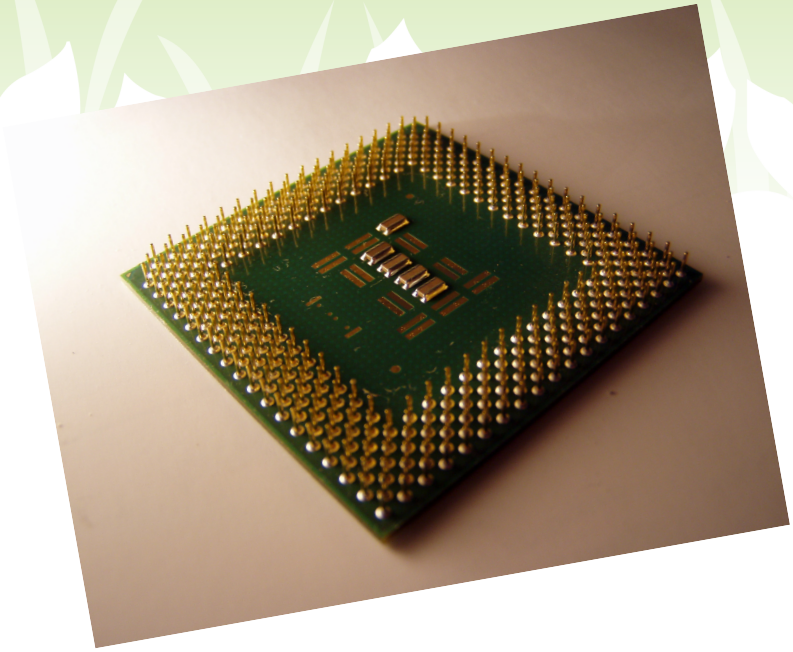
Scenari digitali

Coding unplugged

Materiale connesso al P.N.S.D di Istituto
A cura di Raffaele Ruggiero



Sommario

Il pensiero computazionale nella didattica

Negli ultimi anni il MIUR ha fortemente spinto per l'introduzione di "coding e pensiero computazionale" nella didattica. Proviamo ad approfondire il tema, mostrando come metterlo in pratica con mezzi semplici, partendo dalla modalità 'unplugged'.

CONTESTO

P.N.S.D.
Il programma del M.I.U.R. per lo sviluppo della cultura digitale nella scuola

P.T.O.F.
Piano Triennale dell'Offerta Formativa

Animatore digitale e Team per l'innovazione
Per coadiuvare l'innovazione tecnologica

I.C.S. "Ragazzi d'Europa"
Via E. De Filippo, 80013,
Casalnuovo di Napoli
Tel.: 0815224248
Email: naic84300v@istruzione.it
Sito: www.icragazzideuropa.it

Una delle richieste più importanti emerse dal Questionario docenti sulle T.I.C. presentato in Collegio dei docenti a fine anno scolastico scorso, è stata l'introduzione del "Coding" nell'offerta formativa dell'Istituto.

Questo è il primo tutorial sull'argomento. Esso mira a fornire semplici strumenti operativi, per iniziare subito con attività divertenti e per le quali non è neanche necessario hardware.

Che cos'è il "Coding"?

La parola "*Coding*" è una forma sostantivata del verbo inglese '*to code*', che significa '*codificare*', 'mettere in codice'. Direi, però, che la traduzione più esplicativa, nel nostro contesto, è '*programmare*', nel senso informatico del termine: cioè produrre '*algoritmi*'.

A sua volta, la parola *algoritmo* indica un insieme di istruzioni e di passaggi logici che servono a risolvere un problema, a soddisfare una necessità di natura pratica. I passaggi logici possono essere trasmessi, sotto forma di comandi, a un computer, sfruttando la sua potenza di calcolo per svolgerli in maniera velocissima. La parola deriva da *al-Khuwārizmī*, nome del matematico arabo *Muḥammad ibn Mūsa* del IX secolo.

Bisogna considerare che un algoritmo è indipendente dalla macchina che lo esegue. Esso è frutto della mente umana e quindi ha alcune caratteristiche importanti: 1) è creativo; 2) può essere realizzato anche senza un computer; 3) non è esclusivo (nel senso che per soddisfare una necessità e risolvere un problema, possono esserci più percorsi logici possibili, anche se magari uno di essi è più 'elegante' - a volte geniale - degli altri).

Mi diverte riportare qui due citazioni di due miei conoscenti di molti anni fa, entrambi ingegneri elettronici. Il primo, che lavorava in una multinazionale dell'hardware, mi disse un giorno: "In fondo io mi sono laureato nell'uso di una macchina: è come se mi fossi laureato nell'uso di una caffettiera elettrica e probabilmente una caffettiera mi emozionerebbe di più". Il secondo, docente di Informatica, mi riportò questa frase: "Il più intelligente dei computer è più sciocco del più sciocco degli uomini".

Entrambi sorrisero, dicendo queste frasi...

In effetti, un computer è una macchina priva di fantasia e di autonomia. È solo veloce; dannatamente veloce! Ma tutto quello che fa deve essergli 'imboccato' da un essere umano, perché quest'ultimo ha una dote che manca anche alla macchina più potente: la creatività. Contrariamente a quanto per istinto si potrebbe pensare, un buon informatico è prima di tutto una persona ricca di fantasia; una fantasia che deve sposare la logica. E questo è quello che tocca ai docenti: provare a fare in modo che gli alunni riescano a dosare, sapientemente, la creatività e il rigore formale del ragionamento.

Esistono - ormai è per me fuori di dubbio - due modi di vivere il digitale:

- ✓ Un modo passivo, che consiste nell'usare i dispositivi elettronici e le applicazioni per ottenere risultati immediati in maniera acritica, talora compulsiva, più che altro per divertirsi e trovare informazioni senza alcuna fatica, accettandole per

date senza abitudine a verificarle. Si tratta di un modo purtroppo molto diffuso tra i ragazzi (e anche tra gli adulti).

- ✓ Un modo attivo, che consiste nel cercare di capire cosa c'è dietro lo schermo magico di uno smartphone e cosa è vero e cosa no, nell'immane quantità di dati presenti in Rete.

Un utente attivo è quello che prova, almeno in parte, ad 'addomesticare' l'informatica alle proprie necessità, che non cede alla tentazione della dipendenza, che si pone domande, che considera la propria mente più forte e potente di un computer, che ha gli strumenti (principalmente psicologici) per vivere l'epoca digitale in maniera positiva e stimolante.

E che ogni tanto riesce a spegnere il telefono...

Bene: diciamo che il Coding è uno degli strumenti (non il primo, né l'unico e magari neanche il più importante, per carità) con il quale la scuola prova a far prevalere nei ragazzi il secondo atteggiamento, rispetto al primo.

Prima di andare oltre, facciamo qualche osservazione propedeutica:

1) Il Coding non è di pertinenza dei docenti di una determinata disciplina: la materia più vicina è sicuramente la Matematica ed è giusto pensare che già molti colleghi di Matematica, probabilmente tutti, pratichino forme di Coding, anche senza chiamarlo così. Allo stesso modo, è ragionevole pensare che molti docenti di Tecnologia tengano alle competenze digitali in modo tale da stimolare gli alunni all'utilizzo, più o meno avanzato e consapevole, del software. Il Coding, però, appartiene alla comunità educante nel suo complesso. E se è vero che la scuola copre già lo sviluppo logico nell'età evolutiva con insegnamenti curricolari 'tradizionali', è anche vero che la comunità scientifica internazionale ha intravisto nelle società avanzate un 'vuoto' che va coperto in qualche modo: quel vuoto è proprio nel rapporto troppo spesso sbilanciato e 'passivo' tra la popolazione e gli strumenti digitali.

2) Per praticare attività di Coding, non servono subito i computer. Esiste una **modalità 'unplugged', cioè 'disconnessa', che non si serve di dispositivi digitali, ma di strumenti semplicissimi ed economici**.

3) **Il Coding ha una caratteristica che l'insegnamento dell'informatica non ha: è divertente**. Lo è per i ragazzi, ma - perché no? - anche per i docenti. La dimensione ludica pervade le attività di questo tipo, sposandosi con quella formativa.

Un'ultima precisazione, di carattere generale: **praticare Coding non significa insegnare Informatica**. Quest'ultima si studia negli istituti appositi, usando linguaggi potenti e piuttosto complessi. Diciamo che il Coding viene prima dello studio dell'Informatica e serve ad allenare la mente alla logica, non a creare dei programmi.

Dai processi della mente ai "linguaggi macchina" (andata e ritorno).

Prima di andare oltre, occorre anche un breve excursus sul concetto di programmazione. Ribadisco che il Coding non è l'insegnamento dell'Informatica, ma visto che questo tutorial è introduttivo occorre toccare anche questo aspetto.

Gli elaboratori eseguono delle operazioni espresse in un linguaggio detto 'binario', perché composto di informazioni organizzate nel più semplice sistema numerico possibile, che viene trasmetto al calcolatore tramite pieni e vuoti elettrici. In termini matematici, essi vengono simboleggiati da 0 e 1. Per 'parlare' con un computer, quindi, occorre esprimersi con una serie lunghissima di sequenze numeriche, organizzate in byte. Uno 0 o un 1 corrispondono a un 'bit'; otto bit corrispondono a un 'byte'. I linguaggi composti da queste sequenze numeriche vengono chiamati "linguaggi macchina". Vengono anche detti "di basso livello", perché parlano direttamente al processore di un computer.

Cercare di organizzare un programma usando queste sequenze è operazione decisamente difficile, riservata a tecnici superspecializzati.

Data la sua difficoltà, per diffondere l'informatica, nel tempo sono stati elaborati linguaggi detti "di alto livello". Anche questi sono piuttosto complessi, per la verità, ma sono stati pensati in modo da rendere più facile ai programmatori lo sviluppo di software. Essi sono fatti di parole inglesi (if, when, else...) e di simboli matematici (>, <, *, +, =, ==...). I simboli e le parole sono decisamente più facili da gestire, dal punto di vista semantico, di una sequenza infinita di zeri e di uno e possiamo, quindi, dire che tali linguaggi, orientati più al programmatore che al processore, in qualche modo svolgono la funzione di 'traduttori' tra essere umano e macchina.

Guardiamo il seguente esempio:

```
if (T > 100) {
  alert("Butta la pasta");
}
else {
  alert("Pazienta, ci vuole ancora tempo per mangiare");
}
```

Questa è una fantasiosa sequenza di istruzioni, scritta in un linguaggio che si chiama Javascript. Sebbene fantasiosa, però, essa è sintatticamente corretta e - almeno in parte - facile da capire. T è la temperatura dell'acqua e l'istruzione "alert" ordina al computer di aprire una finestrella con la frase scritta tra doppi apici: per il resto mi sembra tutto intuitivo, no?

Va da sé, però, che anche i linguaggi di alto livello hanno due problemi:

1. per quanto più semplici del linguaggio macchina, sono comunque difficili da imparare, soprattutto quando, invece di buttare la pasta, il problema da risolvere diventa più complesso;
2. essi sono destinati alla programmazione di un software per un elaboratore.

Entrambe queste caratteristiche li rendono poco adatti all'apprendimento nell'infanzia a nella prima adolescenza, età in cui, più che programmare, occorre sviluppare capacità logiche.

Per questo motivo è stato fatto, negli ultimi anni, un ulteriore passo avanti, creando un livello ancora superiore. Le istruzioni dei linguaggi di programmazione sono state organizzate in blocchi colorati. Essi si possono assemblare, sovrapporre, incastrare l'uno nell'altro, a creare percorsi logici anche di una certa complessità, ma comunque molto più facili da gestire rispetto alle istruzioni verbali. Ecco, per esempio, una

schermata tratta da Code.org, un sito di riferimento del Coding:. In questo sito è stato emulato un sistema visuale di programmazione per principianti ideato da un docente del MIT di Boston (ebbene sì, sempre lì accadono le cose importanti...). Il sistema si chiama MIT App Inventor e serve, appunto, a creare applicazioni.

Code.org riproduce, in maniera molto simile, quel sistema:

Come si vede, a sinistra c'è l'omino di un famoso videogioco. I ragazzi, invece di giocarci e basta, come si fa con un touch o un joystick, devono creare dei percorsi usando blocchi visuali di istruzioni, come si vedono a destra. A ogni blocco corrispondono righe di codice. Ma di questo parleremo in altri tutorial. In questo momento interessa capire che nel passaggio dal linguaggio macchina al linguaggio visuale a blocchi si è saliti a un livello in cui anche un preadolescente può creare, divertendosi, degli algoritmi.

Le macchine non danno niente per scontato

Il problema più grosso quando si approccia il pensiero computazionale è che gli esseri umani sono abituati a comunicare con altri esseri umani; al limite con animali, per lo più domestici, come un cane.

E in tutti questi casi ci sono sempre informazioni 'contestuali' che vengono date per scontate.

Con le macchine, invece, bisogna sempre esplicitare tutto, con il massimo rigore, con una consequenzialità e una completezza che finiscono col diventare stucchevoli. E in questo fastidio consiste la fatica del programmare.

Facciamo un esempio per capirci.

Immaginiamo di avere ospiti in casa e di offrire loro il caffè in salotto. A un certo punto, ci rendiamo conto di aver dimenticato lo zucchero in cucina. Perciò chiediamo a nostro fratello di andare a prenderlo.

La frase sarà più o meno la seguente:

"*Per favore, vai di là a prendere lo zucchero.*"

Non ci sarà bisogno di dire altro, perché nostro fratello, che vive nella stessa casa o comunque la conosce bene, sa che deve andare in cucina, aprire un dato sportello della dispensa, prendere un certo contenitore, di colore e forma a lui noti.

Ora, immaginiamo che nostro fratello sia in vacanza e che l'ospite voglia essere gentile e insista per andare a prendere lui lo zucchero. La frase, a quel punto, dovrà essere ben diversa:

"*Per favore, vai in cucina, guarda nella dispensa accanto alla finestra, ma non la dispensa di quercia, no: quella laccata bianca; apri lo sportello in basso a sinistra; dentro ci sono una serie di barattoli di colore diverso: quello giallo contiene il sale; in quello blu c'è lo zucchero. Se vuoi quello di canna, però, ci sono delle bustine monodose nella scatolina di tessuto bianco, lì accanto.*"

Abbiamo fatto molta più fatica. Sarebbe stato meglio andarci di persona a prendere 'sto benedetto zucchero, no?... :)

Ma il peggio deve ancora venire! Ora, infatti, immaginiamo di avere come maggiordomo un robot. A lui, che è una macchina, dovremmo parlare, più o meno, così:

1. *Gira di 180 gradi*
2. *Fai cinque passi*
3. *Gira a sinistra di 90 gradi in corrispondenza della porta*
4. *Quando sei nel corridoio gira a destra di 90 gradi e fai otto passi*
5. *Gira di nuovo a sinistra di 90 gradi, in corrispondenza della porta della cucina e fai due passi in avanti*
6. *Gira a destra di 90 gradi e fai due passi*
7. *Piega il busto di 45 gradi in avanti, poi alza il braccio destro, apri la mano e afferra la maniglia della dispensa bianca*
8. *Tira il braccio all'indietro per aprire lo sportello*
9. *Alza il braccio sinistro di 45 gradi, apri la mano, poi afferra il barattolo blu*
10. *Tira il braccio sinistro all'indietro fino a che non è uscito dal mobile*

E così via.

Che faticaccia! Ma del resto il mestiere di informatico consiste nel fare un lavoro immane per rendere la vita più semplice agli altri.

Ora, se vi mettete d'impegno potrete notare che ho volutamente commesso due errori in questa sequenza di istruzioni. L'ho fatto apposta per sottolineare che con le macchine niente può essere lasciato al caso, niente può essere taciuto, proprio perché la macchina non capisce il contesto, non può cavarsela da sola ed esegue solo comandi chiari, inequivocabili, elementari, consequenziali e completi.

Magari, chi vuole può prendersi due minuti per scovare gli errori. Io vado a prendere lo zucchero da me... :)

Allora, trovati? Per chi non ci fosse riuscito, diciamo quali sono:

1) Al punto 3, ho detto al robot di girare in corrispondenza della porta, ma non gli ho detto di attraversare la porta facendo due passi avanti (al punto 5, invece, l'azione è stata esplicitata).

2) Al punto 8, ho detto al robot di tirare il braccio all'indietro: per un essere umano sarebbe logico e inequivocabile, dato il contesto, che in quel punto della sequenza il braccio deve essere quello destro; ma un robot non ragiona così. Anzi, un robot non ragiona per niente: esegue soltanto. Chi ragiona per lui è l'essere umano che lo istruisce su cosa fare.

Questo semplice esempio ci fa intuire qual è la difficoltà di introdurre i ragazzi al pensiero computazionale. Adesso, però, passiamo alla descrizione di alcune attività con cui cominciare.

Giocare con il Coding

Innanzitutto occorre il **set da gioco**. Esso è diffuso insieme a questo tutorial.

Il set contiene, tanto per cominciare, carte come le seguenti:

Ci sono anche altri materiali, insieme alla descrizione dei giochi. Si tratta di un set originale e in divenire, poiché nel tempo si aggiungeranno altri giochi, altri simboli, altri schemi. Ma bisogna pur cominciare.

Attività 1. Salvate il Coniglio Gigio (8 - 11 anni)

Il gioco si può realizzare sia su un tavolo, sia usando come 'scacchiera' il pavimento. Nel primo caso si stamperà lo "Schema 1" contenuto nel Set e si useranno come 'pedine' i tre elementi ivi contenuti: il Coniglio Gigio, la Terribile Sfinge di Winckelmann (il quale si starà rivoltando nella tomba... :) e il pallino blu.

Nel secondo caso si useranno come caselle le mattonelle, se ci sono, oppure si riprodurrà una scacchiera con dei fogli di carta bianca oppure - meglio ancora - delle strisce di carta gommata attaccate al pavimento. Nel caso delle mattonelle sarebbe comunque un bene delimitare lo spazio di gioco con la carta gommata. Lo scopo, lo si sarà capito, è raggiungere il Coniglio Gigio partendo dalla casella del pallino blu e creando un percorso con le carte, che vanno stampate in numero congruo. Nel caso

del pavimento, il ruolo del pallino può essere svolto da un alunno. Per creare l'algoritmo si disporranno in sequenza le carte contenute nel set. La prima carta deve imprimere sempre la direzione al pallino, che essendo circolare non ha direzione sua: sarà un gira a destra, o gira a sinistra, o gira di 45 gradi, ecc.

Il percorso deve essere creato in modo da non incappare nella Terribile Sfinge, che, a quanto si sa, ama il coniglio alla cacciatora (sigh!).

Semplice, no? Mica tanto... O almeno semplice all'inizio. Già, perché i ragazzi vanno divisi in due gruppi e, a turno, i ragazzi di una squadra posizionano le sfingi (**solo cinque**) in caselle sulla scacchiera, lungo il possibile percorso, in modo da mettere in difficoltà gli avversari.

Anche essi hanno una limitazione, però: le sfingi non possono essere posizionate una accanto all'altra o una sopra l'altra (in orizzontale o verticale): va bene, invece, in diagonale.

Ok: anche così sembra semplice. E no! Ogni partita dura due manche: in ognuna una squadra posiziona le sfingi e l'altra cerca il percorso. Vince chi riesce a raggiungere il Coniglio Gigio usando il minor numero possibile di carte! E la faccenda non è più tanto semplice.

Un consiglio: quando un'azione va ripetuta diverse volte (per esempio andare avanti di quattro caselle) si usi la carta "Ripeti ..... volte", dichiarando (verbalmente o per iscritto) il numero delle volte. Accanto ad essa, si posizioni la carta "Vai avanti di un passo". In tal modo si sarà creato un 'ciclo', cioè l'azione di andare avanti di un passo verrà ripetute per il numero di volte dichiarato e si useranno meno carte. I software funzionano esattamente così.

Attività 2. Il Gioco dei se (8-11 anni)

Il seguente gioco serve a stimolare l'uso di semplici "istruzioni condizionali". Esse - diffusissime nella programmazione - sono quelle che, prima di eseguire un'operazione, impongono al calcolatore di verificare una condizione. Tipico caso è un intervallo di valori numerici. Esempio:

Se Mario ha più di 18 anni
*   può guidare un'auto*
altrimenti
*   non può*

L'esempio riportato è molto semplice, perché la condizione da verificare è una sola. Ma proviamo a complicarlo un po':

Se Mario ha più di 18 anni e se ha la patente
*   può guidare un'auto*
altrimenti
*   non può*

Questa seconda espressione condizionale è più corretta logicamente, perché aderisce in maniera rigorosa alla casistica reale (non basta essere maggiorenni, per

guidare). Essa è anche più complessa, perché verifica due condizioni, anziché una. Entrambe devono essere verificate per eseguire il comando.

Ora proviamo a costruirci un gioco.

Si può dividere una classe in due gruppi da dieci (se sono meno di venti si può fare da otto o da sei). Se sono più di venti si fa a turno, cercando di coinvolgere tutti. L'importante è che il numero di entrambe le squadre sia uguale e pari. Poi si stampano le carte apposite presenti nel Set. Eccone due esempi:

Le carte contengono due sequenze di numeri da 1 a 10; le sequenze sono di due colori diversi, rosso e nero. Se si ha a disposizione solo una stampante in B/N, le due sequenze di carte saranno comunque riconoscibili per il colore diverso. Sempre meglio stampare su cartoncino. Il costo della stampa sarà, in ogni caso, di pochi centesimi. Si mischino le carte, in modo che le due sequenze si confondano l'una con l'altra.

A questo punto l'insegnante passerà tra i ragazzi posti in due file l'una di fronte all'altra e consegnerà ad ognuno una carta. È di fondamentale importanza che i ragazzi non mostrino la carta a nessuno. Ognuno dovrà vedere solo la propria, nascondendola tra le mani. In questo modo, in entrambe le file ci saranno carte di entrambi i colori e di valori numerici distribuiti a caso.

Si noti che, usando questo sistema, è molto difficile che si ricostituiscano due sequenze complete dallo stesso lato, anche se di colori diversi. In altre parole, una squadra potrebbe avere i due 10, l'altra i due 4 e così via. Sarà il caso a decidere, come in tanti giochi di carte.

Bene: ci siamo quasi. A questo punto si scelga, per sorteggio o a turno, la squadra che comincia a giocare. Uno dei ragazzi della squadra che inizia (Es. Squadra A) sceglierà un ragazzo della squadra avversaria (Squadra B) e questi girerà la carta che ha in mano. Se la carta è uguale o maggiore di 6 e se il colore è lo stesso del ragazzo che ha scelto, allora l'iniziativa resterà al ragazzo che ha iniziato e l'avversario riconsegnerà la carta al docente e uscirà dal gioco; altrimenti, se anche una sola delle due condizioni non si verifica, uscirà dal gioco il ragazzo che ha scelto, l'avversario manterrà la carta in mano e farà lui una scelta nella Squadra A.

L'insegnante, per tenere la situazione del gioco sotto controllo, metterà le carte restituite su un banco, un po' come si fa con il solitario.

Ovviamente a un certo punto il gioco si fermerà, perché i numeri compresi nell'intervallo da 6 a 10 finiranno. Sarà l'insegnante, verificato che detti numeri sono stati tutti riconsegnati, a fermare il gioco.

La squadra vincente sarà quella che, a gioco chiuso, avrà il punteggio maggiore, che si otterrà facendo la somma delle carte in mano ai giocatori rimasti in campo.

Variante, il Gioco dei se semplificato (6-9 anni)

Il Gioco dei se si può semplificare per i bambini che hanno da poco cominciato a contare. Sarà sufficiente decidere che la condizione da verificare sia una sola: che il numero trovato sia maggiore o uguale a 6. Per il resto è uguale.

Variante: il Gioco dei se con livello di difficoltà maggiore (9-12 anni)

Se le condizioni di partenza della classe ce lo consentono, il gioco si può complicare, facendo in modo che le condizioni da verificare siano tre: le carte che consentono a un giocatore di mantenere la 'matta' (cioè di rimanere in gioco e continuare a scegliere) sono: 1) la carta trovata deve essere dello stesso colore che lui ha in mano; 2) la carta trovata deve essere uguale o maggiore di 6; 3) la carta trovata non deve essere la più bassa ancora in gioco. In altre parole, se i due 1 sono già usciti, la carta non dovrà essere un 2; se gli 1 e i 2 sono già usciti, la carta non dovrà essere un 3. E così via.

Questa ultima variante è decisamente complessa, ma riproduce situazioni che nella realtà sono diffusissime e che i software devono gestire. Si pensi a una lavatrice. Il software che la regola (ebbene sì: lavatrici, frullatori, frigoriferi, ecc. sono regolati da linguaggi di programmazione - tra i più diffusi ce n'è uno che si chiama Java - e sono quelli che ci fanno spendere tanti, tanti soldini quando una scheda elettronica si rompe...) deve gestire situazioni come la seguente:

SE il programma è stato selezionato e SE è stato premuto il tasto Avvio e SE il peso del bucato è minore di 2 Kg
avvia la modalità risparmio acqua
ALTRIMENTI
avvia la modalità consumo acqua standard

In realtà una lavatrice controlla molte altre cose contemporaneamente: apertura del rubinetto, temperatura dell'acqua, chiusura ermetica del portello...

E così via: da un 'SE', all'altro.

Attività 3: Algoritmi di comportamento - Tre azioni (9-12 anni)

Questo gioco consiste nell'associare delle sequenze comportamentali ad alcune situazioni date. Le sequenze di azioni sono tre: Fermarsi - Controllare - Procedere.

Le situazioni sono sei. Nel file con le carte contenuto nel set ce ne sono sei che riproducono tre sequenze possibili da tre azioni, in ordine diverso, delle tre azioni previste. Ogni sequenza è prevista due volte. Lo Schema 2 contiene sei situazioni. L'alunno dovrà posizionare le sei carte sullo schema, scegliendole, in maniera razionale, affinché rispecchino il giusto comportamento in ognuna di esse.

La seconda fase del gioco è decisamente più impegnativa. L'alunno, infatti, dovrà inventare tre situazioni nelle quali posizionare ognuna delle carte (una per ogni carta).

Se si vuole aggiungere un pizzico di agonismo all'esercizio, si può definire vincitore chi svolge, nel minor tempo possibile, purché in maniera corretta, le due fasi. Ognuna delle situazioni inventate dagli alunni sarà sottoposta a discussione, per evidenziare coerenza e/o eventuali incongruenze.

Se, poi, si svuole aggiungere al gioco anche l'elemento del brainstorming, si può dividere la classe in due o più gruppi, stampando le carte e il set tante volte quante sono le squadre. Attenzione: è assolutamente necessario che tutti i gruppi (o i singoli alunni) esprimano le loro proposte. Su un foglietto, o alla lavagna, si segnerà l'ordine in cui hanno finito, ma tutti devono terminare, anche se gli altri gruppi hanno finito già. Farli finire, infatti, è eticamente e didatticamente giusto, ma è anche indispensabile ai fini del gioco, perché non è detto che chi ha finito prima abbia espresso situazioni congruenti con le tre carte. Potrebbero essere sbagliate: in quel caso, si passerà al gruppo che ha finito per secondo, finché non se ne trova uno che ha espresso situazioni accettabili. Ovviamente, è comunque un bene che tutte le soluzioni espresse - anche di chi non ha vinto - vengano lette e commentate.

E buon algoritmo a tutti!

Algoritmi di comportamento - Quattro azioni (10-12 anni)

Questo gioco è uguale al precedente, salvo che le carte contengono una sequenza di quattro azioni. Per il resto valgono tutte le regole della versione a tre. Nel set di carte ce ne sono tre che riproducono possibili sequenze di quattro azioni. Per giocare, occorre utilizzare lo Schema 3.

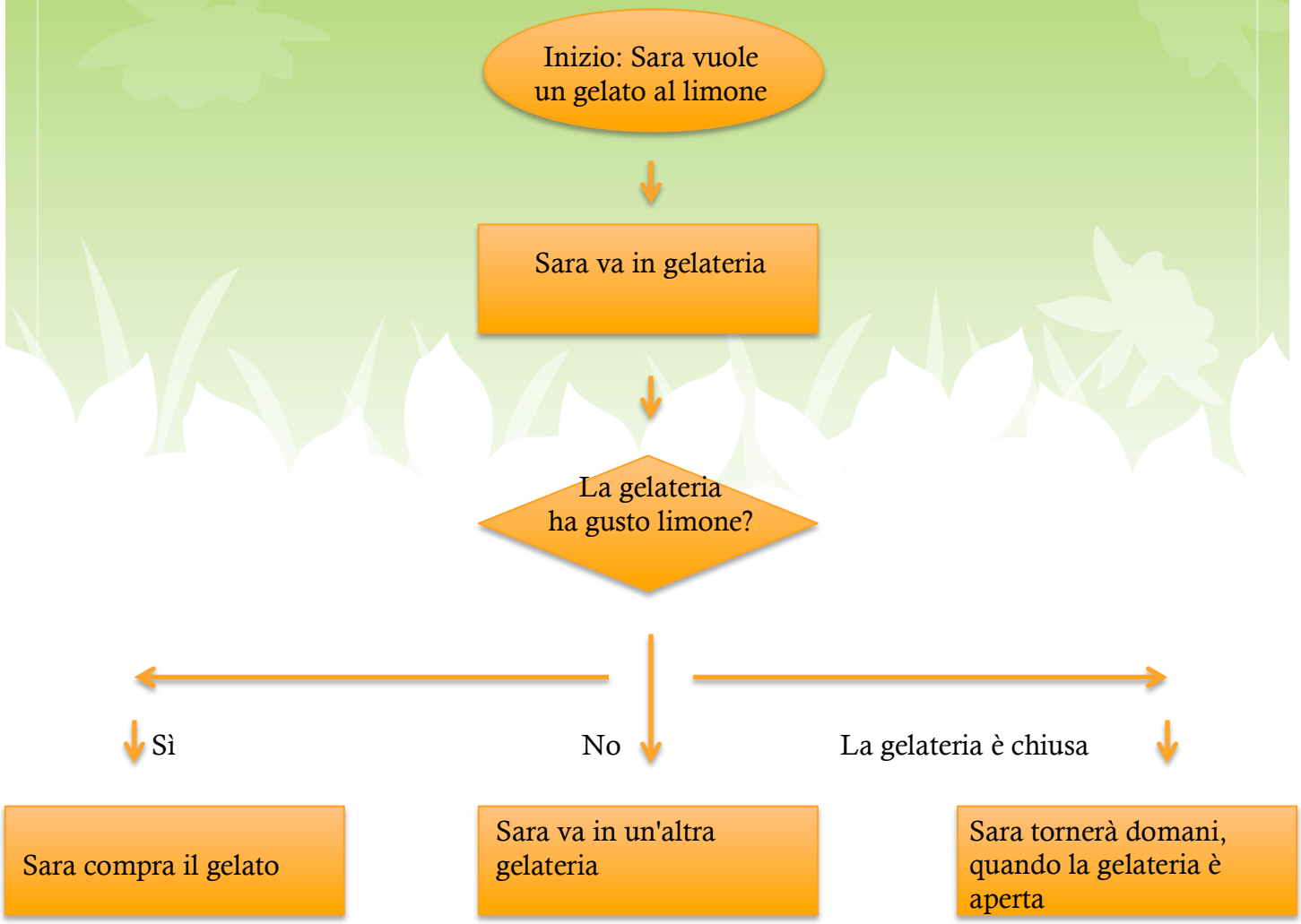
Attività 4: Diagrammi di flusso (9-13 anni)

Un diagramma di flusso è, semplicemente, un insieme di blocchi e di frecce che rappresentano, graficamente, un algoritmo. Alle pagina 1 e 2 dello Schema 4 se ne trovano due esempi, che risolvono due situazioni reali, quella di un impiegato che deve andare al lavoro in una giornata nuvolosa e quella dello zio Piero, che deve pitturare il salotto.

Creare diagrammi di flusso è divertente, ma occorre seguire alcune semplici regole. Per comprenderle, guardiamo i due esempi dello Schema 4.

- *Un diagramma di flusso comincia sempre con un Inizio e termina con una fine. I blocchi di inizio e fine sono rappresentati, generalmente, con una forma ovoidale.*
- *Un diagramma si articola in 'passaggi', ognuno dei quali è contenuto in un blocco. Ci sono blocchi di processo, che sono generalmente schematizzati con un rettangolo. Nei blocchi di processo i soggetti coinvolti compiono, appunto, un'azione.*
- *Esistono, poi, blocchi detti di 'controllo'; essi sono rappresentati, generalmente, con dei rombi. Nei blocchi di controllo si effettua un test, che si risolve spesso con un Sì o un No. Essi, però, possono prevedere anche più di due esiti. Dai risultati del controllo, dipende il proseguimento del percorso all'interno del diagramma.*
- *Occorre ricordare che nei diagrammi di flusso si deve sempre partire dall'inizio e si deve sempre poter raggiungere una fine.*

I blocchi di processo hanno una sola freccia in ingresso e una sola in uscita. I blocchi di test, invece, come dicevamo, possono contenere anche più di due uscite. Ecco un esempio a tre uscite:



Ovviamente, a ognuna delle possibilità corrisponderà una scelta di Sara.

Nello Schema 4, dopo gli esempi, vengono proposte due attività inverse l'una all'altra. Nella prima, partendo da una situazione proposta, i ragazzi dovranno ricavare un diagramma coerente.

Nella seconda è proposto un diagramma vuoto. I ragazzi dovranno inventare una situazione che corrisponda al diagramma stesso.

Sia per la prima attività che per la seconda, il docente potrà lavorare di fantasia, creando nuove situazioni, oppure disegnando alla lavagna dei diagrammi di propria invenzione, sui quali gli alunni potranno desumere situazioni.

Un consiglio per l'attività 2: prima di disegnare alla lavagna un diagramma di propria invenzione, è sempre bene partire mentalmente da una situazione reale, che non verrà comunicata agli alunni: ciò consentirà di produrre diagrammi coerenti.

Attività 5 (13 - 14 anni)

Per svolgere questa attività occorrono solo un gessetto e la lavagna di ardesia. In alternativa un foglio di carta e la matita. Gli alunni avranno a disposizione quattro

formule inglese: "if", "else if", "else", "when". I primi tre sono universalmente presenti nei linguaggi di programmazione di alto livello.

I banchi verranno posizionati lungo le pareti dell'aula e le sedie accostate ad essi, in modo tale da lasciare gran parte dello spazio centrale libero. A questo punto un alunno volontario uscirà un attimo dall'aula e un oggetto verrà nascosto in un punto dell'aula (in un cassetto, in uno zainetto, ecc.). Si chiederà all'alunno di rientrare e lui si posizionerà al centro dell'aula. Poi comincerà a fare delle domande: "Se giro a desta mi avvicino all'oggetto?" Riceverà delle risposte in proposito.

L'aspetto fondamentale è che alla lavagna queste domande vengano trasformate in porzioni di 'codice':

```
if (Giro a destra){
Conseguenza = Mi sto allontanando
}
else if (Giro a sinistra) {
Conseguenza = Mi sto avvicinando
}
```

L'alunno, ovviamente, girerà a sinistra. Vediamo, però, il caso seguente:

```
if (Giro a destra){
Conseguenza = Mi sto allontanando
}
else if (Giro a sinistra) {
Conseguenza = Mi sto allontanando
}
else if (Vado avanti)
{
Conseguenza = Mi sto allontanando
}
```

All'alunno non rimarrà che un'ultima soluzione: girarsi di 180°, perché evidentemente l'oggetto è nascosto dietro le sue spalle.

Mano a mano che si individua uno step, l'azione prescelta verrà appuntata in un angolo della lavagna o su un foglietto.

Se l'alunno raggiunge i banchi appoggiati al muro, il codice diventerà:

```
when (raggiungo il muro) {
    if (Giro a destra){
    Conseguenza = Mi sto allontanando
    }
    else if (Giro a sinistra) {
    Conseguenza = Mi sto avvicinando
    }
}
```

Si noterà che qui ci sono istruzioni 'annidate': cioè la classe capirà che 'quando' si verifica una data condizione, occorre fare il ragionamento 'se', 'altrimenti se' e regolarsi di conseguenza.

Con un'attività del genere, che è adatta a ragazzi ormai grandicelli, siamo già a un passo dalla programmazione vera e propria. Ecco un altro esempio.

```
when (Esco di casa) {
    if (Piove){
    Conseguenza = Prendo l'auto
    }
    else if (C'è il sole) {
    Conseguenza = Vado a piedi
    }
}
```

Attività 6 (13-14 anni).

Traslare in codice alla lavagna i seguenti ragionamenti, sull'esempio dei simboli e delle parole inglesi usate nell'attività precedente:

1. Quando passi sotto casa mia, se vedi la finestra aperta citofona; altrimenti vai a scuola da solo.
2. Se compro un maglione rosso, piacerà a mia madre; se, invece, lo compro bianco, piacerà alla mia ragazza; se, invece, lo compro blu, piacerà a me.
3. Quando senti Luca, se è tornato dalle vacanze, fatti ridare la mia bici, se, invece, se torna domani, ci vai domani, altrimenti fattela dare da suo fratello.
4. Se papà tornerà presto, quando sarà a casa, gli chiederemo quale menu preferisce; altrimenti, glielo chiederemo per telefono.

Conclusioni

Pur scrivendo questo tutorial, mi rendo conto, prima che me lo dica qualcun altro, che il Coding, guardato con occhi scettici, può rappresentare "un problema in più". E la scuola di problemi in più non ne ha certo bisogno: bastano e avanzano quelli che ci sono.

Esso, però, oltre ai forti risvolti formativi, ha anche un altro, indubbio aspetto positivo: è motivante. La modalità unplugged, che ho provato ad introdurre in queste prime pagine sull'argomento, ha il limite di non utilizzare la potenza dei computer. Potenza e capacità motivante che si massimizza, invece, su siti come **https://code.org/** oppure **https://scratch.mit.edu/,** siti che consiglio, fin da subito, di visitare e cominciare a utilizzare. Ne riparleremo a breve, ma essi sono abbastanza intuitivi.

Il bisogno, qui, era di trovare attività che avessero due caratteristiche:

1) essere adatte anche ad alunni piccoli;

2) essere praticabili anche in assenza di condizioni ottimali, rappresentate dalla disponibilità di hardware e connessioni di buona qualità. Non sempre e non dappertutto questa disponibilità è presente.

Le prime idee sono venute guardando le attività del sito: http://codeweek.it/cody-roby/, ma poi sono andato oltre, per creare idee diverse. Tutto il materiale, comunque, è originale.

Del resto, la creatività si vede e si misura soprattutto in presenza di risorse e ausili modesti. Inoltre i ragazzi, soprattutto i più piccoli, amano i giochi fatti "di piccole cose", a amano i giochi di squadra e di gruppo.

La fantasia dei colleghi può sbizzarrirsi a trovare soluzioni nuove, giochi nuovi, percorsi logici, modalità, schemi e sistemi a cui, con i miei limiti, non riesco a pensare da solo. I buoni prodotti sono sempre prodotti di equipe.

A me toccava introdurre l'argomento. Nei prossimi mesi, piano piano e in base anche alla risposta della nostra platea, aggiusteremo il tiro.