University of Abdelhamid Mehri - Constantine 2

NTIC Faculty – First Year State Engineer, 2023–2024

Algorithms and Data Structures (ADS1)

Mini Project: Student Record Management System

Due before January 09th, 2024

Overview

In this project, the students will create a program in C to manage student records using arrays. The objectives are to:

- Implement a user-friendly menu system with multiple options.
- Perform basic record management operations.
- Display the report cards of students.
- Handle errors.

Detailed description

- **Menu System**

  The management system uses a menu that offers the following options:

  ✓ Basic management operations
    * Add a new student record.
    * Display a student record.
    * Delete a student record.
    * Fill in the marks of a student.
    * Display the result (s) of a search based on any information entered by the user.

  ✓ Display a report card of a specific student.

  ✓ Exit the program.

  This menu will be shown as long as the user does not choose to exit the program.

- **Data Structures**

  To implement the record management system, the student will use:

  – A one-dimensional array, named *Student_Records*, storing the student records. Each record contains the following: ID number, first name, last name, date of birth, address, student class, emergency contact information and overall average, initially receives -1.

– A two-dimensional array, named *Mark_Matrix*, containing the marks of the students in 4 subjects: Physics, Maths, Arabic and English. The columns represent the subjects whereas the rows represent the students as shown in Figure 1.

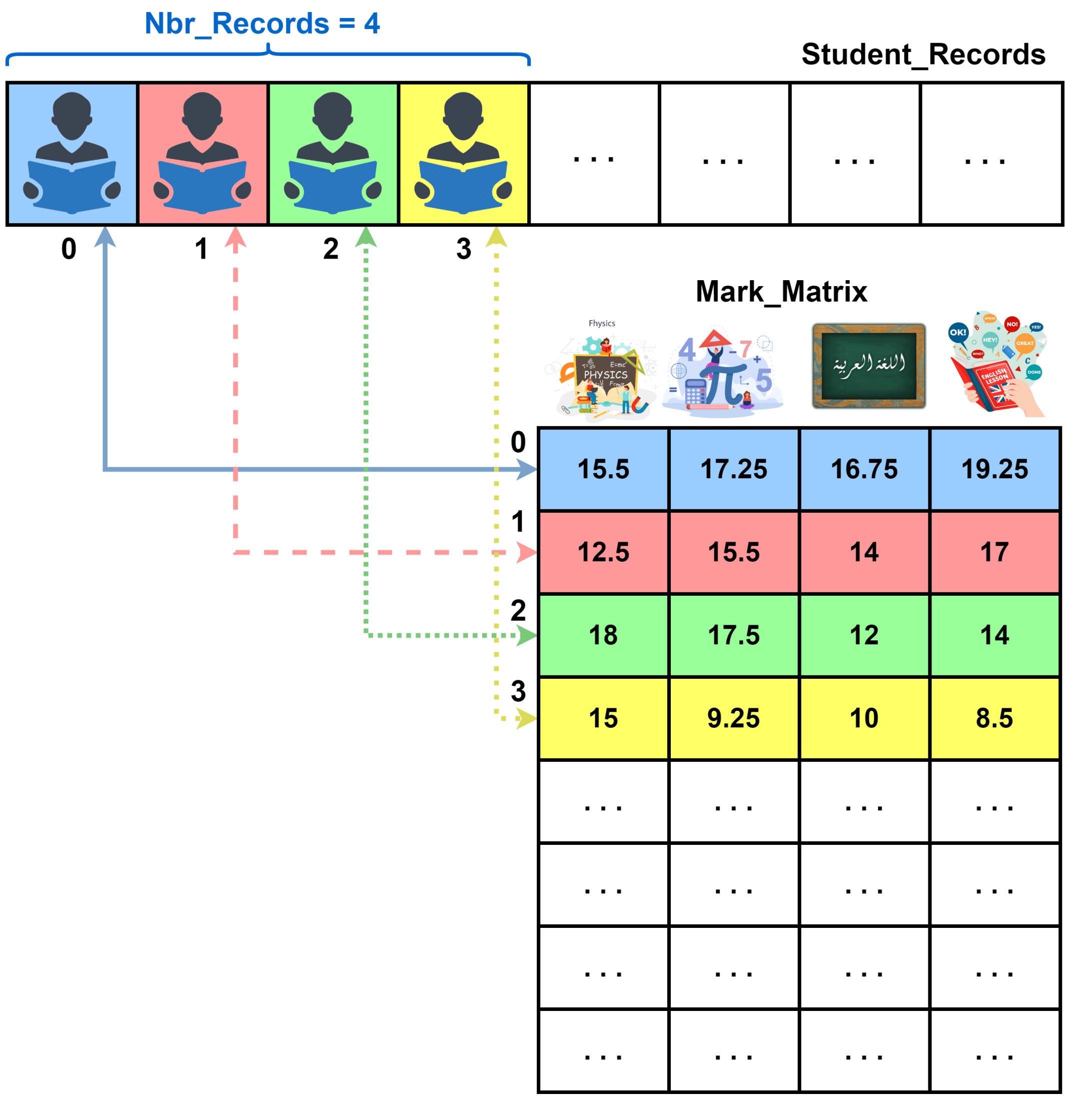


Figure 1: Data structures

- **Functions/Procedures implementation**

  Functions

  – ***Add(Student_Records, Nbr_Records, New_Record):*** this function adds a new record and returns the current number of records in the array *Student_Records*.

  – ***Search(Student_Records, Nbr_Records, First_Name, Last_Name):*** this function returns the index of the student record if it exists, otherwise it returns -1.

  – ***Delete(Student_Records, Nbr_Records, Record_Index, Mark_Matrix):*** this function deletes the record identified by the index *Record_Index* and returns the current number of records in the array *Student_Records*.

- ***Fill_Marks(Mark_Matrix, Record_Index):*** this function fills in the row representing a specific student (row number = *Record_Index*) relying on the marks entered by the user and returns the overall average, which will be used by the *main* to update the student's overall average.

Procedures

- ***Display_Record(Record):*** this procedure displays the information of the record.
- ***Display_Report_Card(Record, Record_Index, Mark_Matrix):*** this procedure displays the report card of the student of the record *Record_Index*, if his/her marks have been filled in. Otherwise, it shows an error message.
- **Display_By_Criteria (Student_Records, Nbr_Records):** this procedure allows the user to perform a search based on any information (e.g., first name, last name, average, etc.) and display the corresponding result (s). For example, if the user enters 15.25, the procedure displays the information of all students having an average equal to 15.25.

Hints

- To call the functions *Delete* and *Fill_Marks*, the *main* firstly calls the function *Search* to determine the *Record_Index* of a student based on his/her first name and last name.
- To call the procedures *Display_Record* and *Display_Report_Card*, the *main* firstly calls the function *Search* to determine the *Record_Index* of a student based on his/her first name and last name. Then, it passes *Student_Records*[*Record_Index*] as a parameter.

• **Error Handling**
If the user enters invalid inputs, a warning message should be shown. For example, if he enters a negative mark, a possible error message is *"Error: A mark cannot be lower than zero."*.

Project Submission

• It is requested to create a folder named *MP2_FirstName_LastName_Group* including:

✓ A PDF file named 'Report' containing:

  ∗ An algorithm for each function and procedure.
  ∗ A flowchart for each function and procedure.

✓ A single C program, named 'main.c' allowing the user to manage student records and display their report cards.

• Only the Google Drive link to this prepared folder should be submitted by sending an email to **ing.ads1.ntic@gmail.com** The subject of the email should be *Mini Project 2 FirstName LastName Group*.

Appendix: Useful Information

Records

A. Declaration of a record Product

Algorithmic Syntax

```
Types
  Product = Record
              Reference: Integer
              PType: Char
              Price: Real
              Quantity: Integer
            End
Variables
  p1, p2: Product
```

Syntax in C

```
typedef struct product {
    int Reference;
    char PType;
    float Price;
    int Quantity;
} Product;


Product p1, p2;
```

B. Reading a product

Algorithmic Syntax

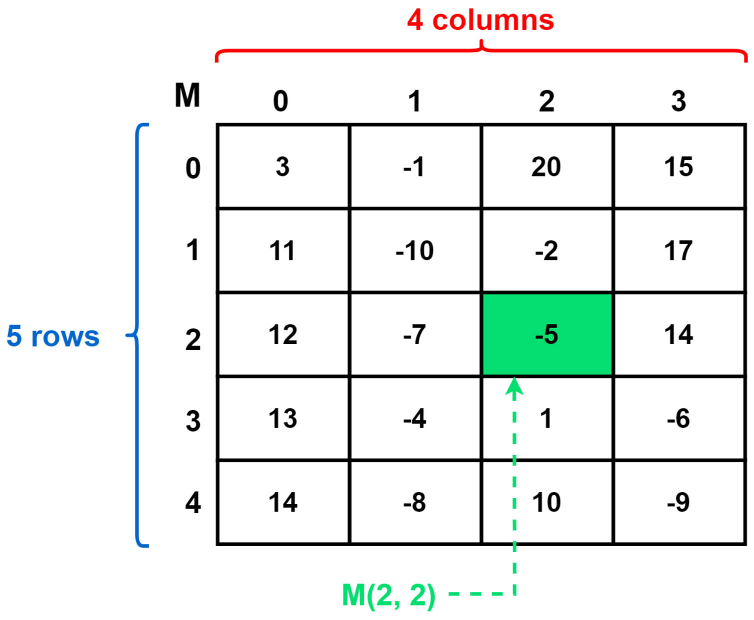
```
Read(p1.Reference, p1.PType, p1.Price, p1.Quantity)
```

Syntax in C

```
scanf(”%d %c %f %d”, &p1.Reference, &p1.PType, &p1.Price, &p1.Quantity);
```

Two-dimensional arrays

Supposing we have M a two-dimentional array containing 5*4 integers as shown below.



A. Declaration of M

<u>Algorithmic Syntax</u>

```
Constants
    D1 = 5
    D2 = 4
Variables
    M: Array of D1*D2 Integer
```

<u>Syntax in C</u>

```c
#define D1 5
#define D2 4


int M[D1][D2];
```

B. Reading the elements of M

<u>Algorithmic Syntax</u>

```
For i ← 0 to D1-1 Do
    For j ← 0 to D2-1 Do
        Read(M(i, j))
    EndFor
EndFor
```

<u>Syntax in C</u>

```c
for (i=0; i < D1; i++)
{
    for (j=0; j < D2; j++)
    {
        scanf("%d", &M[i][j]);
    }
}
```