Ministry of Higher Education and Scientific Research
HassibaBenbouali University of Chlef
Faculty of Exact Sciences and Computer Science
1st year CS
Module: Algorithmics and data structure 1

Tutorial N° 06: Records

Exercise 01:

Create a record named « Student » which is characterized by an identifier, a last name and a first name. You are asked to enter 10 students, arrange them in a table and then display them

Exercise 02:

We repeat the previous exercise but we add two grades for each student. You are therefore asked to create a new record named « Notes » which this time contains the continuous assessment grade noted « NoteCC » as well as the exam grade which is noted « NoteEx ». Edit the previous « Student » record so that it can relate to the « Notes » one. You are asked to write:

1- An algorithm that allows you to enter the list of students as well as their grades.
2- An algorithm that allows you to display the list of students with their grades.
3- An algorithm that displays the student who had the best exam grade.
4- An algorithm that displays the overall class average.

Exercise 03 :

Define a « Date » record containing the day, month, and year. Enter two dates (we will assume that they are valid dates) and display whether the first date is before or after the second.

Supplementary exercises

Exercise 04 :

Declare the types that allow you to store:

1. **An employee**, characterized by their **full name**, **date of birth**, **position**, **wilaya of assignment**, and **gender**.
2. **An association of employees** (a group or list of employees).
3. **An enterprise annex** containing its **name**, the **employees who work in it**, as well as the **annual revenue**, the **expenses**, and the **number of clients served**.
4. **A table of 58 enterprise annexes** (one for each wilaya of Algeria).

Exercise 05 :

We are interested in the management of vehicles in a car fleet. Each vehicle is characterized by a car registration number, a brand, a model, a color, the number of seats, a tax power.

1- Give the adequating record.
2- Break down the car registration number into its elementary components then give the new structure of the record.
3- Write an algorithm that allows you to:
- Store information about a car fleet that includes a maximum of 50 vehicles.
- Establish a statistical array containing the number of vehicles registered per wilaya.

Exercise 06 :

```
TYPE Date = RECORD
    day, month, year : INTEGER;
END RECORD;

TYPE Address = RECORD
    Number : INTEGER;
    Street : STRING[50];
    City : STRING[20];
    Wilaya : STRING[20];
    PC : INTEGER;   /* Postal Code */
END RECORD;

TYPE Student = RECORD
    LastName, FirstName : STRING[20];
    BirthDate : Date;
    HomeAddress : Address;
    BacAverage : REAL;
    Field : STRING[20];      /* Example: Math, Science, IT… */
END RECORD;
```

Questions:

1. Complete an array TStu with n students ($n \leq 300$).
2. Display the LAST NAMES and FIRST NAMES of all students whose baccalaureate average is between two values A and B entered by the user.
3. Display the students who live in a given city and wilaya entered by the user.

Solution:

1. Declare and filling:

```
#include <stdio.h>
//#define MAX 10;
typedef struct {
   int id;
   char ln[20];
   char fn[20];
} Student;

int main() {
Student st[10];// Student st[MAX];

   /* Input 10 students */
   for(int i = 0; i < 10; i++) {
      printf("\n--- Student %d ---\n", i+1);
      printf("ID: ");
      scanf("%d", &st[i].id);
      printf("Last Name: ");
      scanf("%s", st[i].ln);
      printf("First Name: ");
      scanf("%s", st[i].fn);
   }

   /* Display the students */
   printf("\nList of students:\n");
   for(int i = 0; i < 10; i++) {
      printf("ID: %d, Name: %s %s\n", st[i].id, st[i].ln, st[i].fn);
   }

   return 0;
}
```

2. add two grades:

- **solution 1: add the two grades in the same struct or in the same record:**

```
typedef struct {
   int id;
   char ln[20];
   char fn[20];
   float NoteCC;   // Continuous assessment
   float NoteEx;   // Exam grade
} Student;
```

- **Solution 2: creat a new record called grade contains two variables**:

```
#include <stdio.h>
#define MAX 10              //  you can delete it
/* Structure for notes */
typedef struct {
   float NoteCC;   // Continuous assessment
   float NoteEx;   // Exam grade
} Notes;

/* Structure for student */
typedef struct {
   int id;
   char ln[20];
   char fn[20];
```

```
    Notes grade;            // Link to Notes
} Student;

int main() {
    Student st[MAX];

    /* 1. Enter students and their grades */
    for(int i = 0; i < MAX; i++) {
        printf("\n--- Student %d ---\n", i+1);
        printf("ID: "); scanf("%d", &st[i].id);
        printf("Last Name: "); scanf("%s", st[i].ln);
        printf("First Name: "); scanf("%s", st[i].fn);
        printf("Continuous Assessment (NoteCC): "); scanf("%f", &st[i].grade.NoteCC);
        printf("Exam Grade (NoteEx): "); scanf("%f", &st[i].grade.NoteEx);
    }

    /* 2. Display students with their grades */
    printf("\nList of students and their grades:\n");
    for(int i = 0; i < MAX; i++) {
        printf("ID: %d, Name: %s %s, NoteCC: %.2f, NoteEx: %.2f\n",
               st[i].id, st[i].ln, st[i].fn, st[i].grade.NoteCC, st[i].grade.NoteEx);
    }

    /* 3. Display the student with the best exam grade */
    int bestIndex = 0;
    for(int i = 1; i < MAX; i++) {
        if(st[i].grade.NoteEx > st[bestIndex].grade.NoteEx) {
            bestIndex = i;
        }
    }
    printf("\nStudent with the best exam grade:\n");
    printf("ID: %d, Name: %s %s, NoteEx: %.2f\n",
           st[bestIndex].id, st[bestIndex].ln, st[bestIndex].fn, st[bestIndex].grade.NoteEx);

    /* 4. Display overall class average (average of NoteCC + NoteEx) */
    float sum = 0;
    for(int i = 0; i < MAX; i++) {
        sum += (st[i].grade.NoteCC + st[i].grade.NoteEx) / 2.0;
    }
    float classAverage = sum / MAX;
    printf("\nOverall class average: %.2f\n", classAverage);

    return 0;
}
```

Exercise 03:

```
#include <stdio.h>
/* Structure for date */
      typedef struct {
          int day;
          int month;
          int year;
      } Date;

int main() {
    Date d1, d2;

    /* Input two dates */
    printf("Enter first date (dd mm yyyy): ");
```

```
    scanf("%d %d %d", &d1.day, &d1.month, &d1.year);

    printf("Enter second date (dd mm yyyy): ");
    scanf("%d %d %d", &d2.day, &d2.month, &d2.year);

    /* Compare dates */
    if(d1.year < d2.year) {
        printf("The first date is BEFORE the second date.\n");
    } else if(d1.year > d2.year) {
        printf("The first date is AFTER the second date.\n");
    } else {
        // in this case , Years are equal, so we must compare months
        if(d1.month < d2.month) {
            printf("The first date is BEFORE the second date.\n");
        } else if(d1.month > d2.month) {
            printf("The first date is AFTER the second date.\n");
        } else {
            // in this case years and Months are equal,  so we must compare days
            if(d1.day < d2.day) {
                printf("The first date is BEFORE the second date.\n");
            } else if(d1.day > d2.day) {
                printf("The first date is AFTER the second date.\n");
            } else {
                printf("the two  dates are EQUAL.\n");
            }
        }
    }

    return 0;
}
```