UNIVERSITÀ
DEGLI STUDI
DI BERGAMO

Informatica
Modulo di Programmazione

Informatica
Modulo di Programmazione
Files

mauro.pelucchi@gmail.com
Mauro Pelucchi

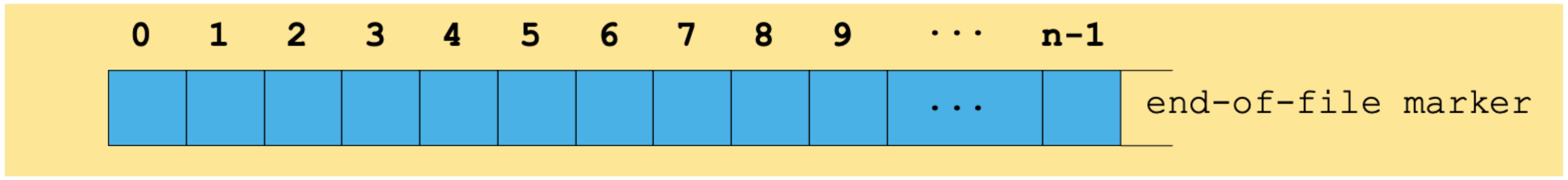
2023/2024

Agenda

- Files

Files and streams

- C++ views file as sequence of bytes
  - Ends with *end-of-file* marker



- When file opened
  - Object created, stream associated with it
  - **`cin`**, **`cout`**, etc. created when **`<iostream>`** included
    - Communication between program and file/device

FStream

- To perform file processing
  - Include **`<iostream>`** and **`<fstream>`**
  - Class templates
    - **`basic_ifstream`** (input)
    - **`basic_ofstream`** (output)
    - **`basic_fstream`** (I/O)
  - **`typedef`**s for specializations that allow **`char`** I/O
    - **`ifstream`** (**`char`** input)
    - **`ofstream`** (**`char`** output)
    - **`fstream`** (**`char`** I/O)

Creating a Sequential-Access File

- C++ imposes no structure on file
  - Concept of "record" must be implemented by programmer
- To open file, create objects
  - Creates "line of communication" from object to file
  - Classes
    - **`ifstream`** (input only)
    - **`ofstream`** (output only)
    - **`fstream`** (I/O)
  - Constructors take *file name* and *file-open mode*

    ```
    ofstream outClientFile( "filename", fileOpenMode );
    ```
  - To attach a file later

    ```
    Ofstream outClientFile;
    outClientFile.open( "filename", fileOpenMode);
    ```

Creating a Sequential-Access File

- File-open modes

Mode	Description
`ios::app`	Write all output to the end of the file.
`ios::ate`	Open a file for output and move to the end of the file (normally used to append data to a file). Data can be written anywhere in the file.
`ios::in`	Open a file for input.
`ios::out`	Open a file for output.
`ios::trunc`	Discard the file's contents if it exists (this is also the default action for **`ios::out`**)
`ios::binary`	Open a file for binary (i.e., non-text) input or output.

  - **`ofstream`** opened for output by default
    - **`ofstream outClientFile( "clients.dat", ios::out );`**
    - **`ofstream outClientFile( "clients.dat");`**

Creating a Sequential-Access File

- Operations
  - Overloaded **`operator!`**
    - **`!outClientFile`**
    - Returns nonzero (true) if **`badbit`** or **`failbit`** set
      - Opened non-existent file for reading, wrong permissions
  - Overloaded **`operator void*`**
    - Converts stream object to pointer
    - **`0`** when when **`failbit`** or **`badbit`** set, otherwise nonzero
      - **`failbit`** set when EOF found
    - **`while ( cin >> myVariable )`**
      - Implicitly converts **`cin`** to pointer
      - Loops until EOF

Creating a Sequential-Access File

- Operations
  - Writing to file (just like **`cout`**)
    - **`outClientFile << myVariable`**
  - Closing file
    - **`outClientFile.close()`**
    - Automatically closed when destructor called

Reading Data from a Sequential-Access File

- Reading files
  - `ifstream inClientFile( "filename", ios::in );`
  - Overloaded `!`
    - `!inClientFile` tests if file was opened properly
  - `operator void*` converts to pointer
    - `while (inClientFile >> myVariable)`
    - Stops when EOF found (gets value `0`)

Esercizio 1

- Scrivere un programma per la gestione di un archivio di videocassette (al massimo 200) Per ogni videocassetta si deve poter memorizzare:
    - <codice cassetta [max 10 caratteri]>
    - <titolo film [max 30 caratteri]>
    - <regista [max 30 caratteri]>
    - <anno di produzione [numero di 4 cifre]>
- Il programma deve proporre dopo aver caricato in memoria i dati presenti su disco un menù che permetta le seguenti operazioni:
    - Inserimento di una nuova cassetta nell'archivio
    - Stampare l'archivio in ordine alfabetico per titolo
    - Salvataggio in un archivio
    - Preparazione di un file HTML per visualizzare una tabella contenente i dati in archivio

Esercizio 2

- Modificare l'esercizio della precedente lezione (Archivio di videocassette) in modo da gestire il tutto utilizzando allocazione dinamica di memoria e liste mantenendo in fase di inserimento i dati sempre ordinati per codice.