Unit 7: Input/Output Files

SUMMARY

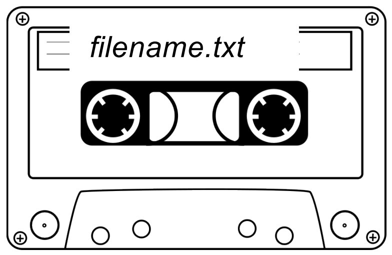
Read and Write ASCII Files

- An ASCII file contains ASCII characters
  - Text files which can be read by a naked eye

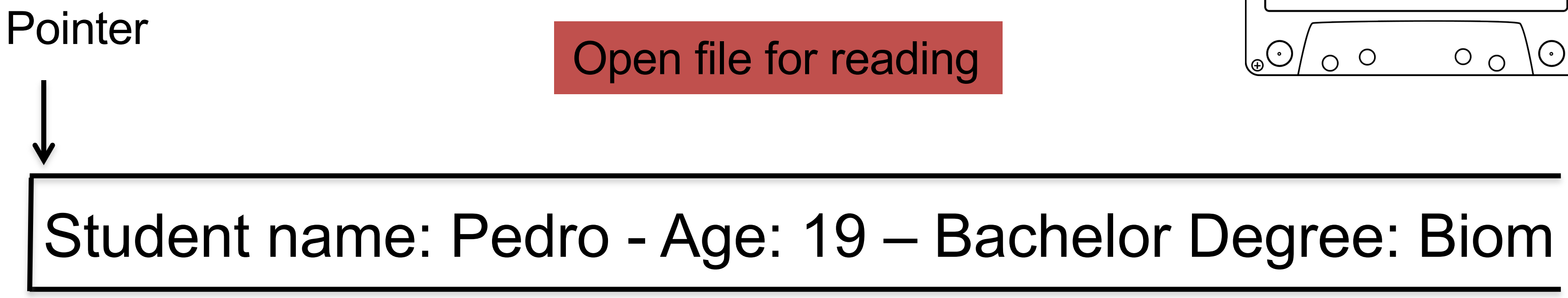
- The process will be:
  - Open file (*fopen*)
  - Read or write information (*fscanf, textscan, fprintf,…*)
  - Close file (fclose)

Opening files

```
fid = fopen ('students.txt', 'rt')
```

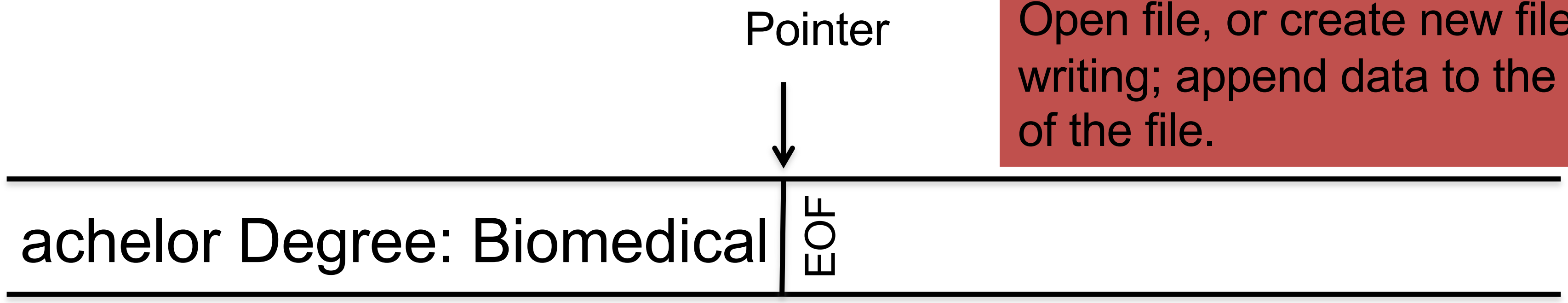


Open file for reading



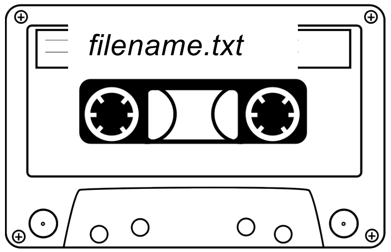
```
fid = fopen ('students.txt', 'at')
```

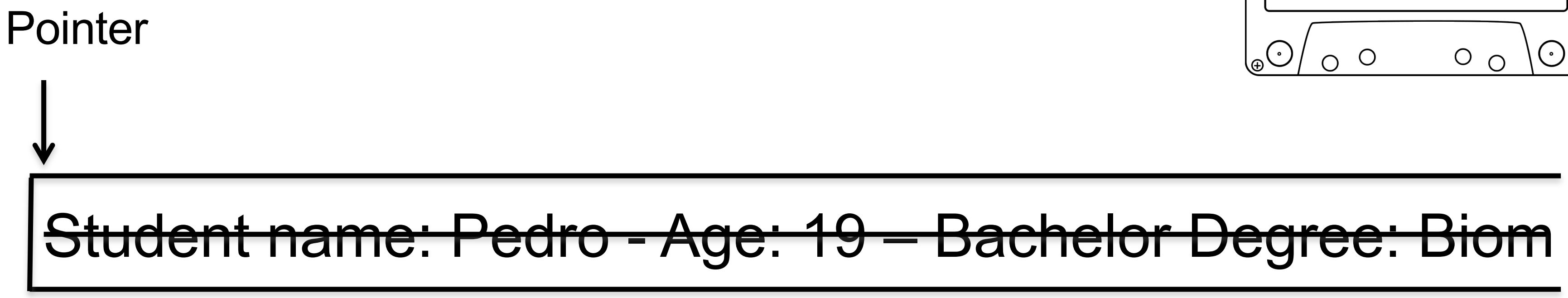
Open file, or create new file, for writing; append data to the end of the file.



Opening files

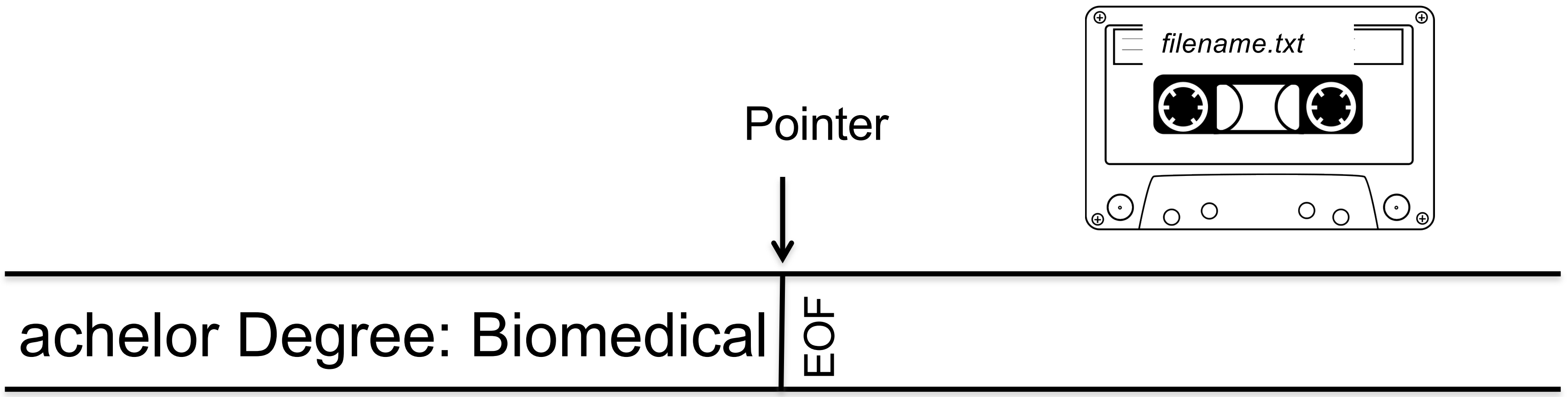
```
fid = fopen ('students.txt', 'wt')
```



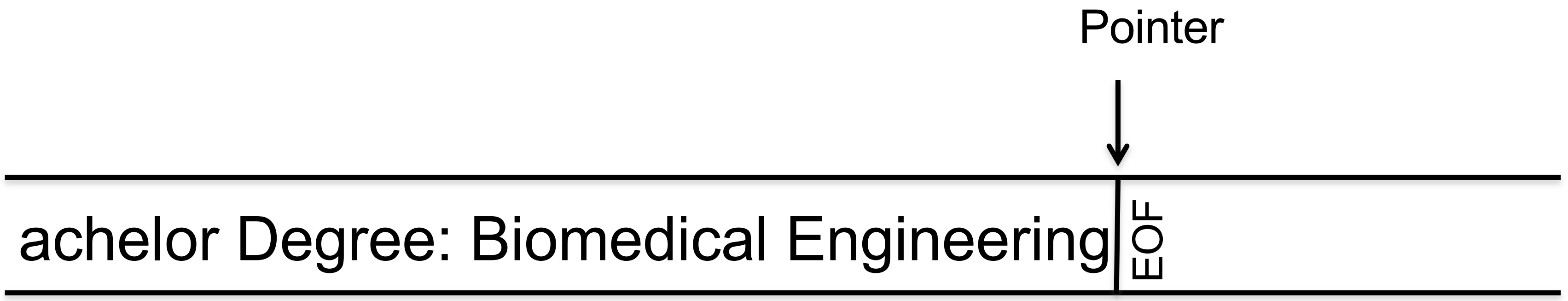


Open file, or create new file, for writing; discard existing contents, if any

Writting ASCII files



```
fprintf(vfile,'%s', 'Engineering');
```



Reading one piece of information each time

Reading ASCII files

- You can use three different commands to read from a text file:
  - fscanf => returns the data read **in a vector or matrix**

```
var = fscanf (fid, ‘%d’, 1);
```

  - textscan => returns the data read **in a cell array**

```
C = textscan (fid, ‘%s’, 1);
```

  - fgets => returns **a whole line** (until \n) of text **in a string**

```
vline = fgets (fid);
```

Reading ASCII files

- You can use three different commands to read from a text file:
  - fscanf => returns the data read **in a vector or matrix**

```
var = fscanf (fid, '%d', 1);
```

  - textscan => returns the data read **in a cell array**

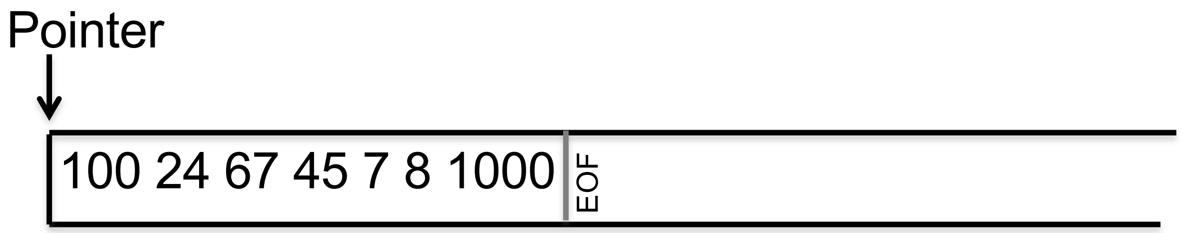
```
C = textscan (fid, '%s', 1);
```

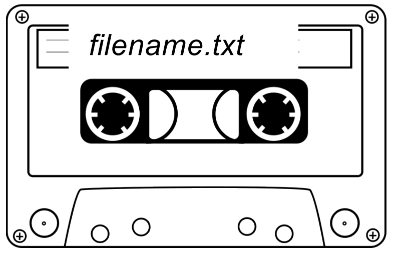
  - fgets => returns **a whole line** (until \n) of text **in a string**

```
vline = fgets (fid);
```

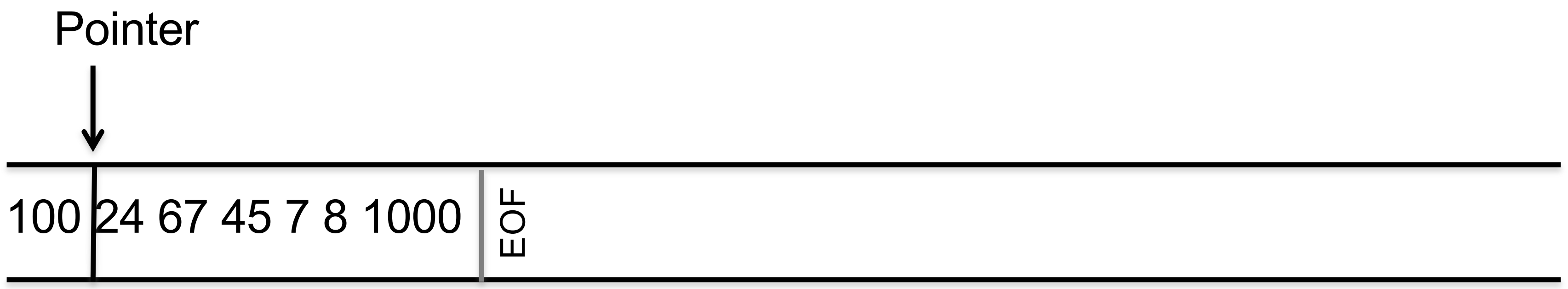
Read one element.
If no number specified it means you want to read the whole file at once

Reading ASCII files





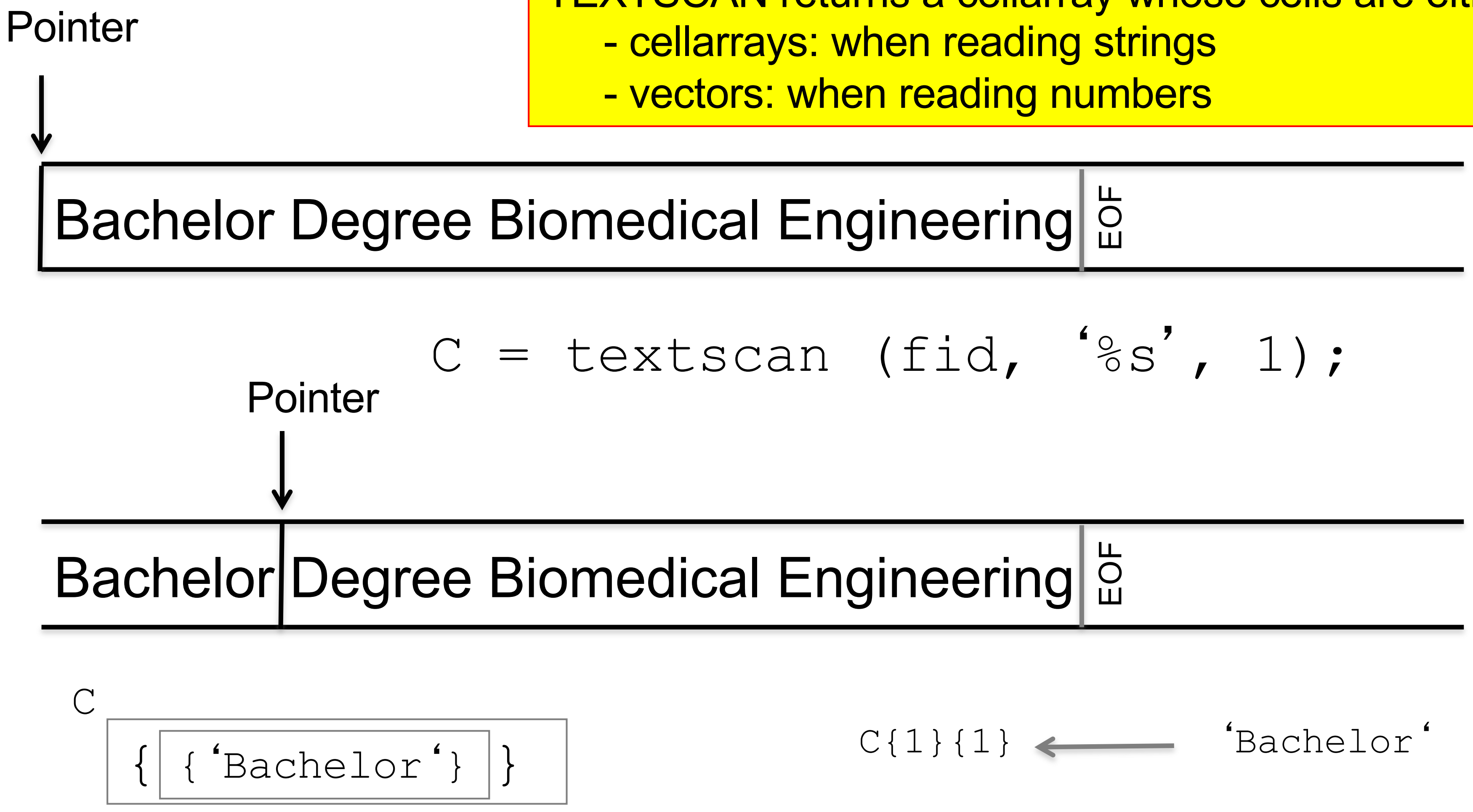
```
var = fscanf (fid, '%d', 1);
```



$$\texttt{var} \leftarrow \texttt{100}$$

Reading ASCII files

IMPORTANT:
TEXTSCAN returns a cellarray whose cells are either:
- cellarrays: when reading strings
- vectors: when reading numbers

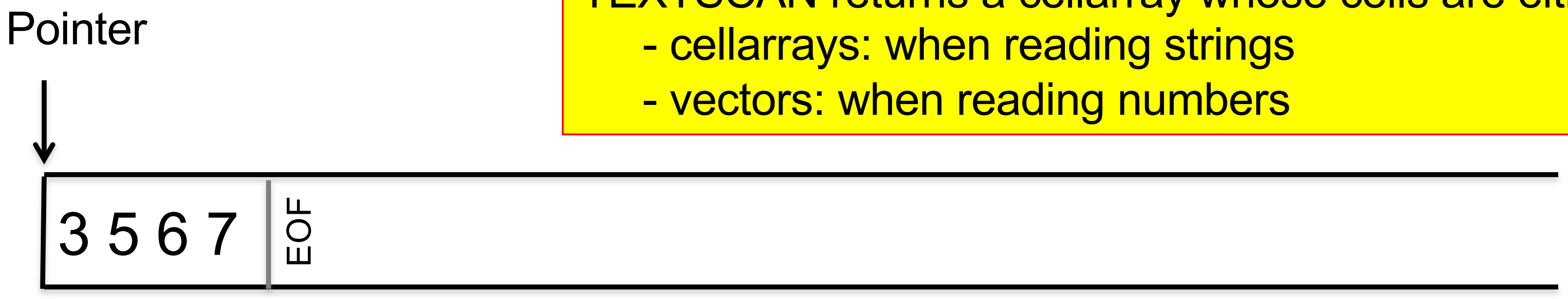


Pointer
Bachelor Degree Biomedical Engineering
EOF
Pointer
Bachelor Degree Biomedical Engineering
EOF
C
{ {'Bachelor'} }
C{1}{1}
'Bachelor'

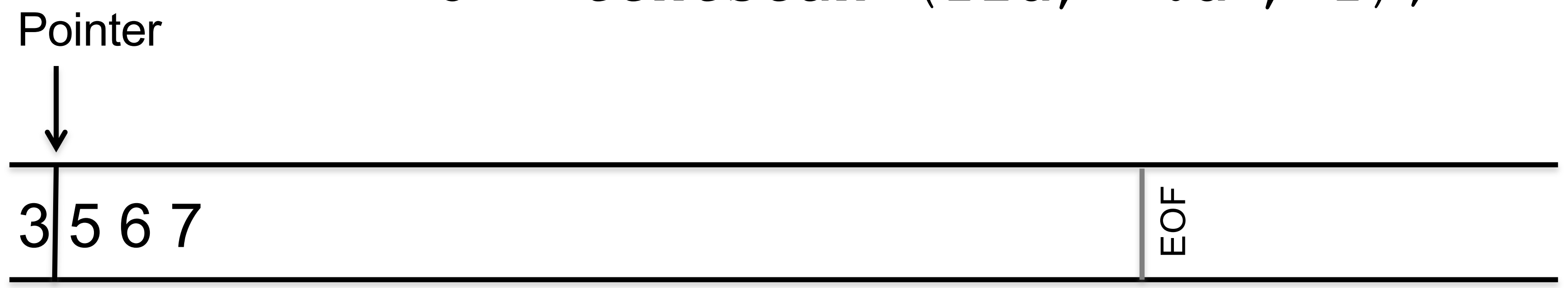
```
C = textscan (fid, '%s', 1);
```

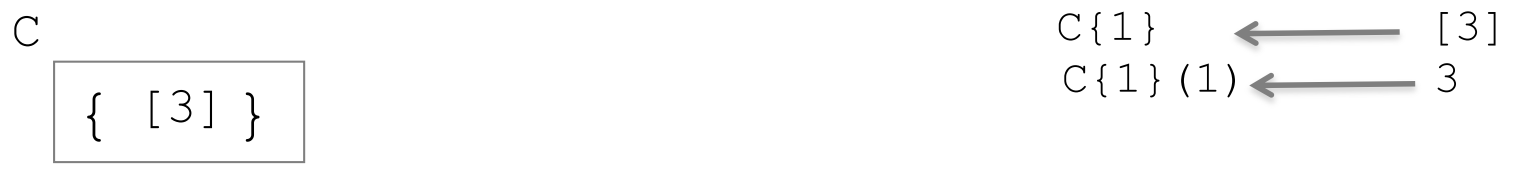
Reading ASCII files

IMPORTANT:
TEXTSCAN returns a cellarray whose cells are either:
- cellarrays: when reading strings
- vectors: when reading numbers

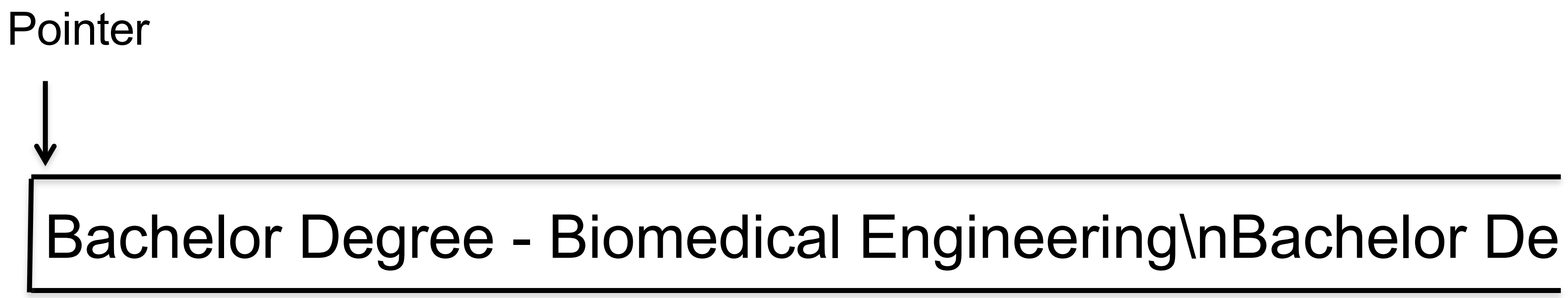


```
C = textscan (fid, '%d', 1);
```

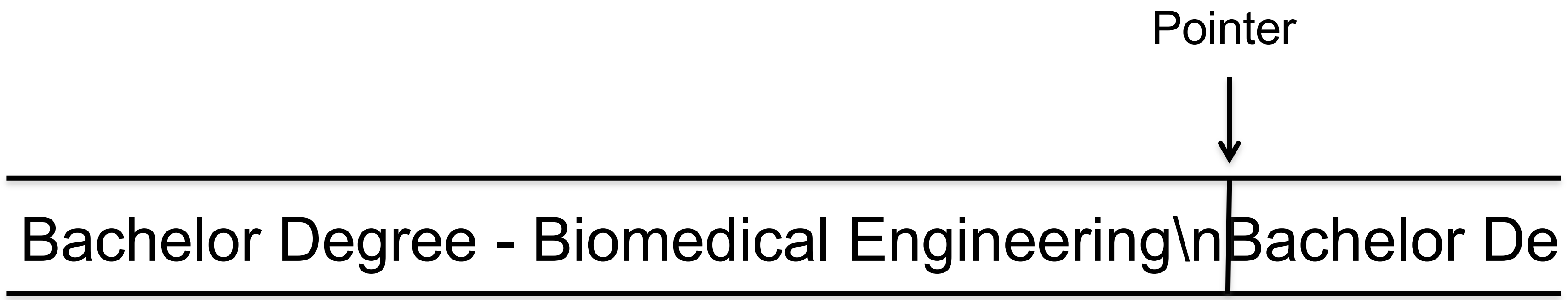




Reading ASCII files in lines



```
vline = fgets (fid);
```



```
vline ← 'Bachelor Degree - Biomedical Engineering\n'
```

Example

- You can read a whole file that contains strings using either one of the three commands and a loop

```
vfile = fopen('sentence.txt','rt');
while feof(vfile) == 0
    vword = fscanf (vfile,'%s',1);
    fprintf('\nThe word is: %s', vword);
end
fclose(vfile);
```

Example

- You can read a whole file that contains strings using either one of the three commands and a loop

```
vfile = fopen('sentence.txt','rt');
while feof(vfile) == 0
    cword = textscan (vfile,'%s',1);
    fprintf('\nThe word is: %s', cword{1}{1});
end
fclose(vfile);
```

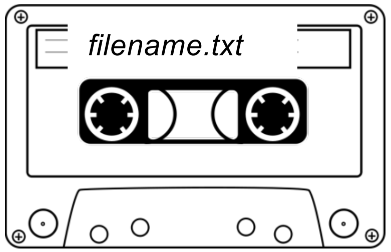
Example

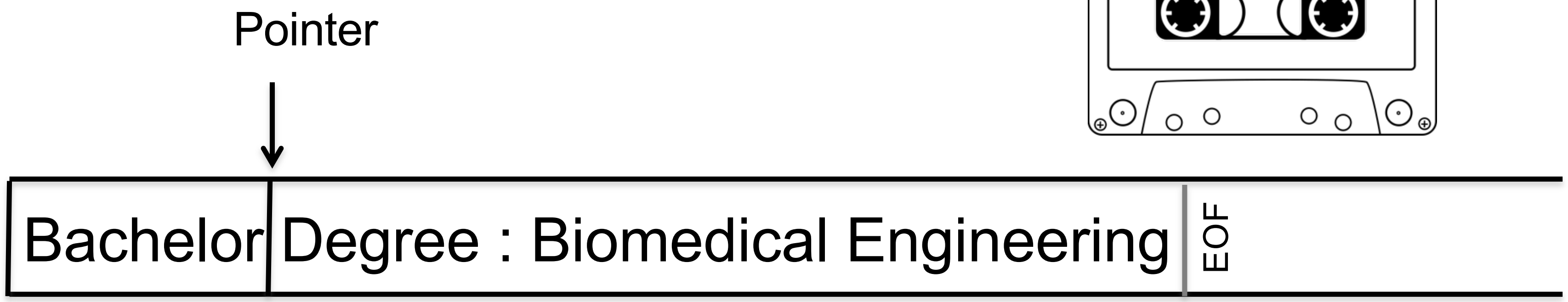
- You can read a whole file that contains strings using either one of the three commands and a loop

```
vfile = fopen('sentence.txt','rt');
while feof(vfile) == 0
    vline = fgets (vfile);
    fprintf('\nThe line is: %s', vline);
end
fclose(vfile);
```

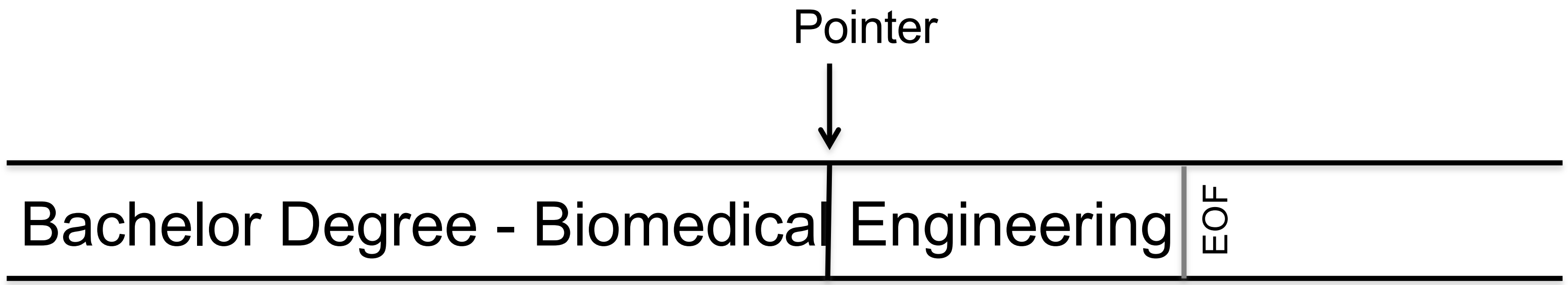
Reading more than one piece of information each time

Reading more than one information at a time: textscan





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



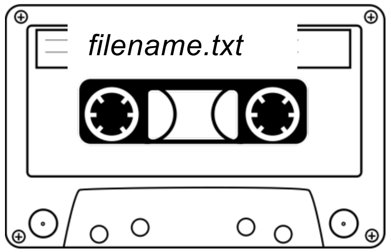
```
C   { { 'Degree' }         }
    { { 'Biomedical' }     }
```

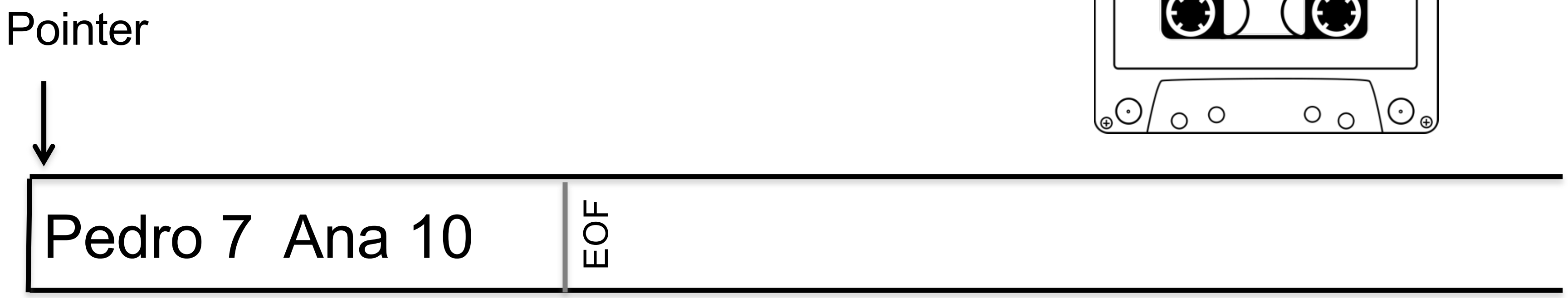
```
C{1}{1}  ←  'Degree'
C{2}{1}  ←  'Biomedical'
```

uc3m | Universidad **Carlos III** de Madrid

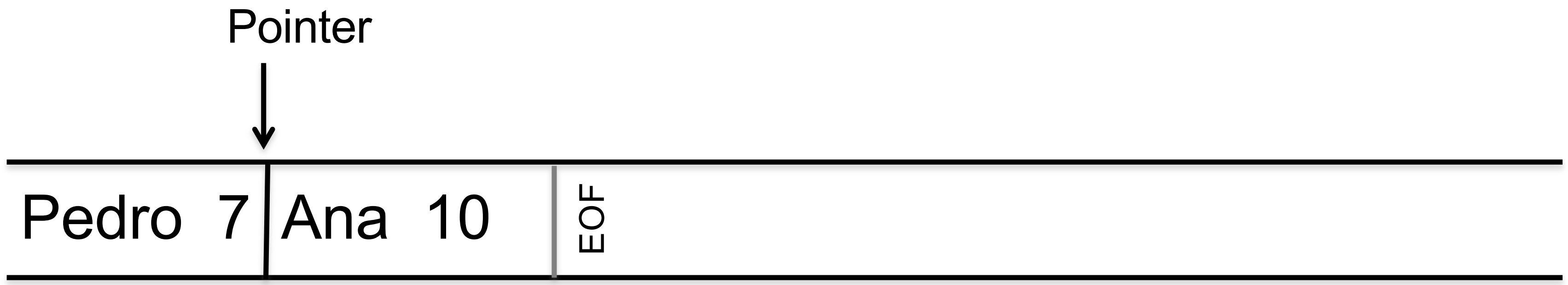
DEI Interactive Systems Group

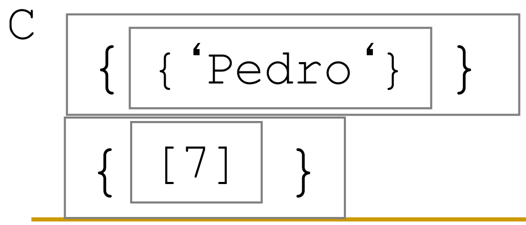
Reading more than one information at a time: textscan





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





```
C{1}{1}  ←  'Pedro'
C{2}(1)  ←  7
```

Example

- You can read a whole file reading more than one piece of data at time

```
vfile = fopen('sentence.txt','rt');
while feof(vfile) == 0
    cdata = textscan (vfile,'%s %d',1);
    fprintf('\n %s scored %d' in the exam, cdata{1}{1}, cdata{2}(1));
end
fclose(vfile);
```

What if I need to keep the whole content of the file in memory?

What if I need to keep the whole content of the file in memory?

You will need to organize the information in a data structure:

- a vector: if there are only numbers
- a vector of structures or a cellarray: if there are strings or something different to just numbers

Example

- If you only read one piece of information of the same type (numbers or characters) you can use a vector

```
marks = []
count = 0;
vfile = fopen(’marks.txt','rt');
while feof(vfile) == 0
     vmark = fscanf (vfile,’%d',1);
    count = count + 1;
    marks[count] = vmark;
end
fclose(vfile);
```

Example

- If you read strings you can’t use vectors, so you could use a vector of structures or a cellarray

```
clear cnames;
count = 0;
vfile = fopen(’names.txt','rt');
while feof(vfile) == 0
    sname = textscan (vfile,’%s',1);
    count = count + 1;
     vst(count).name = sname{1}{1};     or        cnames{count} = sname{1}{1};
end
fclose(vfile);
```

Example

- When you read different types of information the best solution is to use a vector of structuctures

```
 students = struct('name','','mark',0);
count = 0;
vfile = fopen('sentence.txt','rt');
while not(feof(vfile))
    cdata = textscan (vfile,'%s %d',1);
    count = count + 1;
    student(count).name = cdata{1}{1};
    student(count).mark = cdata{2}(1);
end;
fclose(vfile);
```