101,110;191002,201,110,103;19
1003,110,120,132;191006,103,132;191000,102,105]
Number_Students
5
Number_Halls
4
Student_ID_Min
191000
Students_Dropped_ID
191004
191005
Exams_Period_InDays
10
Course_Students 201
191001
191002
List_Course_Students_More 1
201
110
103
132
List_Student_Courses_Less 3
191006
191000
List_Hall_Students A, 5/5/2016
191002
191006
List_Hall_Students_InAnyday A
191001
191002
191006
Quit

Process returned 0 (0x0)   execution time : 1172.274 s
Press any key to continue.
```

Team Roles/Responsibilities

Functionality	Responsible Team Member(s)
Function 1, Function 2, Function 9	Nour El Bakary
Function 3, Function 4	Areej Abdelghany Taher
Function 5, Function 6	Sondos Aly
Function 7, Function 8	Farah sultan
Main, Function 10	Malak Samy
Function 11	Nour El Bakary
INPUT	Malak Samy
Arranged Project Report	Sondos Aly
Project report details	All of us
Presentation Slides	Areej Taher, Sondos Ahmed, Nour El Bakary