
Unit 7:

Input/Output Files (II)

Reading more than one piece of
information each time

Exercise

- Write a program which reads from a file named 'race.txt' containing some race results in the following format:

Content of file 'race.txt':

1111 Duncan Kibet 03-30-28

0234 Heile Gebreselassie 03-35-12

6781 James Kwambai 03-50-01

6331 Juan Pérez 03-55-55

and print the data on screen as this:

<i>Name</i>	<i>Time</i>
<i>D. Kibet</i>	<i>3:30:28</i>
<i>H. Gebreselassie</i>	<i>3:35:12</i>
<i>J. Kwambai</i>	<i>3:50:1</i>
<i>J. Pérez</i>	<i>3:55:55</i>

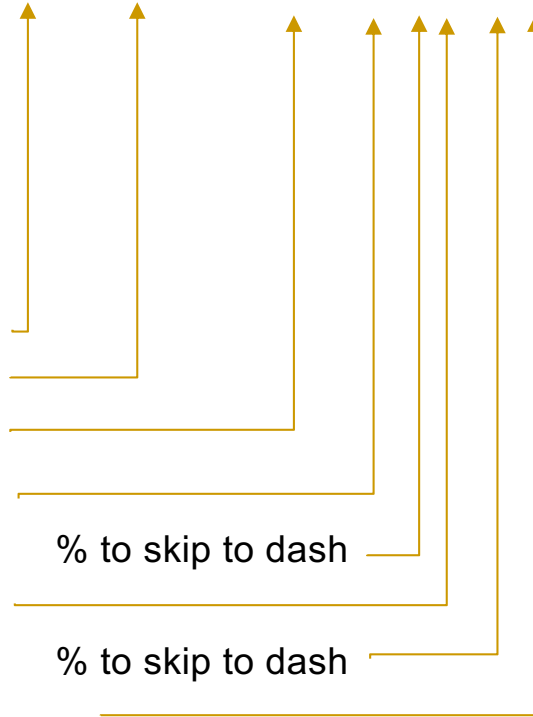
Exercise

```
fid = fopen('race.txt','rt');
if (fid == -1)
    disp('Error. Could not open the files.');
```

else

```
    fprintf('\nName\tTime');
    while notfeof(fid))
        runner.dorsal    = fscanf(fid,'%d',1);
        runner.name      = fscanf(fid,'%s',1);
        runner.surname   = fscanf(fid, '%s', 1);
        runner.hours     = fscanf(fid, '%d',1);
        vdash            = fscanf(fid, '%c',1);    % to skip to dash
        runner.mins      = fscanf(fid, '%d',1);
        vdash            = fscanf(fid, '%c',1);    % to skip to dash
        runner.seconds   = fscanf(fid, '%d',1);
        fprintf('\n%c.%s\t%d:%d:%d', runner.name(1), runner.surename, runner.hours,
                runner.mins, runner.seconds);
    end;
    fclose(fid);
end;
```

1111 Duncan Kibet 03-30-28
0234 Heile Gebreselassie 03-35-12
6781 James Kwambai 03-50-01
6331 Juan Pérez 03-55-55



Exercise

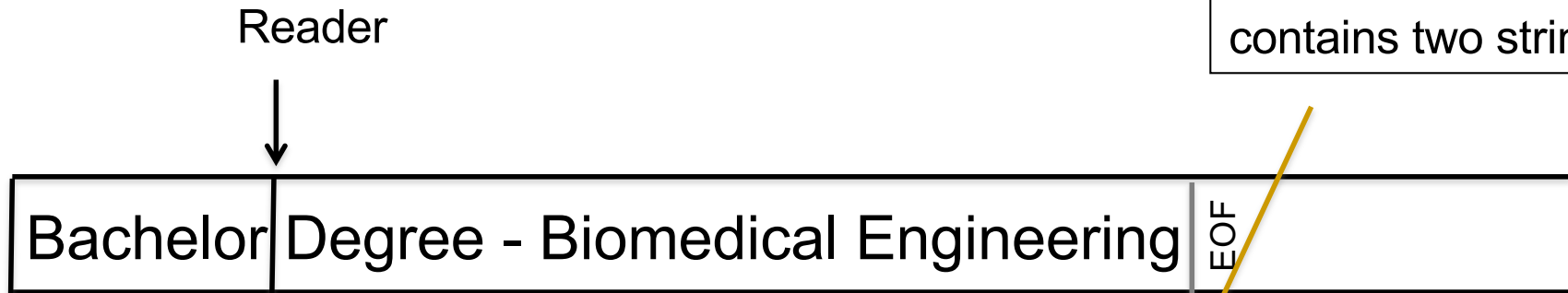
```
fid = fopen('race.txt','rt');
if (fid == -1)
    disp('Error. Could not open the files.');
```

```
else
    fprintf('\nName\tTime');
    while notfeof(fid))
        runner.dorsal    = fscanf(fid,'%d',1);
        runner.name     = fscanf(fid,'%s',1);
        runner.surname  = fscanf(fid, '%s', 1);
        runner.hours    = fscanf(fid, '%d',1);
        vdash           = fscanf(fid, '%c',1);    % to skip to dash
        runner.mins     = fscanf(fid, '%d',1);
        vdash           = fscanf(fid, '%c',1);    % to skip to dash
        runner.seconds  = fscanf(fid, '%d',1);
        fprintf('\n%c.%s\t%d:%d:%d', runner.name(1), runner.surname, runner.hours,
                runner.mins, runner.seconds);
    end;
    fclose(fid);
end;
```

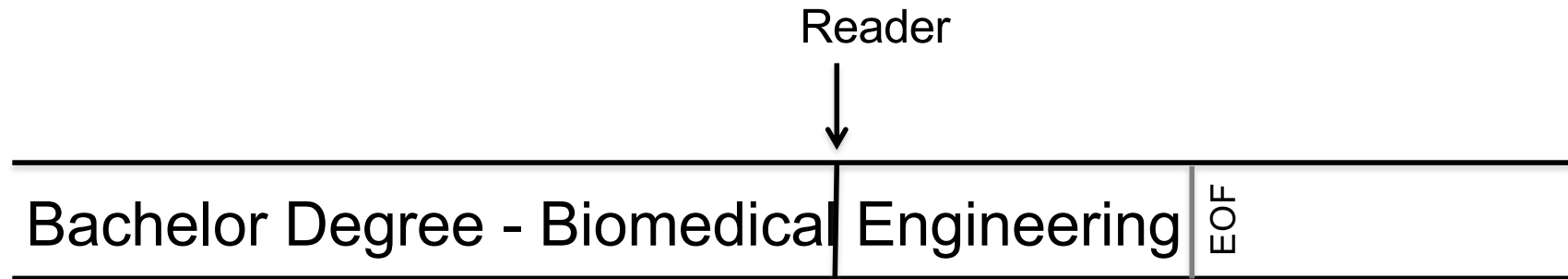
This solution it's ok... but it is really long to write, and it requires to read symbols we are not interested in (i.e. the dash)

Reading more than one information at a time: textscan

We are going to read one 'block' of information that contains two strings



```
C = textscan (fid, '%s - %s', 1);
```



Reading more than one information at a time: textscan

Same example but skipping the two initial words and the first dash

Reader



Bachelor Degree - Biomedical Engineering | EOF

```
C = textscan (fid, 'Bachelor Degree - %s %s', 1);
```

Reader



Bachelor Degree - Biomedical Engineering | EOF

C

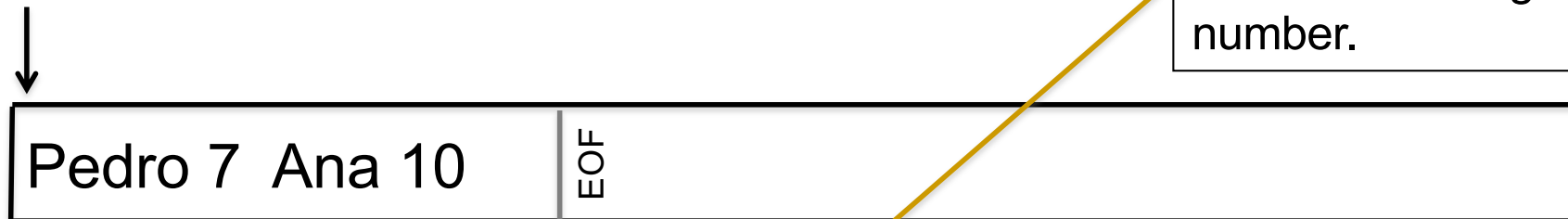
```
{ { 'Biomedical' } }  
{ { 'Engineering' } }
```

```
C{1}{1} ← 'Biomedical'  
C{2}{1} ← 'Engineering'
```

Reading more than one information at a time: textscan

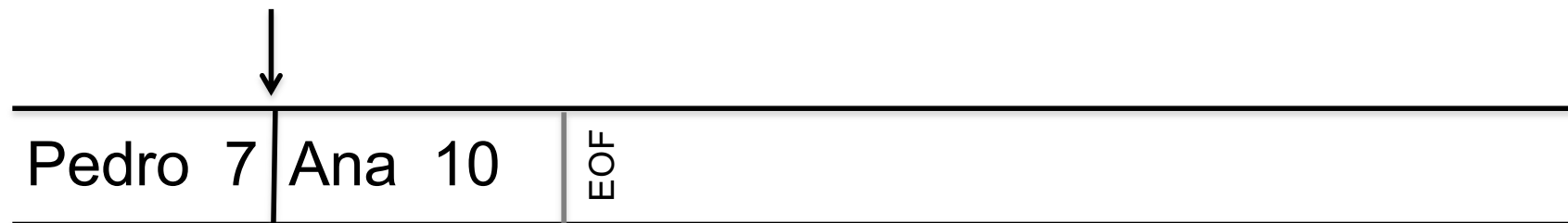
We are going to read one 'block' of information that contains a String and a number.

Reader

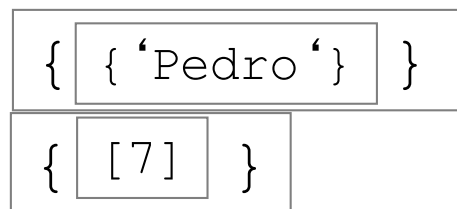


```
C = textscan (fid, '%s %d', 1);
```

Reader



C

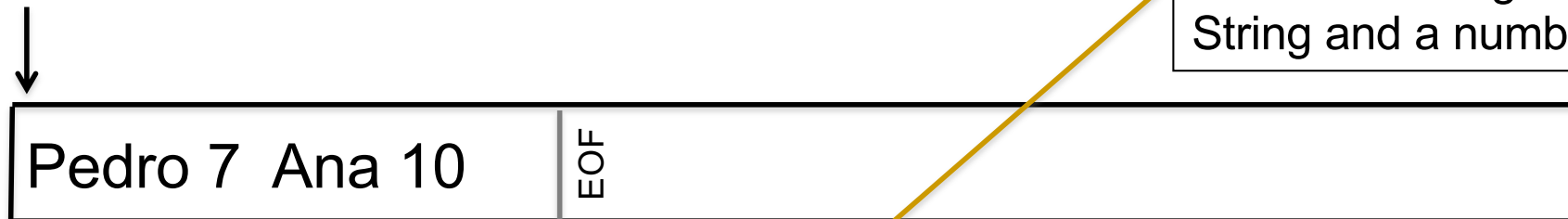


C{1}{1} ← 'Pedro'
C{2}(1) ← 7

Reading more than one information at a time: textscan

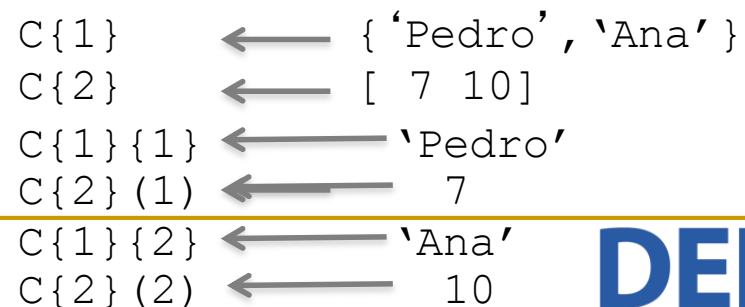
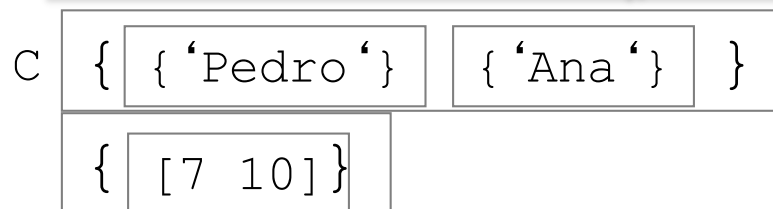
We are going to read **TWO** 'blocks' of information, each of them containing a String and a number.

Pointer



```
C = textscan (fid, '%s %d', 2);
```

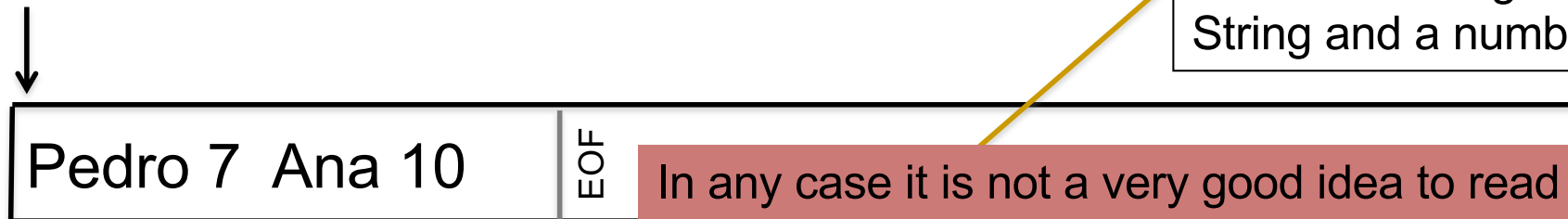
Pointer



Reading more than one information at a time: textscan

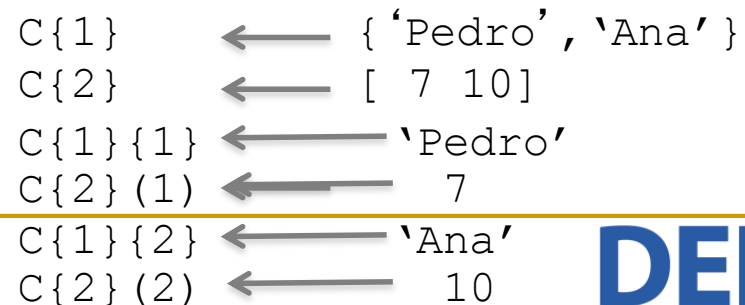
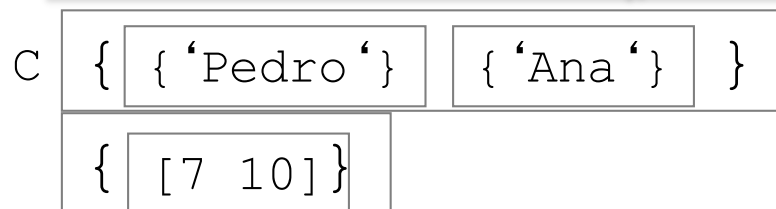
We are going to read **TWO** 'blocks' of information, each of them containing a String and a number.

Pointer



In any case it is not a very good idea to read more than block of information at once... You might end up with a messy structure of cell arrays difficult to process

```
C = textscan (fid,
             Pointer,
```



Reading more than one information at a time: textscan

- **Examples (cont):**

```
C= textscan (fi, '%s\t%s', 1);
```

Reads two strings separated by a tabulator, and places them in C{1}{1} and C{2}{1}.

```
C= textscan (fi, 'My birthday is the %dth of %s',1);
```

Reads an integer and an string and places them in C{1}(1) and C{2}{1} respectively.

```
C= textscan (fi, 'My birthday is the %dth of %s\n');
```

Reads the sentence 'My birthday..' as many times as it can. It returns a cell array in which the first cell contains an array of integers representing the day and the second cell contains and array of cells containing the strings.

Example

- Write a program which reads from a file information about some race results and print it on screen in the following format:

Content of the file 'race.txt'

1111 Duncan Kibet 03-30-28

0234 Heile Gebreselassie 03-35-12

6781 James Kwambai 03-50-01

6331 Juan Pérez 03-55-55

Screen output:

<i>Name</i>	<i>Time</i>
<i>D. Kibet</i>	<i>3:30:28</i>
<i>H. Gebreselassie</i>	<i>3:35:12</i>
<i>J. Kwambai</i>	<i>3:50:1</i>
<i>J. Pérez</i>	<i>3:55:55</i>

Example

This way we obtain the initial of the name

```
clear cRunner;
fid = fopen('race.txt','rt');
if (fid == -1)
    disp('Error. Could not open the files.');
```

else

```
    fprintf('\nName\tTime');
    while feof(fid) == 0
        cRunner = textscan(fid,'%d %s %s %d-%d-%d',1);
        fprintf('\n%c.%s\t%d:%d:%d',cRunner{2}{1}(1),cRunner{3}{1},cRunner{4}(1),
            cRunner{5}(1),cRunner{6}(1));
    end
fclose(fid);
End if;
```

In cRunner{2} we have the name
In cRunner{3} we have the surname
In cRunner{4} we have the hours
In cRunner{5} we have the minutes
In cRunner{6} we have the seconds

Example

- The information about the wages in a company are stored in a text file named 'wages.txt' which follow the format:

employeeID Category Salary

Write a program which updates the salaries of the employees according to their category. This way 'Managers' should see their wage increase a 10%, 'Agents' a 20% and 'Programmers' a 50%.

Example of content of the file 'wages.txt'

34 Agent 1500

74 Manager 2000

66 Manager 1700

99 Programmer 1500

Example

- The information about the wages in a company are stored in a text file named 'wages.txt' which follow the

- Best way to solve this type of problems:

- 1.- *Read the whole file and organize the information in memory using a data structure (in most cases the best one will be a vector of structures)*
- 2.- *Modify/update/transform the information in the data structure*
- 3.- *Write the whole file from beginning to end*

74 Manager 2000

66 Manager 1700


99 Programmer 1500

Example

```
clear cInfo;
fid = fopen('wages.txt','rt');
if (fid == -1)
    disp('Error. Could not open the files.');
```

else

```
    count = 0;
    while feof(fid) == 0
        cInfo = textscan(fid,'%d %s %f',1);
        cont = cont + 1;
        employee(cont).id      = cInfo{1}(1);
        employee(cont).category = cInfo{2}{1};
        employee(cont).salary  = cInfo{3}(1);
    end
    fclose(fid);
```



1st: Read the whole file and store the information in a data structure (in this case a vector of structures)

Example

2nd: Update the information

```
for i=1:cont
    if (strcmp(employee(i).category,'Manager')== 1)
        employee(i).salary = employee(i).salary + employee(i).salary * 0,1;
    elseif (strcmp(employee(i).category,'Agent')== 1)
        employee(i).salary = employee(i).salary + employee(i).salary * 0,2;
    elseif (strcmp(employee(i).category,'Programmer')== 1)
        employee(i).salary = employee(i).salary + employee(i).salary * 0,5;
    end
end
```

3rd: Write the file

```
fid = fopen('wages.txt','wt');
```

```
for i=1:cont-1
    fprintf(fid,'%d %s %f\n', employee(i).id,employee(i).category, employee(i).salary );
end;
% The last line of the file does not include the 'change of line' character.
fprintf(fid, '%d %s %f', employee(cont).id,employee(cont).category, employee(cont).salary );
fclose(fid);
```

Exercise

- Write a program which reads a file named 'exam.txt' that contains the results of the last exam of some students. The format of the lines of file is:

studentNIU studentName result

Then the program displays a menu like this in screen:

Select one of the following options:

- 1.- Display students' information*
- 2.- Modify students' results*
- 3.- Lists students data*
- 4.- Save*
- 5.- Exit*

Option1: the program requires a student's NIU and displays the information of the correspondent student.

Option2: the program requires a student's NIU and a value and updates the information about the student's exam with that value.

Option:3: list in screen the information of all students.

Option4: saves the file with the current information.

Option 5: exit

Exercise

- Example of content of the file:

10004566 Esther 5

10007834 Marta 8

10006667 Laura 7

10007666 Elena 10

Exercise

■ Example of execution:

Select one of the following options:

1.- Display students' information

2.- Modify students' results

3.- Lists students data

4.- Save

5.- Exit

Your selection is: 1

***** DISPLAYING STUDENTS' INFORMATION ****

Introduce a NIU: 10006667

NIU: 10006667

Name: Laura

Result: 7

Exercise

■ Example of execution:

Select one of the following options:

1.- Display students' information

2.- Modify students' results

3.- Lists students data

4.- Save

5.- Exit

Your selection is: 2

*** MODIFYING STUDENT'S RESULTS ***

Introduce a NIU: 10007834

Introduce the result: 10

Information updated

Exercise

■ Example of execution:

Select one of the following options:

- 1.- Display students' information*
- 2.- Modify students' results*
- 3.- Lists students data*
- 4.- Save*
- 5.- Exit*

Your selection is: 3

**** STUDENTS LIST ****

<i>Name</i>	<i>NIU</i>	<i>Result</i>
<i>Esther</i>	<i>10004566</i>	<i>5</i>
<i>Marta</i>	<i>10007834</i>	<i>10</i>
<i>Laura</i>	<i>10006667</i>	<i>7</i>
<i>Elena</i>	<i>10007666</i>	<i>10</i>

Exercise

■ Example of execution:

Select one of the following options:

1.- Display students' information

2.- Modify students' results

3.- Lists students data

4.- Save

5.- Exit

Your selection is: 4

Information saved in the file

Exercise

■ Example of execution:

Select one of the following options:

1.- Display students' information

2.- Modify students' results

3.- Lists students data

4.- Save

5.- Exit

Your selection is: 5

Bye!

Exercise

```
clear;
cont = 0;
fid = fopen('results.txt','rt');
if (fid == -1)
    disp('Error when opening the file.');
```

else

```
    while feof(fid) == 0
        vNIU    = fscanf(fid,'%d',1);
        vName   = fscanf(fid,'%s',1);
        vResult = fscanf(fid,'%d',1);
        cont = cont + 1;
        students(cont).NIU    = vNIU;
        students(cont).name   = vName;
        students(cont).result = vResult;
    end
    fclose(fid);
```

or you can do

```
C = textscan(fid, '%d %s %d', 1);
cont = cont + 1;
students(cont).NIU    = C{1}(1);
students(cont).name   = C{2}{1};
students(cont).result = C{3}(1);
```

Exercise

```
option = -1;
while (option ~= 5)
    fprintf('\n ***** ');
    disp('Select one of the following options:');
    disp('1.- Display students information');
    disp('2.- Modify students results');
    disp('3.- Lists students data');
    disp('4.- Save');
    disp('5.- Exit');
    option = input('Your selection is: ');
    if (option < 1) || (option>5)
        disp('Incorrect option');
    else
        switch option
            case 1
```

Exercise

case 1

```
disp('**** DISPLAYING STUDENTS INFORMATION ****');
usNIU = input('Introduce a NIU: ');
index = searchStudent(students, usNIU);
if (index ~= -1)
    fprintf('\n NIU: %d', students(index).NIU);
    fprintf('\n NIU: %s', students(index).name);
    fprintf('\n NIU: %d', students(index).result);
else
    disp('Student does not exist');
end
```

case 2

```
disp('**** MODIFYING STUDENTS RESULTS ****');
usNIU = input('Introduce a NIU: ');
index = searchStudent(students, usNIU);
if (index ~= -1)
    nResult = input('Introduce the result: ');
    students(index).result = nResult;
    disp('Information updated');
else
```

Exercise

```
        disp('Student does not exist');
    end;
case 3
    disp('**** STUDENTS LIST ****');
    fprintf('\nName\tNIU\tResult');
    for i=1:length(students)
        fprintf('\n%s\t%d\t%d', students(i).name, students(i).NIU,students(i).result);
    end;
case 4
    fid = fopen('results.txt','wt');
    for i=1:length(students)-1
        fprintf(fid,'%d %s %d\n', students(i).NIU, students(i).name,students(i).result);
    end;
    fprintf(fid,'%d %s %d', students(i).NIU, students(i).name,students(i).result);
    disp('Information saved in the file');
    fclose(fid);
end;
end;
end;
disp('bye');
end;
```

Exercise

```
function [indexout] = searchStudent(studentVect, niu)
bFound = 0;
cont = 1;
while ((cont <= length(studentVect)) && (bFound ==0))
    if (studentVect(cont).NIU == niu)
        bFound = 1;
    else
        cont = cont +1;
    end;
end;

if bFound == 1
    indexout = cont;
else
    indexout = -1;
end;
end
```

BINARY FILES

Read and Write Binary Files

- A binary file contains any kind of data encoded as a sequences of bytes.
 - The format is specific to the software which handles that information.
 - Ex: .mp3, .jpg, .doc, .
- The process will be:
 - Open file (*fopen*)
 - Read or write information (*fwrite*, *fread*)
 - Close file (*fclose*)

Opening and closing files

- Same functions than ASCII files though different control characters

Characters	Operation
'r'	Open file for reading (default).
'w'	Open file, or create new file, for writing; discard existing contents, if any
'a'	Open file, or create new file, for writing; append data to the end of the file.
'r+'	Open file for reading and writing.
'w+'	Open file, or create new file, for reading and writing; discard existing contents, if any.
'a+'	Open file, or create new file, for reading and writing; append data to the end of the file

Example

- Read 1 integer of 16 bytes from a file named 'example.bin'

Example

- Read 1 integer of 16 bytes from a file named 'example.bin'

```
fid = fopen('example.bin','r');
if fid == -1
    disp('Error when opening the file');
else
    [data,count] = fread(fid, 1, 'int16');
    fclose(fid);
end
```

READING IMAGES

DIRECT ACCESS

Direct access to the elements of a file

- Obtain the position of the file were the next read will be carried out

```
position = ftell(fid)
```



Positive integer specified in bytes from the beginning of the file. When fails its value is -1



File identifier

- To reposition the file position indicator

```
status = fseek (fid, offset, origin)
```



Move position indicator offset **bytes** towards the end of the file (offset > 0) or the beginning of the file (offset < 0)



'bof'
'cof'
'eof'

Beginning of the file
Current position
End of file