

# Programmazione I-B 2020-21

Laboratorio T2

Attilio Fiandrotti

[attilio.fiandrotti@unito.it](mailto:attilio.fiandrotti@unito.it)

1 Ottobre 2020

# Orari del corso

- se (matricola pari) allora T2
- se (matricola dispari) allora T1

Roversi

Fiandrotti

Roversi

## Orario delle lezioni del primo anno del Corso B

Ora	Lun	Mar	Mer	Gio	Ven	Sab
9-10	Prog I B (Aula B)	Log B (Aula B)	Mate Discr B (Aula B)	Prog I B T2 (Laboratorio Turing)	Mate Discr B (Aula B)	
10-11	Prog I B (Aula B)	Log B (Aula B)	Mate Discr B (Aula B)	Prog I B T2 (Laboratorio Turing)	Mate Discr B (Aula B)	
11-12	Log B (Aula B)	Mate Discr B (Aula B)	Prog I B (Aula B)	Prog I B T2 (Laboratorio Turing)		
12-13	Log B (Aula B)	Mate Discr B (Aula B)	Prog I B (Aula B)			
13-14				Prog I B T1 (Laboratorio Turing)		
14-15	RO B (Aula B)	RO B (Aula B)	Ingl I (Aula A)	Prog I B T1 (Laboratorio Turing)		
15-16	RO B (Aula B)	RO B (Aula B)	Ingl I (Aula A)	Prog I B T1 (Laboratorio Turing)		
16-17						
17-18						
18-19						

# Materiale del corso

- Messo a disposizione su Moodle
  - Strumento per didattica a distanza, esami, etc.
- Materiale delle lezioni  
<https://informatica.i-learn.unito.it/course/view.php?id=2033>
- Materiale del turno 1 di laboratorio  
<https://informatica.i-learn.unito.it/course/view.php?id=2035>
- Materiale del turno 2 di laboratorio  
<https://informatica.i-learn.unito.it/course/view.php?id=2071>

# Tipico tema d'esame

- Quattro esercizi
  - 2 esercizi di programmazione
    - Programmazione dicotomica iterativa o ricorsiva
    - Simulazione JVM
  - 1 esercizio semi-teorico
    - Correttezza parziale
  - 1 esercizio teorico

# Organizzazione del Laboratorio

- Un docente, due esercitatori
- Situazione tipica: studente in difficoltà con un esercizio
- Lo studente richiede aiuto «alzando la mano» o via chat
- Lo studente viene spostato temporaneamente in una *breakout room* con un esercitatore (o il docente)
- Risolto il problema, esercitatore e studente tornano alla room del corso

# Obiettivi Odierni

- Il laboratorio odierno ha come obiettivi:
- Gestire programmi e il filesystem utilizzando unicamente la linea di comando
- Scrivere una semplice applicazione Java, compilarla ed eseguirla utilizzando un semplice editor di testo
- Per alcuni di voi questi esercizi saranno banali, per altri saranno una novità: non esitate a domandare

# L'interprete dei Comandi

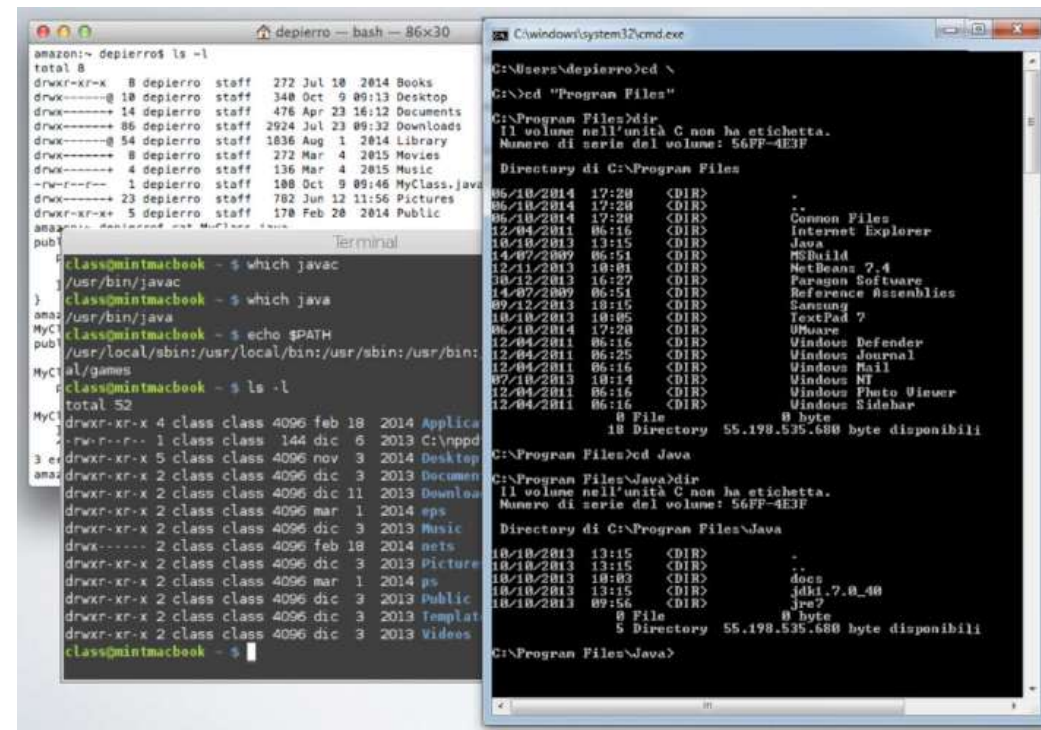
# INTERAGIRE CON IL PC

- Nei primi PC il mouse non esisteva, c'era solo la tastiera
- Spesso lo schermo poteva generare solo caratteri (es: remoto)
- Modalità grafica utilizzata solo da particolari applicazioni con hardware grafico input/output specifico (CAD, etc.)
- L'utente interagiva con il sistema operativo (OS) (es, MS-DOS, UNIX-V) attraverso un'interfaccia testuale detta *shell*, *terminale*, o *command line interface (CLI)*
- Tutte le operazioni avvenivano impartendo *comandi* in forma testuale
- Anche le applicazioni avevano una interfaccia basata su testo



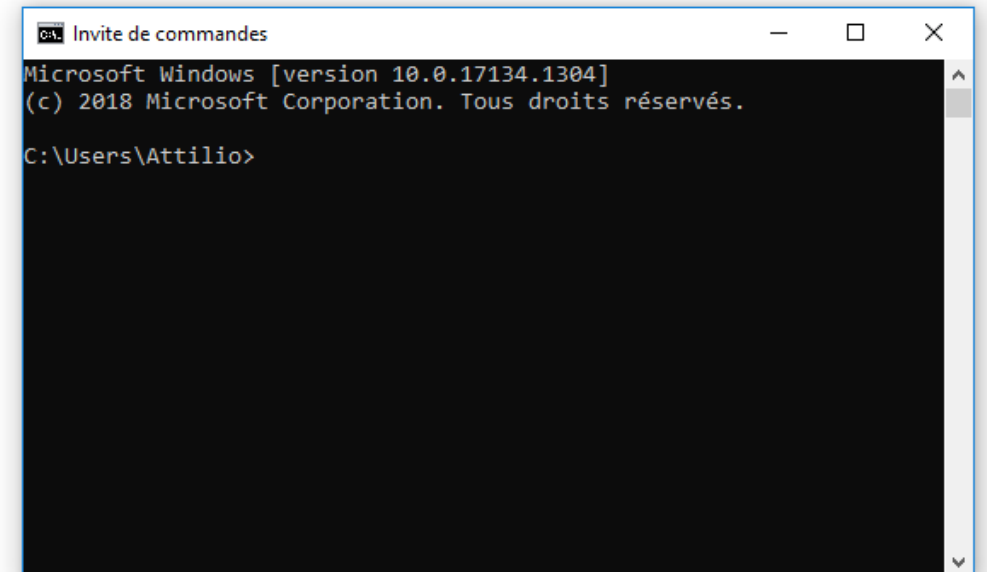
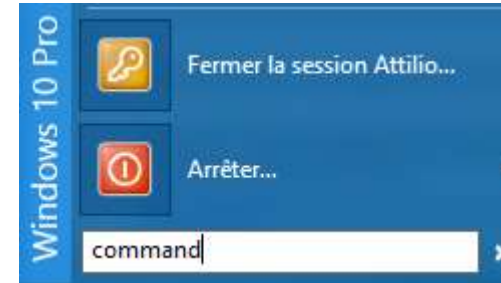
# INTERAGIRE CON IL PC

- Gli OS moderni sono basati su interfacce grafiche
- Conservano la possibilità di interagire attraverso interfacce testuali in finestre
  - MS Windows: “Prompt dei comandi»
  - In Unix (es, MacOS)/Linux, bash, zsh, tcsh, etc.



# INTERAGIRE CON IL PC

- In Windows, fare click su *Start*
- Scrivere *command* in *Ricerca*
- Si aprirà un terminale MS-DOS
- Siete pronti a impartire dei *comandi*
- Windows é *case-insentive*!
- Cosa succede se premo TAB ?



# SINTASSI DI UN COMANDO

- Un comando è composto da un nome che distingue il comando, eventualmente seguito dalle sue opzioni e dagli argomenti

nome-comando [opzioni] <argomenti> [<opzionali>]

- gli argomenti sono tipicamente i dati sui quali il comando opera

MD Prog1

- le opzioni definiscono il modo nei quali un comando può operare

DIR /A /L

- In generale nome-comando / ? per conoscere *sintassi* comando

# FILE NAMING

- Molti comandi lavorano su dati memorizzati in file passati come argomenti del comando :

```
copy <file.txt> <disco USB>
```

- Come indicare univocamente «*Il file file.txt che si trova nella directory Documenti dell'utente Attilio all'interno del disco C:\* »?
- Non è sufficiente scrivere semplicemente il nome del file
- C'è bisogno di una forma testuale per indicare un file nel filesystem
- Questi é il *pathname*, o *percorso*, di un file

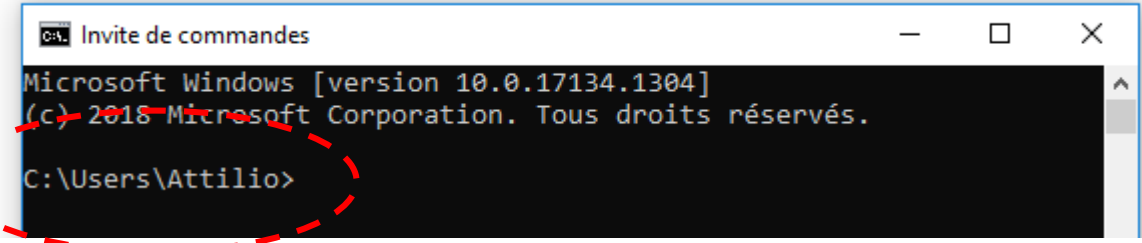
# FILE NAMING ASSOLUTI E RELATIVI

- Un pathname può essere:
- assoluto: è il nome completo di un file nel sistema e si forma componendo in ordine i nomi delle directory da attraversare a partire dalla cartella radice del sistema per arrivare al file separando gli elementi costituenti con il carattere '\' ('/' in Unix).

DEL C:\users\Attilio\Documenti\file.txt

- relativo: lavora unitamente al concetto di cartella di lavoro (*working directory*)

# LA CARTELLA DI LAVORO



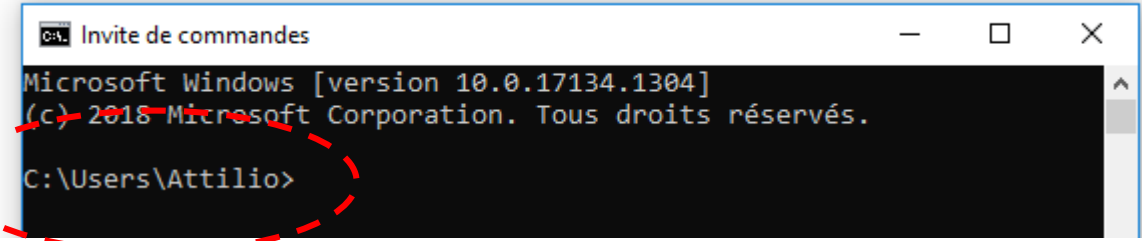
- La cartella di lavoro corrente (working directory) é indicata nel prompt oppure si può conoscere con CD
- la cartella di lavoro è considerata la cartella iniziale dalla quale partire per nominare un file in modalità relativa
- La cartella iniziale è la cartella di lavoro

DEL Documenti\file.txt

in questo esempio equivale a

DEL C:\users\Attilio\Documenti\file.txt

# LA CARTELLA DI LAVORO



- Nel pathame le stringhe ‘.’ e ‘..’ indicano cartelle particolari
- ‘.’ è sinonimo di path corrente

`DEL .\Documenti\file.txt`

equivale a

`DEL C:\users\Attilio\Documenti\file.txt`

- ‘..’ é la directory di livello gerarchico successivo

`DEL C:\users\Attilio\..\Attilio\Documenti\file.txt`

`CD Documenti; DEL ..\Documenti\file.txt`

# ALCUNI COMANDI MS-DOS/WINDOWS

- `cd [C:\users\Attilio\Documenti]`
- `copy file1 [percorso a directory]`
- `del file`
- `dir /a /l`
- `md <percorso a directory>`
- `move file1 file2`
- `ren file1 file2`
- `<command> /?`

<https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/windows-commands>



# ALCUNI COMANDI UNIX (BASH)

- `cd /home/attilio/Documenti/file.txt`
- `cp file1 [percorso a directory]`
- `rm file`
- `ls -a -l`
- `mkdir <percorso a directory>`
- `mv file1 file2`
- `mv file1 file2`
- `<command> --help` **or** `man <command>`

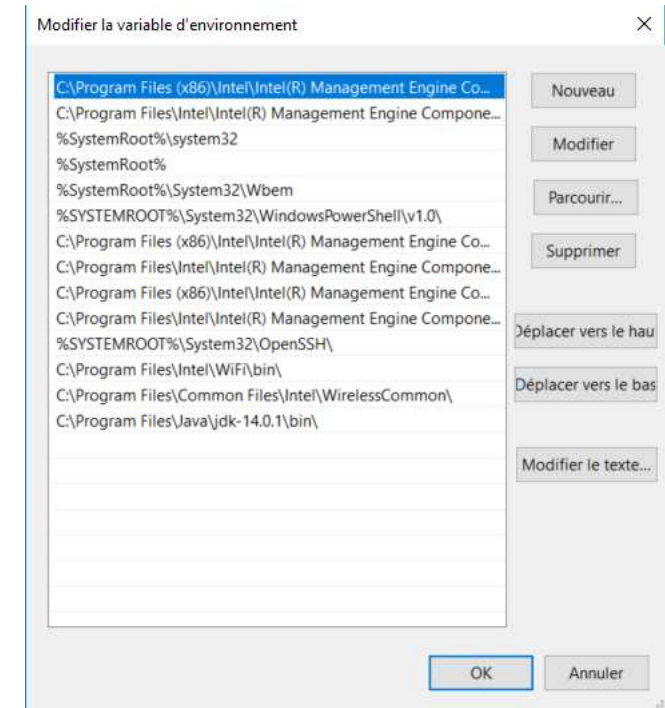
<https://courses.cs.washington.edu/courses/cse391/16sp/bash.html>

# LA VARIABILE DI SISTEMA PATH

- I comandi sono spesso file eseguibili
- Possono essere memorizzati in qualsiasi percorso a patto che siano elencati nella variabile di sistema PATH

ECHO %PATH%

- Separati da «;»
- Il path può essere modificato dal pannello di configurazione



```
C:\Windows\System32>echo %PATH%
C:\Program Files (x86)\Intel\Intel(R) Management Engine Components\iCLS\;C:\Program Files\Intel\Intel(R) Management Engine Components\iCLS\;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\WINDOWS\System32\WindowsPowerShell\v1.0\;C:\Program Files (x86)\Intel\Intel(R) Management Engine Components\DAL;C:\Program Files\Intel\Intel(R) Management Engine Components\DAL;C:\Program Files (x86)\Intel\Intel(R) Management Engine Components\IPT;C:\Program Files\Intel\Intel(R) Management Engine Components\IPT;C:\WINDOWS\System32\OpenSSH\;C:\Program Files\Intel\WiFi\bin\;C:\Program Files\Common Files\Intel\WirelessCommon\;C:\Program Files\Java\jdk-14.0.1\bin\;C:\Users\Attilio\AppData\Local\Microsoft\WindowsApps

C:\Windows\System32>
```

# ESERCIZIO 1

- DA LINEA DI COMANDO:
- Creare nuova cartella PROVA e (con Notepad++) un nuovo file Miaprova
- Rinominare il file in MiaProva1
- Fare una copia di MiaProva1 (usando pathname assoluto e poi relativo)
- Cancellare la copia
- Risalire alla cartella genitore di PROVA
- Passare al disco C: (o W:)
- Rientrare nella cartella PROVA
- Verificare la struttura del filesystem con il file browser

# LE VARIABILI LOCALI in MS-DOS

- E' possibile impostare variabili locali tramite il comando *set*

```
set miaVariabile=«valore»
```

- E' possibile richiedere l'input utente e salvarlo in una variabile (si attende un input dall'utente)

```
set /p var= "Input: "
```

- Il valore della variabile é accessibile come %miaVariabile%

```
ECHO %%miaVariabile%%
```

Permette l'interazione con l'utente via shell

# I FILE BATCH

- E' possibile automatizzare processi inserendo una lista di comandi in un file *batch*
- Un file batch ha estensione .bat e contiene liste di comandi
- Può essere eseguito da terminale come un normale comando
- Spesso si disabilita un aparte dell'input per ragioni di correttezza visiva

```
@echo off
```

# ESERCIZIO 2

- Scrivere un file batch *saluta.bat* che chieda all'utente di inserire il proprio nome, salvi tale informazione in una variabile chiamata *miaVariabile* e infine saluti l'utente con il messaggio

Hello «nome utente»

# ESERCIZIO 2 - SOLUZIONE

- Scrivere un file batch *saluta.bat* che chieda all'utente di inserire il proprio nome, salvi tale informazione in una variabile chiamata *miaVariabile* e infine saluti l'utente con il messaggio

Hello «nome utente»

```
@echo off  
set /p var="Inserire il nome: "  
echo Ciao %var%
```

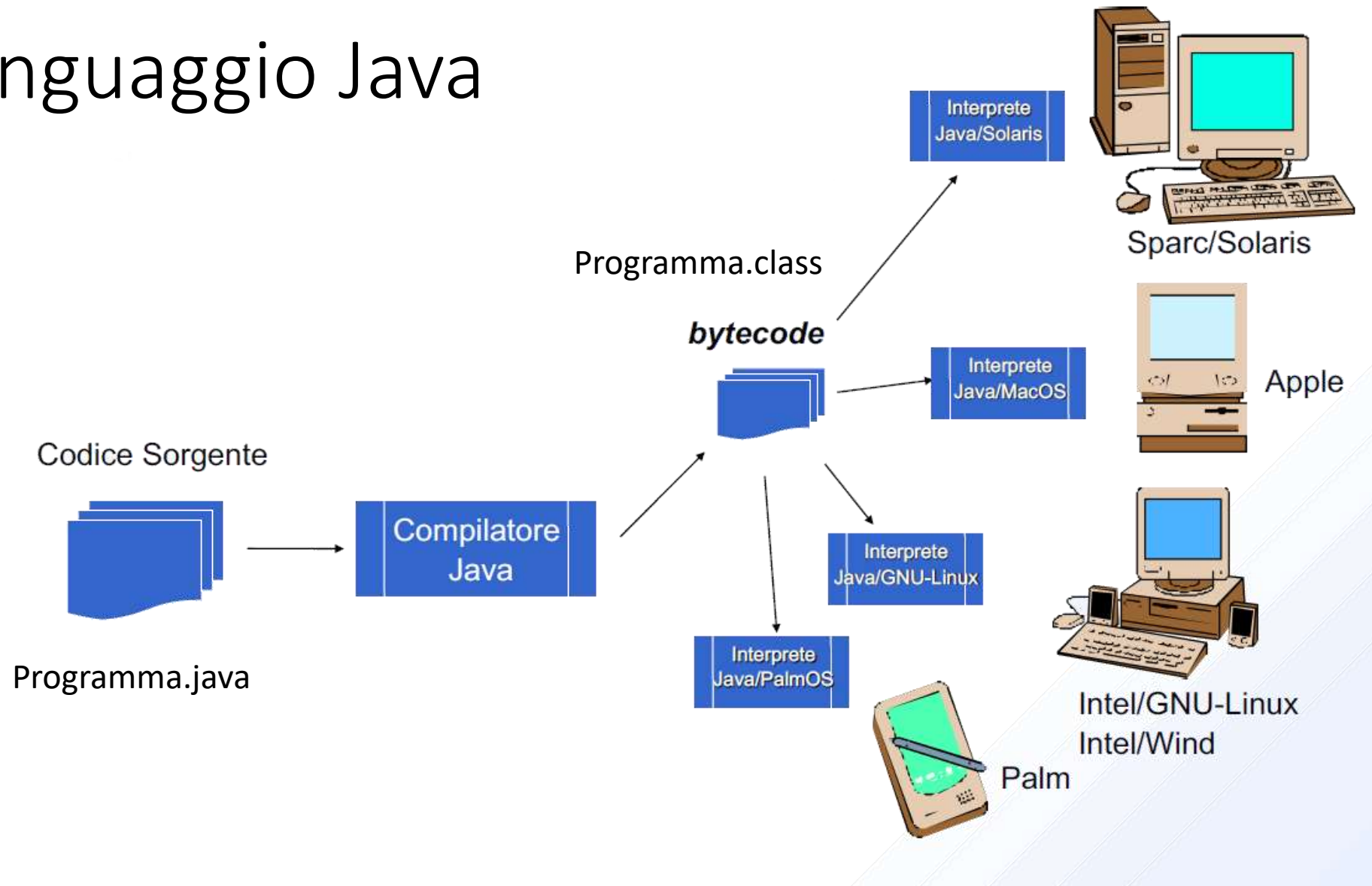
# La programmazione in Java



# Il Linguaggio Java

- Java é un linguaggio interpretato
  - Per contro, C, C++, .. sono linguaggi *compilati*
- Il compilatore java traduce il codice sorgente in bytecode per la Java Virtual Machine (JVM)
- Ogni architettura hardware implementa la propria JVM per il bytecode Java
- Lo stesso bytecode può essere distribuito su più architetture
  - Il binario compilato C/C++ sarà eseguibile solo su una data architettura

# Il Linguaggio Java



# Software per la programmazione in Java

- Installare JDK SE: Java Development Kit Standard Edition. La versione corrente é la 15.

<https://www.oracle.com/java/technologies/javase-jdk15-downloads.html>

- Tipicamente, installato in C:\Program Files\Java\jdk-XX.Y.Z
- Linux: installabile tramite il vostro gestore di pacchetti (apt, rpm,...)

# Prima di utilizzare il JDK in Windows

- Bisogna aggiornare la variabile di sistema CLASSPATH
- Procedura tramite interfaccia grafica (permanente)
  - Click Start, poi Pannello di Controllo, poi Sistema
  - Click Avanzate, poi Variabili d'Ambiente

# Gli Strumenti del JDK

- Tra i file .exe in “bin”:
- **javac** è il *compilatore* di programmi scritti in linguaggio Java. Traduce il programma Java (.java) in *bytecode* (.class)

`javac HelloWorld.java -> HelloWorld.class`

```
Répertoire de C:\Program Files\Java\jdk-14.0.1\bin

27/05/2020 12:46      50 832 java.exe
27/05/2020 12:46      20 112 javac.exe
27/05/2020 12:46      20 112 javadoc.exe
27/05/2020 12:46      20 112 javap.exe
27/05/2020 12:46      50 832 javaw.exe
          5 fichier(s)          162 000 octets
          0 Rép(s) 267 896 008 704 octets libres

C:\Program Files\Java\jdk-14.0.1\bin>
```

# Gli Strumenti del JDK

- Tra i file .exe in “bin”:
- **java** é l'*interprete* Java (é la *Java virtual machine*): traduce bytecode in linguaggio macchina per quel computer particolare

```
java HelloWorld.class      <-  helloWorld.Class
```

```
Répertoire de C:\Program Files\Java\jdk-14.0.1\bin
27/05/2020 12:46      50 832 java.exe
27/05/2020 12:46      20 112 javac.exe
27/05/2020 12:46      20 112 javadoc.exe
27/05/2020 12:46      20 112 javap.exe
27/05/2020 12:46      50 832 javaw.exe
          5 fichier(s)          162 000 octets
          0 Rép(s) 267 896 008 704 octets libres

C:\Program Files\Java\jdk-14.0.1\bin>_
```

# Gli Strumenti del JDK

- Tra i file .exe in “bin”:
- **javap** é il disassemblatore Java: visualizza il bytecode Java in cui il vostro sorgente é stato tradotto

```
javap -c HelloWorld
```

```
Répertoire de C:\Program Files\Java\jdk-14.0.1\bin

27/05/2020 12:46      50 832 java.exe
27/05/2020 12:46      20 112 javac.exe
27/05/2020 12:46      20 112 javadoc.exe
27/05/2020 12:46      20 112 javap.exe
27/05/2020 12:46      50 832 javaw.exe
          5 fichier(s)          162 000 octets
          0 Rép(s) 267 896 008 704 octets libres

C:\Program Files\Java\jdk-14.0.1\bin>
```

# Le API Java

- Le *funzioni* disponibili in Java v.XX sono descritte nel API Specifications

## Java® Platform, Standard Edition & Java Development Kit Version 15 API Specification

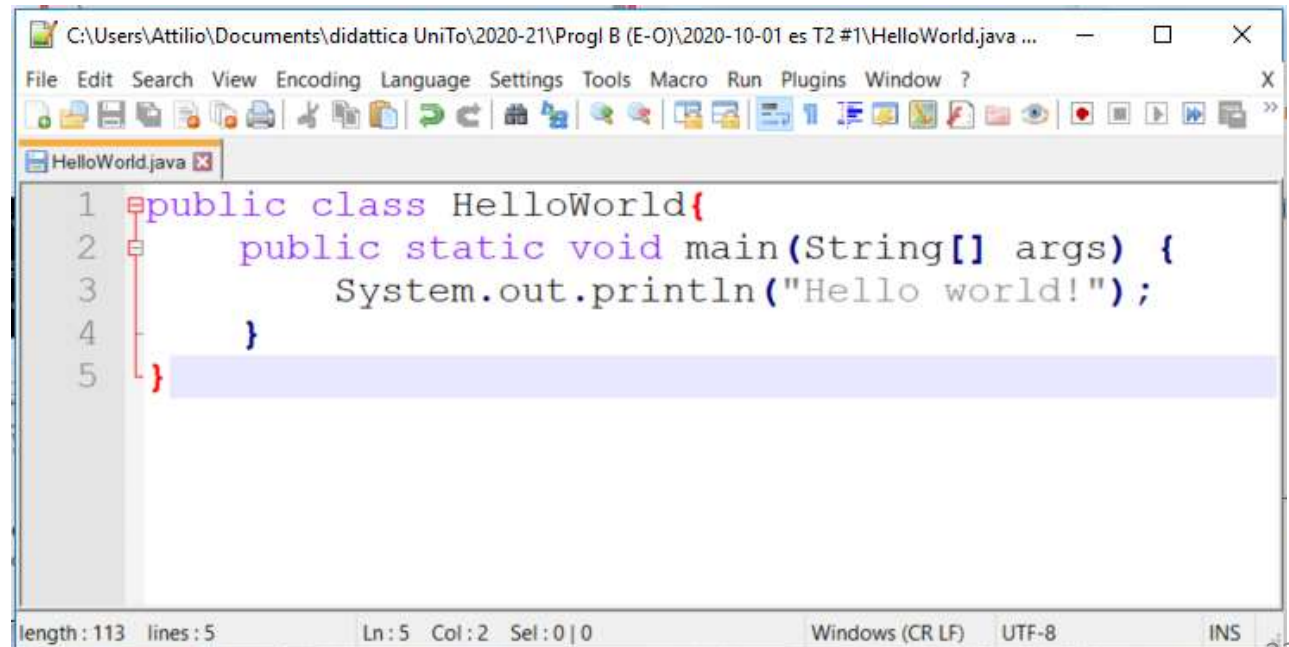
All Modules	Java SE	JDK	Other Modules
Module	Description		
<code>java.base</code>	Defines the foundational APIs of the Java SE Platform.		
<code>java.compiler</code>	Defines the Language Model, Annotation Processing, and Java Compiler APIs.		
<code>java.datatransfer</code>	Defines the API for transferring data between and within applications.		
<code>java.desktop</code>	Defines the AWT and Swing user interface toolkits, plus APIs for accessibility, audio, imaging, printing, and JavaBeans.		
<code>java.instrument</code>	Defines services that allow agents to instrument programs running on the JVM.		

<https://docs.oracle.com/en/java/javase/15/docs/api/index.html>



# L'editor Notepad++

- Editor di testo consigliato per MS Windows
- <https://notepad-plus-plus.org/downloads/>
- Features utili alla programmazione java
  - Syntax highlighting
  - Bracket matching
  - Multiple tabs
  - Unix/MSWindows newline encoding



The screenshot shows the Notepad++ application window. The title bar indicates the file path: C:\Users\Attilio\Documents\didattica UniTo\2020-21\Progl B (E-O)\2020-10-01 es T2 #1\HelloWorld.java ... The menu bar includes File, Edit, Search, View, Encoding, Language, Settings, Tools, Macro, Run, Plugins, Window, and ?. The toolbar contains various icons for file operations and editing. The editor window shows a single tab titled 'HelloWorld.java'. The code is as follows:

```
1 public class HelloWorld{
2     public static void main(String[] args) {
3         System.out.println("Hello world!");
4     }
5 }
```

The code is syntax-highlighted: 'public' is purple, 'class' is blue, 'void' is purple, 'main' is blue, 'String' is blue, 'args' is blue, 'System.out' is blue, 'println' is blue, and 'Hello world!' is in quotes. Red brackets on the left side of the code indicate that the opening and closing braces are matched. The status bar at the bottom shows 'length: 113 lines: 5', 'Ln: 5 Col: 2 Sel: 0 | 0', 'Windows (CR LF)', 'UTF-8', and 'INS'.

# ESERCIZIO 1

- Scriviamo un primo programma *HelloWorld* che stampi a video la scritta «Hello World!», lo compiliamo e lo eseguiamo
- Si usi questo template:

```
public class <...>{  
    public static void main(String[] args) { ... }  
}
```

- Dove salverò il sorgente .java ?
- Quale nome darò alla mia classe Java ?
- Con quale nome salverò il file sorgente *.java* ?

# ESERCIZIO 1 - SOLUZIONE

Il nome della *classe*  
deve corrispondere al  
nome del file

Il metodo *main()* é il  
punto di ingresso nel  
programma

Il metodo *println()*  
stampa una *stringa di*  
*caratteri* a schermo

```
/* Commento su
 * più linee
 */
public class HelloWorld{
    //Commento su una linea sola
    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}
```

# ESERCIZIO 2

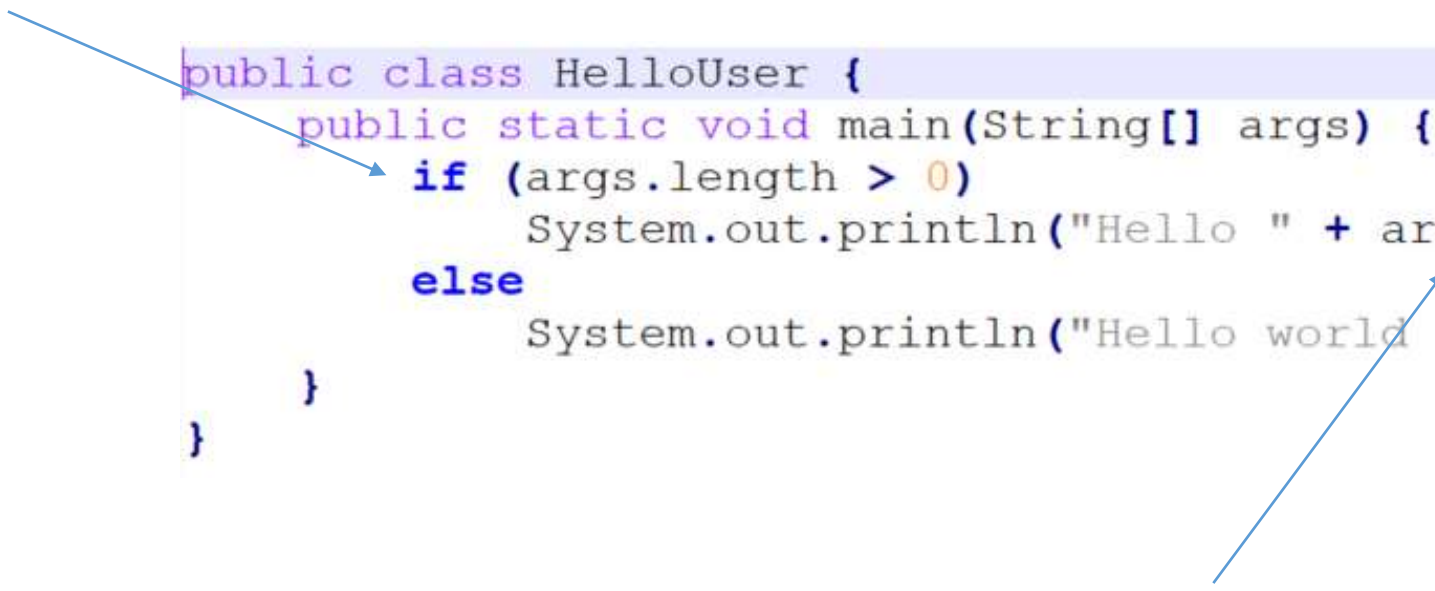
- Modificare il precedente programma *HelloWorld* perché sia invocabile con la seguente sintassi

```
java HelloWorld ["Nome utente"]
```

- Nel caso l'argomento "Nome utente" sia specificato, stampare a schermo «Hello *nome utente* !»
  - Nel caso nessun argomento sia stato specificato, stampare l'output del programma visto in precedenza
- Come verificherò se l'utente ha passato un argomento o meno al programma ?

# ESERCIZIO 2 - SOLUZIONE

Controllo se sono stati passati  
parametri da linea di comando



```
public class HelloUser {  
    public static void main(String[] args) {  
        if (args.length > 0)  
            System.out.println("Hello " + args[0] + " !");  
        else  
            System.out.println("Hello world !");  
    }  
}
```

Stampo il primo degli eventuali  
parametri passati da linea di comando  
(primo elemento di un *array* di stringhe  
di caratteri)

# ESERCIZIO 3

- Scrivere un file batch *HelloWorld.bat* che
  1. chieda all'utente di inserire il proprio nome e salvi tale informazione in una variabile chiamata *mioNome*
  2. invochi il programma java *HelloUser* visto precedentemente passandogli la variabile *mioNome* in modo che *HelloUser* stampi tale nome a schermo
- Come invocare il programma Java dall'interno dello script batch ?
- Come passare il nome utente memorizzato come *mioNome* come argomento da linea di comando di *HelloUser* ?

# ESERCIZIO 3 - SOLUZIONE

Nasconde il prompt dei comandi

Memorizza il nome utente  
nella variabile *mioNome*

```
@echo off  
set /p mioNome="Inserisci il tuo nome: "  
java HelloUser %mioNome%
```

Invoca *HelloUser* passando  
*mioNome* come argomento  
della linea di comando

# ESERCIZIO 4

- Scrivere un programma IncDec che inizializzi la variabile «i» a 0, poi la incrementi a 1 e quindi la decrementi nuovamente a 0
- Disassemblare il bytecode di Inc usando javap

```
javap -c IncDec
```

- Cosa osservo nell'output del programma e come posso ricondurre tale output al codice sorgente ?