TEMA

Matlab for Math

Tra matematica e coding… per appassionare i ragazzi!

Lorenzo Dursi

AICA – Supervisore ECDL

lorenzo.1796@gmail.com

keywords: *: coding, matematica, programmazione, Matlab, competenze digitali, cooperative learning, problem posing, problem solving, pensiero computazionale*

Introduzione

L'approvazione della Legge 107/2015, nota come "La Buona Scuola", ha portato con sé il Piano Nazionale Scuola Digitale, "*documento di indirizzo del Ministero dell'Istruzione, dell'Università e della Ricerca per il lancio di una strategia complessiva di innovazione della scuola italiana e per un nuovo posizionamento del suo sistema educativo nell'era digitale"*.

In quest'ottica di innovazione, che ha dato vita alla nuova figura dell'Animatore Digitale, ai progetti di digitalizzazione di aule e segreterie e a nuove competenze per docenti e studenti, si inseriscono la programmazione ed il *coding*.

Il contesto: le competenze digitali

Cosa sono queste competenze digitali? Come ci aspettiamo, indicano la capacità di un individuo di saper utilizzare bene e con spirito critico le nuove tecnologie in campo domestico, lavorativo, comunicativo,… in ogni aspetto della propria vita. Per tale scopo, ovviamente, è necessaria:

- una buona conoscenza del computer, dagli aspetti più elementari a quelli più articolati (*computer literacy*);
- una capacità critica di ricerca, rielaborazione e produzione di informazioni o contenuti (*information literacy*);

- una conoscenza riguardo i network (le reti), in un mondo sempre più interconnesso (*network literacy*). A titolo di esempio, non possiamo dimenticare come oggi si parli di *IoT*, *Internet of Things*, "Internet delle cose", anzi, come sostiene qualche azienda, di *Internet of every-Things*, "Internet di ogni cosa". La rete (in questo caso, Internet) è ormai presente in ogni ambito della nostra vita e, in quanto tale, non può più essere allontanata ma compresa e integrata nella quotidianità.

L'idea è quella di favorire e sviluppare sia la cittadinanza digitale (ogni individuo della società deve conoscere, comprendere e saper usare le tecnologie) sia l'inclusione digitale (ogni persona deve avere libero accesso alle tecnologie e deve utilizzarle criticamente).

Con questi temi quali faro, tramite il documento *DigComp*, l'Unione Europea ha fornito all'intera comunità internazionale un quadro di riferimento delle competenze digitali per i cittadini. Il documento sviluppa le 3 *literacy* precedenti in 5 aree di competenze:

1. alfabetizzazione su informazioni e dati;
2. comunicazione e collaborazione;
3. creazione di contenuti digitali;
4. sicurezza;
5. problem solving.

A cavallo tra la terza e la quinta area di competenza, è contemplato il coding.

Coding, programmazione e pensiero computazionale

Quante volte, nei vari corsi di aggiornamento, abbiamo sentito la frase *"Il coding sviluppa il pensiero computazionale"*? Ma perché? Cosa sottende questa espressione?

Con *programmazione* intendiamo la scrittura di istruzioni che consentano al PC di svolgere le operazioni desiderate. Ovviamente, non possiamo scrivere i codici nel linguaggio specifico del PC (detto linguaggio macchina) in quanto estremamente complesso. Allora, necessitiamo di un "interprete", cioè di un programma che traduca le nostre istruzioni (scritte in un linguaggio detto di programmazione o di alto livello) in linguaggio macchina. Questo lo fanno alcuni programmi, come Matlab o Python.

L'espressione *coding* ha lo stesso significato del termine precedente, anche se, nei corsi di aggiornamento per i docenti, solitamente viene utilizzato per indicare un tipo di programmazione mediata da software che stimolino la creatività, tipo Scratch. Il desiderio è quello che gli alunni, prevalentemente di scuola primaria, possano imparare le strutture di controllo (sequenza, selezione e iterazione) che sono alla base della programmazione, col gioco, divertendosi, creando delle situazioni a loro congeniali per favorire l'apprendimento (si veda, a tal proposito, la teoria del costruzionismo). A titolo d'esempio, nei primi anni di scuola elementare si insegnano ai bambini le quattro operazioni fondamentali con l'aiuto di un animale che salta sulla retta naturale orientata fino a raggiungere il risultato. Allora, sfruttando Scratch, perché non realizzare questo giochetto al PC?

Riflettendoci, mediante questa strategia e fornendo esercizi man mano più complessi, aiutiamo i bambini a sviluppare il *pensiero computazionale*: posto loro un problema, lo suddividono in problemi più semplici, per i quali realizzano dei diagrammi di flusso (*flow-chart*) da implementare poi al PC, con i software scelti. Così facendo, si risolve il problema principale con l'ausilio del calcolatore.

Dunque, si può osservare che il pensiero computazionale risulta essere la sintesi tra *problem solving* e *programmazione*.

L'esperimento: "Matlab for Math"

Ora voglio proporre un'attività che intreccia programmazione e matematica.

Si prende a campione una classe di quinta superiore e si introducono i ragazzi alla programmazione mediante un software ampiamente usato in ambito universitario, Matlab.

La scelta di questa applicazione è motivata dal fatto che, imparando ad utilizzarlo, i ragazzi possano trovarsi avvantaggiati in sede universitaria e che, al contempo, sia occasione per attività di *problem posing and solving*.

Brevemente, presento una scaletta del progetto, che poi analizzeremo.

a) Organizzazione

Si predispongono 5 incontri pomeridiani da 2 ore e mezza, in cui spiegare ai ragazzi come utilizzare Matlab. La restante parte del progetto è svolta in classe, durante alcune ore curriculari di matematica, applicando le nuove conoscenze informatiche a quelle matematiche.

b) Materiale laboratoriale

Si impiega un laboratorio informatico con un numero di postazioni tale che ogni alunno abbia un PC e con tutti i calcolatori collegati a quello del docente. L'ideale sarebbe se l'istituto scolastico fosse dotato di un'Aula 3.0, in modo da integrare nel corso anche il *cooperative learning*. Si fa comprare una multi-licenza Matlab dalla segreteria della scuola (mediante presentazione di un progetto PON o altro) e poi si installa il programma sui PC, con l'ausilio del tecnico.

Osservazione: se la scuola non potesse acquistare il materiale, si può utilizzare il software gratuito Python in luogo di Matlab.

c) Materiale didattico

Si possono predisporre slide o realizzare video (ad esempio, tramite PowToon) oppure far uso della mole di materiale presente nel web.

Entriamo nello specifico.

I temi introduttivi

Si suddividono le due ore e mezza di ciascuno dei cinque incontri nel seguente modo:

- un'ora di spiegazione;
- break di 10 minuti;
- attività pratica (esercizi manuali e/o al calcolatore, dibattiti).

I temi da trattare sono quelli fondamentali per poter utilizzare Matlab, raggruppabili in due macro-aree:

1. area informatica
   1.1. gli algoritmi e i flow-chart;
   1.2. i cicli if, for e while. Teorema di Böhm-Jacopini;
   1.3. il concetto di variabile e di assegnazione;
   1.4. Matlab: Matrix Laboratory.
2. area matematica
   2.1. le matrici: definizione ed operazioni;
   2.2. i vettori: definizione ed operazioni.

I *flow-chart* (o diagrammi di flusso), data la loro immediatezza, si introducono direttamente mediante esercizi ed esempi rintracciabili in rete. Possono essere realizzati su carta oppure sfruttando alcuni programmi *freemium* online, come Creately.

Con tali diagrammi, si può far osservare ai ragazzi che anche noi siamo dei "calcolatori", cioè eseguendo le istruzioni possiamo giungere alla soluzione dell'algoritmo, come fanno i computer. Ovviamente, il nostro è un linguaggio naturale, mentre quello del PC è un linguaggio più complesso, un linguaggio macchina!

Per presentare i successivi argomenti, si possono adoperare i seguenti due video, da me realizzati su Powtoon.

- Video1: Introduzione alle strutture di controllo.
- Video2: Intro to Matlab.

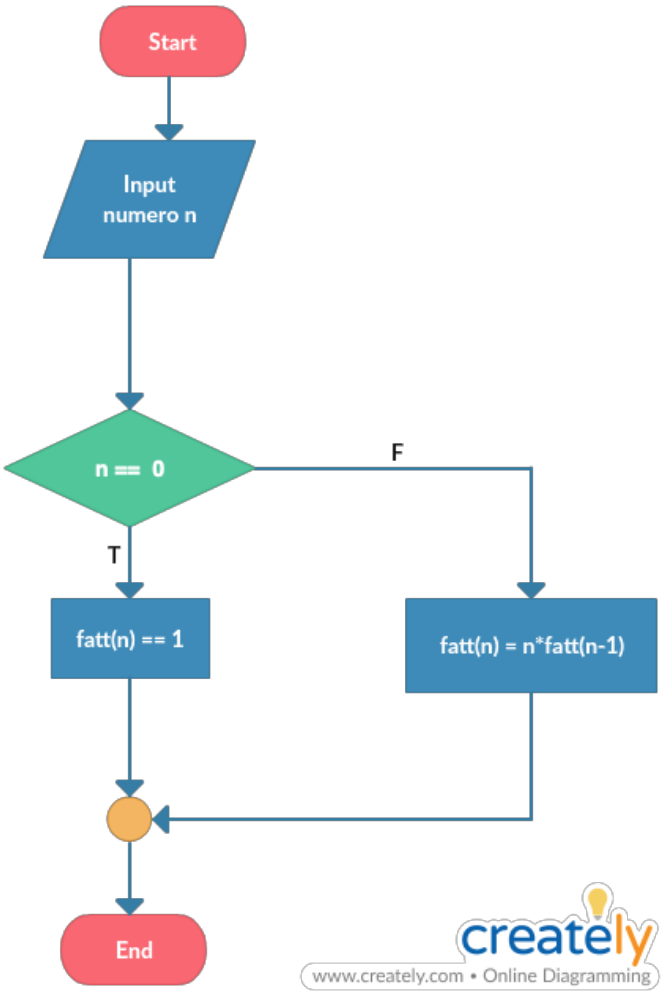

Figura 1 - Flow-chart per il calcolo ricorsivo del fattoriale di un numero.

La visione dei due video è poi approfondita dal docente. L'idea è che non si debba scendere nello specifico degli argomenti ma fornire ai ragazzi quegli elementi utili per concretizzare, dal punto di vista informatico, quelle che saranno le loro intuizioni nel momento in cui dovranno realizzare (semplici) algoritmi su Matlab.

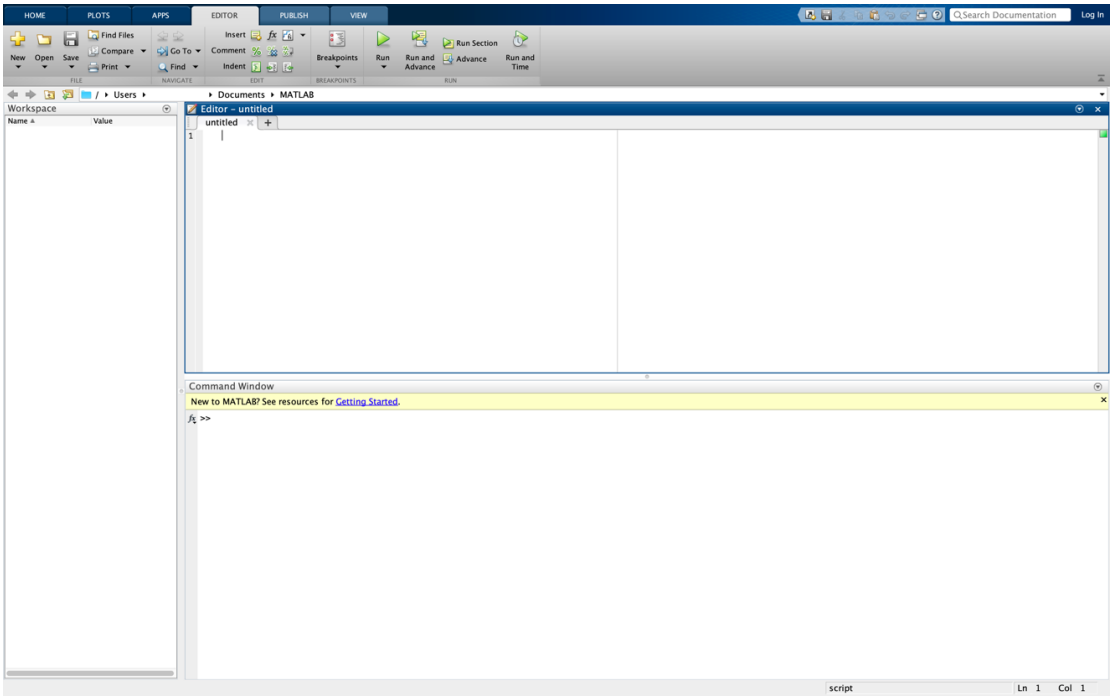

Figura 2 - Schermata iniziale di Matlab.

Come preannunciato, dopo la parte teorica, vi sono degli esercizi da svolgere in aula. Il materiale può essere predisposto dai docenti oppure ricercato in Internet. Ovviamente, l'operazione di ricerca non va effettuata durante la lezione pomeridiana o nei 10 minuti di *break* ma precedentemente, per garantire un'ottima qualità delle risorse individuate. Inoltre, devono essere esercizi che integrino teoria con la pratica. Ad esempio, come primi programmi in Matlab, si può pensare all'implementazione degli algoritmi delle 4 operazioni fondamentali con delle varianti. Pensando alla sottrazione, si può chiedere agli studenti di supporre di lavorare con i numeri naturali e quindi di distinguere i casi in cui sia possibile eseguire tale operazione oppure no (*implementazione del ciclo if*). Un altro esempio può essere l'algoritmo babilonese per l'approssimazione di $\sqrt{2}$, la cui formula è rintracciabile online (*implementazione del ciclo while*). Di esercizi di tale tipologia ve ne sono tantissimi (formula per la soluzione di equazioni di secondo o terzo grado, calcolo dei punti fondamentali delle coniche – raggi, fuochi, vertici, semiassi, …).

Tutto sta (anche *e/o* soprattutto) nella fantasia del docente!

Appena i ragazzi diventano più esperti con l'utilizzo del programma (si può chieder loro di far pratica a casa utilizzando Python), possiamo passare ai temi propri di matematica del quinto anno di scuola superiore.

I temi di matematica

Generalmente, il quinto anno si introducono i temi base dell'analisi matematica moderna (limiti, derivate, integrali) e a tutti i docenti sarà sicuramente capitato di dover spiegare l'approssimazione di integrali che non ammettono primitiva in forma chiusa. Pensiamo alla formula del rettangolo, del trapezio, di Simpson (semplici o composte). Durante le spiegazioni, si sarà notata la scarsa attenzione, a volte, dei ragazzi, che percepiscono l'argomento come noioso, fine a se stesso e pieno di

formule strane da ricordare a memoria. Questo perché, per cogliere la potenza di tali formule, è necessario implementarle al calcolatore, in Matlab.

Allora procediamo nel seguente modo.

Dopo aver enunciato le formule di approssimazione, partendo da quelle semplici fino a quelle composte, chiediamo inizialmente ai ragazzi di realizzare possibili *flow-chart* per ciascuna formula e poi scegliamo quello corretto. Invitiamo i ragazzi a collaborare tra di loro, come dovranno fare in Università (*può essere anche questa un'attività di orientamento?*).

Scelto il diagramma di flusso, lo implementiamo su Matlab e sfruttiamo questo *script* per il calcolo di integrali presi dal libro di testo o ricercabili in rete. Mostriamo ai ragazzi che effettivamente il calcolatore approssima il valore degli integrali, confrontando quest'ultimo risultato con quello effettivo, calcolabile su Matlab tramite la funzione (già implementata) *integral* (scrivendo su Matlab, nel *Command Window*, *help integral*, è possibile vedere la sintassi e come opera tale *function*).

Con questo approccio, gli studenti non sono più passivi memorizzatori di formule astratte, ma entrano attivamente nel processo di apprendimento, rendendosi conto di come le formule vengano effettivamente utilizzate (per la seria *"A che cosa serve la matematica?"*), e vengono anche stuzzicati da tale approccio: scommetto che la maggior parte della classe, terminata la lezione, tornerà a casa per implementare al PC un suo personale *script*, di cui poi si vanterà il giorno dopo!

Oltre agli integrali, un altro argomento può essere la ricerca degli zeri di una funzione. Alcuni docenti lo affrontano in seconda superiore, altri in quinta, dopo aver enunciato il teorema di esistenza degli zeri di Bolzano. Io consiglio quest'ultimo caso, perché più completo e adatto all'implementazione.

Il metodo di ricerca degli zeri basato su tale teorema è noto come metodo delle successive bisezioni: dato un intervallo $[a, b]$, si controlla prima che nell'intervallo dato vi sia uno zero (in base all'ipotesi del teorema) e, in caso affermativo, si procede dimezzando via via l'intervallo di partenza fino ad ottenere un'approssimazione dello zero.

Si può, poi, richiedere ai ragazzi di lavorare in gruppo e cercare altre formule per la ricerca degli zeri (metodo di Newton, delle corde, delle secanti, …), scrivere i *flow-chart* e implementare gli algoritmi al PC. Una volta conclusi i lavori, mediante degli esempi, insieme confrontiamo i risultati ottenuti con i vari metodi da un punto di vista delle cifre significative corrette o del numero di iterazioni necessarie per giungere alla soluzione.

Con tale approccio, si portano i ragazzi a ragionare criticamente sui programmi e sugli algoritmi, come fa solitamente chi lavora nell'ambito dell'Analisi Numerica.

Può essere questa occasione per appassionare qualcuno di loro alla matematica e ai suoi aspetti applicativi?

Ai posteri l'ardua sentenza!