

Capitolo 11

Girando in tondo: dal cerchio all'orbita di Halley

11.1 Low floor...



Figura 11.1: Per le fotografie da cui ho tratto le immagini e per buona parte dei testi in questa sezione sono debitore della maestra Antonella Colombo che ringrazio per avere condiviso le sue preziose esperienze.

78CAPITOLO 11. GIRANDO IN TONDO: DAL CERCHIO ALL'ORBITA DI HALLEY

Percorrendo il caotico dibattito intorno al coding non è raro imbattersi in coloro i quali pensano che si tratti di pratiche buone giusto per i primi anni di scuola come in quelli che invece ritengono che il coding sia adatto agli ultimi anni delle scuole superiori. Sono ambedue in errore. Per illustrare la questione, anche in questo capitolo esploriamo una sorta di progressione dal *low floor* allo *high ceiling* di Papert. Vale la pena di riprendere le sue parole.

L'obiettivo delle prime esperienze dei bambini nell'ambiente di apprendimento della Tartaruga non è quello di imparare regole formali ma di sviluppare nuovi modi di concepire i propri movimenti. Tali modi sono esprimibili nella Lingua della Tartaruga e in questa diventano "programmi", "procedure" o "equazioni differenziali". Proviamo a guardare più da vicino come un bambino, che abbia già imparato a muovere la Tartaruga in linea retta per disegnare quadrati, triangoli e rettangoli, possa imparare a farle disegnare un cerchio. Immaginiamo – cosa che ho osservato un centinaio di volte – un bambino che domandi: "Come faccio a fare un cerchio con la Tartaruga?" L'insegnante, nell'ambiente Logo, non dà la risposta a domande del genere bensì introduce il bambino a un metodo per risolvere non solo questo problema ma anche un'intera categoria di altri problemi. Il metodo si può riassumere in una frase: "Gioca con la Tartaruga." Il bambino viene incoraggiato a muoversi come farebbe la Tartaruga sullo schermo per ottenere il disegno desiderato. Per il bambino che vuole disegnare un cerchio, l'atto di provare a muoversi circolarmente potrebbe tradursi nella descrizione seguente: "Quando ti muovi in cerchio tu fai un piccolo passo e poi giri subito un poco. E continui a fare sempre così." Una volta giunti ad un a simile descrizione, la formulazione nella Lingua della Tartaruga viene spontanea:

REPEAT [FORWARD 1 RIGHT 1]

Qualche bambino meno esperto potrebbe necessitare di ulteriore aiuto. Ma questo aiuto non dovrebbe consistere nella spiegazione di come fare a disegnare il cerchio bensì nell'insistere sul metodo, che concerne (oltre il consiglio di "giocare con la Tartaruga") nello sviluppare una forte connessione fra l'attività personale e la creazione di conoscenza formale.

...

L'episodio del cerchio disegnato con la Tartaruga illustra l'apprendimento sintonico. Questo termine, preso in prestito dalla psicologia clinica, sta in contrapposizione con l'apprendimento dissociato di cui abbiamo già discusso. Talvolta il termine viene usato con degli specificatori che denotano certi tipi di sintonicità. Ad esempio, il cerchio della Tartaruga è sintonico per il corpo perché tale cerchio è saldamente collegato alla percezione fisica del proprio corpo da parte del bambino. Oppure è anche sintonico per l'ego perché è coerente con

la percezione di sé propria dei bambini, come persone con intenzioni, obiettivi, desideri, preferenze e avversioni. Un bambino che disegna un cerchio con la Tartaruga vuole disegnare un cerchio; quando ci riesce è orgoglioso e eccitato.

La geometria della Tartaruga si impara bene perché è sintonica. E questo aiuta anche nell'apprendimento di altre cose perché incoraggia l'uso consapevole e deliberato di strategie matematiche di problem solving.

Una delle cose più belle che mi capita di vivere in questi anni è la ricca e spontanea "restituzione" da parte dei miei studenti, sia da i più giovani che sono in procinto di laurearsi che da coloro che incontro in percorsi di aggiornamento di vario tipo. Un patrimonio inestimabile che consente di realizzare quasi un miracolo, quello di situare immediatamente ciò di stiamo parlando. E allora qui ci possiamo prendere il lusso di passare dal racconto di Papert delle proprie visioni a quello dove Antonella, una maestra di Paderno d'Adda, a tali visioni ha dato vita nella propria classe. Il racconto è nella forma del diario, che è un forma di compito che chiedo quasi sempre a tutti. Mi defilo nuovamente e lascio la parola questa volta a Antonella.

23 maggio 2017

Anch'io ho utilizzato Code.org in questi ultimi anni con buoni risultati. Quest'anno in prima ho tentato – dopo averne sentito parlare in questo ambiente e grazie ad un progetto che si attua nel mio Istituto e che prevede l'intervento nelle classi di ragazzi di un Istituto di Scuola superiore ad indirizzo di Informatica (alternanza scuola-lavoro) – di mandare avanti i due sistemi in parallelo: la classe è stata divisa in due gruppi, un gruppo lavorava con Logo, l'altra con l'Ora del Codice

Dopo un'ora cambio di gruppo; io ho seguito maggiormente il gruppo che lavorava con Logo, l'altro gruppo è stato seguito dall'insegnante di sostegno compresente. In realtà, prima ancora che arrivassero i ragazzi dell'ITS, avevo già avviato i bambini all'uso di Code.org.

I ragazzi vengono preventivamente informati su che cosa devono proporre alle varie classi: si parte dall'ora del Codice nelle prime classi per arrivare in quarta-quinta a lavorare con il Lego WeDo e Scratch. Per queste due ultime attività è stato predisposto un mini laboratorio con 6 notebook.

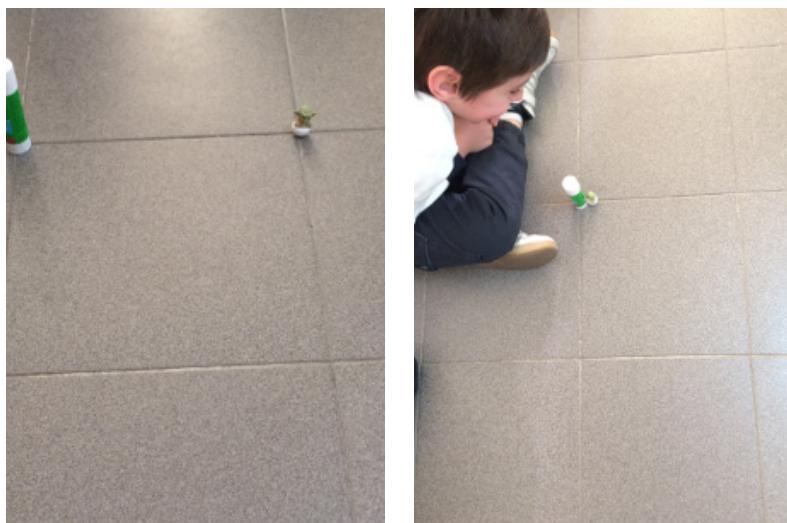
Oggi era il turno delle mia classe e finalmente ho realizzato il mio progetto: ho diviso la classe in due gruppi, un gruppo nel laboratorio grande ha continuato l'ora del codice con l'insegnante di sostegno, un gruppo con me nel laboratorio più piccolo si è avvicinato al linguaggio della tartaruga.

Prima di arrivare a capire le istruzioni per disegnare un quadrato ho "utilizzato" il loro corpo: ho chiesto loro di camminare e di descrivere un quadrato, cercando di dare istruzioni precise.

Vado avanti, poi giro, poi giro, poi giro e arrivo al punto di partenza.
Non siamo ancora precisi, riproviamo.

80CAPITOLO 11. GIRANDO IN TONDO: DAL CERCHIO ALL'ORBITA DI HALLEY

Consegno ad ogni bambino un piccolo pupazzetto e chiedo di farlo camminare sul bordo di una piastrella, introduco i comandi, perché mentre noi proviamo, il nostro studente Davide (ex alunno) sul foglio di writer traduce i nostri comandi, così impariamo che forward significa vai avanti, right gira a destra.



Chiedo a Davide di far partire la tartaruga, percorre un tratto, ma poi non si gira, anche se le abbiamo detto di girare a destra, allora spiego che bisogna dirle di girare di 90 (gradi): i bambini osservano che la tartaruga si gira e allora...

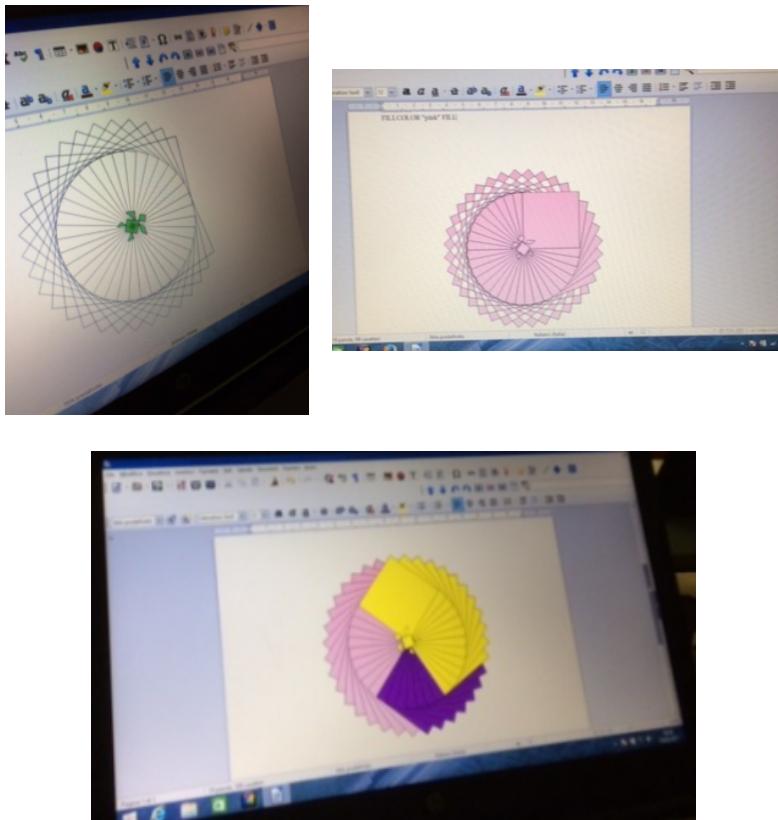
```
FORWARD 100  
RIGHT 90  
FORWARD 100  
RIGHT 90  
FORWARD 100  
RIGHT 90  
FORWARD 100
```

Davide scrive i comandi, introduce i vari tasti e ...magia finalmente la tartaruga disegna il quadrato; intenzionalmente decido di non scrivere home, così la tartaruga può continuare a muoversi e formare quattro quadrati. È tutto questo ha già il suo fascino. Successivamente introduco una rotazione a caso della tartaruga e le chiediamo di continuare a lavorare: "che bello, sembrano le nostre cornicette" commenta Giulia; ma l'osservazione più significativa, che formula sempre Giulia, è: "Guarda, sta diventando rotondo".

I bambini a coppie al portatile aprono LibreOffice e fanno da soli la stessa cosa che abbiamo visto insieme con interpretazioni differenti

sull'ultima rotazione.

Introduco anche il comando del colore e anche qui dopo qualche tentativo avvengono libere interpretazioni e combinazioni di colori.



Abbiamo avuto un momento di pausa e la settimana scorsa sono tornati di nuovo gli studenti con i quali abbiamo ripreso l'esperienza. Ho colto un'occasione d'oro quando è uscita la domanda: "Ma come si disegna il cerchio?" "Secondo voi?" ho risposto. Sono partite le ipotesi, si sono avvicinati, ma non hanno ancora trovato la soluzione; poi, ad un certo punto, dopo che hanno intuito che right 1 poteva essere utile, una bambina ha osservato: "Con sempre right la tartaruga gira su se stessa e non si muove".

Quello che ho trovato di molto significativo in questa esperienza è che la progettazione aiuta i bambini a mobilitare e raggiungere molte competenze, per esempio destra e sinistra: in prima si dedica parecchio tempo a questo obiettivo, schede, esercizi..., forse l'attività di progettazione, come del resto affermava Papert, è più stimolante e lascia tracce indelebili nei processi di apprendimento.

Sabato, 3 giugno

Ripreendo solo oggi le fila del discorso. Il 23 maggio appunto ho voluto sperimentare la dimensione sintonica dell'apprendimento. Siamo

82CAPITOLO 11. GIRANDO IN TONDO: DAL CERCHIO ALL'ORBITA DI HALLEY

andati in corridoio e prima abbiamo sperimentato con il corpo i comandi che abbiamo imparato a dare alla tartaruga

L'obiettivo era quello di scoprire i comandi per il cerchio, allora abbiamo per prima cosa percorso un disegno che c'è a terra nel nostro corridoio, osservando attentamente i nostri movimenti.



Poi abbiamo abbiammo camminato sul bordo di un cerchio che si usa in educazione motoria in palestra.



Prima di far continuare Antonella, è interessante notare nella foto a destra come la bambina cerchi di seguire la curva della circonferenza descrivendo così, con i piedi, il processo: faccio un piccolo passo e giro un poco, faccio un piccolo passo e giro un poco...

Infine...”via le calzine, tracciamo un cerchio con le orme dei nostri piedini, camminando sulla farina, come se fossimo al mare” (complice il tempo che era particolarmente soleggiato e caldo!).



Dopo l'esperienza gioiosa e coinvolgente del camminare a piedi scalzi sulla farina, c'è il momento della discussione: quali comandi dare alla tartaruga per disegnare il cerchio?

La settimana scorsa ho proposto ai bambini di usare il pc e di provare a disegnare il cerchio: usano forward e right, ma assegnano valori come 90, 40...; io suggerisco di verificare che cosa disegna la tartaruga con le procedure scritte: procedono dunque per prove ed errori. Ben presto si rendono conto che qualcosa non va.

”Bisogna cambiare i numeri”

”Dobbiamo andare in curva”

”Bisogna continuare a fare right”

”Se facciamo sempre right, viene un punto piccolissimo”.

Poi l'insight:

”Dobbiamo fare forward 1 right 1”.

”Proviamo” suggerisco io

”Bisogna ripeterlo tante tante volte.”

Li aiuto a scrivere, faccio finta di non sapere e di essere stanca di scrivere, per cui faccio vedere che con Ctrl+V il comando può essere ripetuto tante volte quanto vogliamo noi, per cui continuo a cliccare Ctrl+V finché loro mi dicono stop.

La tartaruga disegna e traccia un arco: è già qualcosa. Ne approfitto per introdurre repeat e chiedo quante volte devo far ripetere alla tartaruga forward1, right 1.

100...200...300 e intanto, grazie a questi tentativi, la tartaruga traccia archi sempre più ampi. Manca poco, Giovanni mi propone di

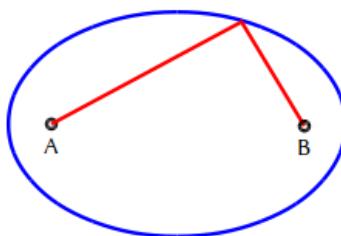
scrivere 355, ma in classe ho delle ragazzine di V (la loro maestra è assente e tutti gli alunni sono stati distribuiti nelle classi) che interrompono il loro lavoro e seguono il nostro, le coinvolgo e i bambini chiedono il loro parere – "Perché sono di V, sono più grandi di noi". Immediatamente mi rispondono 360 e finalmente la tartaruga disegna un cerchio.

È affascinante come queste prime esperienze possano indurre a riflettere sul concetto di cerchio senza che se ne abbia ancora una conoscenza formale. Eppure, così facendo, si lambisce un concetto assai più avanzato di tutto ciò che i ragazzi vedranno prima della maturità, ovvero il concetto di calcolo differenziale. Tuttavia, appena essi verranno in possesso della nozione formale di circonferenza, sarà possibile tornare in Logo e riprendere il discorso attraverso il modo sintetico di produrre un cerchio, con l'istruzione CIRCLE D, dove D rappresenta il diametro del cerchio, riflettendo sulla definizione come luogo dei punti equidistanti da un dato punto. E qui si potrebbe andare oltre, partendo dall'idea di un "cerchio sbagliato", o "schiacciato", e ricorrendo alle riflessioni di Emma Castelnuovo [3].

Sempre un argomento di matematica, quale lo studio dei triangoli isoperimetrici con uguale base, porta a osservare quello che abbiamo sotto gli occhi.

Il materiale è, anche questa volta, un pezzo di spago.

Per costruire dei triangoli di uguale base e uguale perimetro facciamo così: fissiamo due chiodi – siano A e B – su un tavolo su cui è disteso un foglio di carta; AB sarà la base dei nostri triangoli. Leghiamo poi gli estremi di un pezzo di spago ai due chiodi, tenendo presente che lo spago deve essere più lungo del tratto AB. Facciamo in modo, valendoci di una matita, che lo spago resti sempre ben teso e... lasciamoci guidare dalla matita.



Questa, guidata dallo spago, disegnerà sul foglio una curva a forma di ovale: è un'ellisse. I punti A e B si chiamano fuochi dell'ellisse. Dunque: i vertici dei triangoli isoperimetrici e di uguale base si trovano su un'ellisse.

Un problema di geometria ci ha condotti al disegno dell'ellisse. Con lo stesso pezzo di spago possiamo costruire un'ellisse più o meno "schiacciata", a seconda della distanza fra i punti A e B. Si può ottenere anche un cerchio, se i due punti coincidono: il cerchio, infatti, è un'ellisse particolare.

L'ellisse, dopo averla incontrata in problemi di geometria, la ritroviamo per la strada, quando la "calpestiamo" (perché un disco sennaletico dà, come ombra, un'ellisse). Nella nostra vita convulsa raramente ci soffermiamo a osservare l'ombra di un oggetto data dai raggi del soleo da una lampadina. Ma ecco che ora un'attività di geometria ci sollecita a guardare di più ed è proprio il confronto fra l'effetto-ombra dato dai raggi del sole e quello dato da una lampada puntiforme che stimola la nostra facoltà di osservazione.

Guardiamo, ad esempio, due matite disposte in verticale su un tavolo. Se vengono illuminate dal sole accade che anche le ombre sono parallele; se invece è una lampada che le illumina le ombre si divergono.

Da qui lo studio matematico delle trasformazioni affini e delle trasformazioni proiettive, fino ad arrivare alla prospettiva, all'arte, a come si guarda un quadro, alla storia.

È un piccolo problema di geometria che ha stimolato a osservare e a... guardarsi intorno.

Sono esperienze fisiche quella che propone Emma Castelnuovo ma non vanno considerate in contrapposizione a quelle che si possono fare con un computer, nel modo che abbiamo mostrato; che poi è il modo che suggerisce Papert attraverso il concetto di apprendimento sintonico.

Purtroppo l'avvento del nuovo viene solitamente vissuto in contrapposizione al vecchio, generando l'usuale dicotomia integrati-apocalittici. È il tempo a risolvere solitamente ciò che la saggezza potrebbe evitare subito, con molto maggiore profitto. Quella da promuovere è quindi una visione integrata, dove si fa ricorso a tutti i mezzi possibili, tradizionali e moderni, manuali e virtuali, per raggiungere gli obiettivi didattici in una prospettiva più ampia e completa possibile.

11.2 ... high ceiling

Fin qui abbiamo esplorato "il tondo" dal basso, dal *low floor* di Papert, almeno in parte. Molto altro potrebbe essere aggiunto, ma ora, per rifarsi al quesito iniziale, proviamo a muoverci verso l'alto per vedere dove si possa trovare l'*high ceiling*. Si tratta di un quesito che è emerso per esempio recentemente durante un incontro con un nutrito gruppo di animatori digitali, dove ci si poneva la domanda: ma si può fare il *coding* anche negli anni successivi, alle scuole superiori? Molti sostenevano di no, ritenendo che si trattasse di pratiche troppo banali.

Riprendiamo le parole di Papert.

Il bambino che ha disegnato il cerchio con la Tartaruga non ha imparato qualcosa sul formalismo dell'analisi, per esempio che la derivata di x^n è nx^{n-1} , ma qualcosa sul suo impiego e sul suo significato. Infatti il codice per disegnare il cerchio con la Tartaruga conduce a un formalismo alternativo di quelle che sono tradizionalmente chiamate "equazioni differenziali" ed è un veicolo efficace delle idee che soggiacciono al differenziale. Questo è il motivo per cui si possono capire

così tante cose con la Tartaruga; il codice del cerchio rappresenta un'analogia intuitiva dell'equazione differenziale, un concetto che appare in quasi tutti gli esempi di matematica applicata tradizionale.

La potenza del calcolo differenziale risiede molto nella capacità di descrivere le variazioni in base a ciò che accade nelle loro immediate vicinanze. È questa caratteristica che ha consentito a Newton di descrivere il moto dei pianeti. Via via che questi tracciano l'orbita, sono le condizioni locali nel luogo dove si trova il pianeta che determinano il suo prossimo passo. Nelle nostre istruzioni della Tartaruga, FORWARD 1 RIGHT 1, ci si riferisce solo al luogo dove si trova la Tartaruga e a quello dove si troverà il momento dopo. Questo è quello che rende differenziale un'equazione. In ciò non vi è alcun riferimento a luoghi remoti rispetto al percorso. La Tartaruga vede il cerchio cammin facendo, nell'immediata vicinanza, ed è cieca rispetto a tutto il resto che si trova più lontano. Questa proprietà è così importante che i matematici hanno un nome per essa: la geometria della Tartaruga è "intrinseca". Lo spirito della geometria differenziale intrinseca si palesa quando si considerano i diversi modi di concepire una curva, ad esempio il cerchio. Per Euclide la caratteristica che definisce il cerchio è la distanza costante dei suoi punti da un altro punto, il centro, che però non fa parte di esso. Nella geometria di Cartesio, in questo caso più similmente a Euclide, i punti del cerchio sono caratterizzati dalla loro distanza rispetto a qualcos'altro, vale a dire dai due assi perpendicolari delle coordinate. Così, per esempio, un cerchio è definito da:

$$(x - a)^2 + (y - b)^2 = R^2 \quad (11.1)$$

Nella geometria della Tartaruga un cerchio è definito dal fatto che questa continua a ripetere uno stesso atto: FORWARD un poco, GIRA un poco. Questa ripetizione garantisce che la curva abbia "curvatura costante", dove si stabilisce di quanto si deve girare ad ogni passo.

La geometria della Tartaruga appartiene ad una famiglia di geometrie che godono di proprietà assenti in quelle euclidea e cartesiana. Queste sono le geometrie differenziali che si sono sviluppate a partire da Newton e che hanno reso possibile gran parte della fisica moderna. Abbiamo osservato come quello delle equazioni differenziali sia il formalismo che ha consentito alla fisica di descrivere il moto di una particella o di un pianeta. Nel capitolo 5 [NdR: di Minstorms], dove descriveremo questo fatto con maggiori dettagli, vedremo come questo sia anche il formalismo appropriato per descrivere il moto di un animale oppure l'evoluzione di un'economia. E arriveremo anche a capire che non è un coincidenza il fatto che la geometria della Tartaruga sia collegata sia all'esperienza di un bambino che alle principali conquiste della fisica, in quanto, la visione del moto di un bambino, sebbene meno precisa nella forma, condivide la struttura matematica dell'equazione differenziale con le leggi del moto di un pianeta che gira intorno al sole o quelle delle falene che girano intorno alla

fiamma di una candela. E la Tartaruga non è ne più ne meno che la ricostruzione in forma computazionale intuitiva del nucleo qualitativo di questa struttura matematica. Quando torneremo su queste idee nel capitolo 5, vedremo come la geometria della Tartaruga apra le porte alla comprensione intuitiva dell'analisi, della fisica e della modellazione matematica così come viene impiegata nelle scienze biologiche e sociali.

Prendiamo quindi le mosse dal riferimento di Papert alla fisica di Newton, rifacendosi dai principi della dinamica e dalla legge di gravitazione universale.

11.2.1 La visione analitica classica

Poniamo il problema in maniera generale. Non potrà essere questa l'impostazione proposta a scuola, perché occorrono conoscenze matematiche superiori, ma qui ci serve per inquadrare e comprendere correttamente la questione.

Affrontiamo il problema di due corpi che siano isolati da perturbazioni esterne. Ad esempio quello di un satellite che gira intorno alla terra o di una cometa che gira attorno a un pianeta. Il moto di questi corpi è governato dalla legge di gravitazione universale di Newton:

$$\mathbf{F}_{Mm} = -\frac{GMm \mathbf{r}}{r^2} \quad (11.2)$$

Dove \mathbf{F}_{Mm} è la forza, G la costante di gravitazione universale, pari a 6.67×10^{-11} ($\text{Newton} \times \text{metri}^2/\text{Kg}^2$), M è la massa del pianeta, m la massa del satellite, \mathbf{r} la distanza fra i baricentri dei due corpi e dove, ricordiamo, il grassetto si denota per indicare la naturale vettoriale di una quantità. Quindi in questo specifico caso, poiché il nostro ragionamento si svolge tutto in due dimensioni, la forza \mathbf{F}_{Mm} e la distanza \mathbf{r} sono vettori con due componenti: una lungo x e l'altra lungo y .

Ma cosa vuol dire in pratica che "il moto dei corpi è governato dalla legge di gravitazione"? Significa che, per individuare il moto di un corpo, dobbiamo applicare il II principio della dinamica, secondo il quale:

$$\mathbf{F} = m\mathbf{a} \quad (11.3)$$

Dove \mathbf{F} è la forza applicata al corpo di massa \mathbf{F} e \mathbf{a} è l'accelerazione impressa al corpo in virtù di tale forza. Lo scopo di queste relazioni è quello di consentire la determinazione del moto dei corpi a partire da predeterminate condizioni iniziali. Per fare questo occorre riscrivere l'equazione 11.3 tenendo conto della forza data nell'equazione 11.2:

$$-\frac{GMm \mathbf{r}}{r^2} = m\mathbf{a} \quad (11.4)$$

Risolvere questa equazione significa in primo luogo ricavare l'accelerazione \mathbf{a} :

$$\mathbf{a} = -\frac{GM}{r^2} \frac{\mathbf{r}}{r} \quad (11.5)$$

e da questa derivare l'andamento nel tempo della velocità e quindi dell'accelerazione, tenendo conto che:

$$\mathbf{a} = \frac{d^2\mathbf{r}}{dt^2} \quad (11.6)$$

ovvero

$$\mathbf{v} = \frac{d\mathbf{r}}{dt}, \mathbf{a} = \frac{d\mathbf{v}}{dt} \quad (11.7)$$

È qui che ci dobbiamo fermare, perché siamo di fronte a equazioni dove non appaiono le grandezze incognite, ma le derivate di tali grandezze rispetto a un'altra variabile, il tempo in questo caso. In particolare, l'equazione 11.5 è un'equazione differenziale del II ordine perché fornisce un'espressione per l'accelerazione, ovvero la derivata seconda della posizione rispetto al tempo. Ci dobbiamo fermare perché la soluzione delle equazioni differenziali non rientra nell'orizzonte contemplato dai programmi della scuola secondaria. Di fatto l'approfondimento che ci possiamo permettere è assai limitato. L'apprendimento della fisica nella scuola si limita per di più alla memorizzazione di alcune formule e all'esecuzione di qualche esercizio, con procedimenti che la maggioranza degli studenti assimilano in maniera meccanica e effimera, senza realmente comprendere l'essenza dei concetti e tanto meno senza recepire il benché minimo sentore del "pensiero fisico", come del resto del "pensiero matematico". I risultati si vedono dal fatto che nella cittadinanza sopravvivono solo brandelli di pensiero scientifico, un pensiero che non ha quasi nulla a che vedere con l'avere imparato a memoria $F = ma$ una volta nella vita, giusto per superare un'interrogazione.

Ci colpisce tuttavia il fatto che di calcolo differenziale abbiamo già parlato, e che lo abbiamo fatto addirittura trovandoci al *low floor*, dove stavamo parlando di attività da fare con allievi assai più piccoli! Lì avevamo descritto, anzi Papert aveva descritto, il concetto come un processo locale: il bambino (o la Tartaruga) che produce un cerchio preoccupandosi solo di fare un passo e girare, senza alcuna nozione esplicita e sintetica del concetto di cerchio. Ebbene questa è proprio la descrizione verbale di un'equazione differenziale, dove anziché esprimere delle quantità in funzione di altre si esprimono le loro variazioni, le quali sono sempre inherentemente locali.

11.2.2 Física computazionale

Affrontiamo quindi il problema espresso dall'equazione 11.5 in maniera computazionale. Riscriviamo l'equazione così:

$$\frac{d^2\mathbf{r}}{dt^2} = -\frac{GM}{r^2} \frac{\mathbf{r}}{r} \quad (11.8)$$

O volendo così (ognuno faccia riferimento alla versione che preferisce):

$$\frac{d}{dt} \frac{d\mathbf{r}}{dt} = -\frac{GM}{r^2} \frac{\mathbf{r}}{r} \quad (11.9)$$

E affrontiamolo nel seguito come avevamo fatto con il cerchio alla maniera di Papert, dove si faceva un piccolo passo per volta e poi si girava di un po', e via di seguito. Anche qui facciamo un piccolo passo per volta.

11.2.3 Le condizioni iniziali

Ma prima di tutto: a partire da dove e in quale direzione dobbiamo fare il passo? Ecco questo è un primo concetto fondamentale del procedimento di conoscenza fisica: la determinazione delle condizioni iniziali, o delle "condizioni al contorno". Qui possiamo subito osservare come, ponendo il problema in termini computazionali anziché analitici, l'allievo sia costretto a misurarsi concretamente con il concetto basilare di condizione al contorno e in una maniera per lui perfettamente comprensibile. Nella fisica convenzionalmente insegnata al liceo questo concetto non appare o, al più, compare in un forma astratta che ben difficilmente lascerà qualche traccia nella mente del giovane. Orbene, il punto di partenza rappresenta la condizione iniziale della posizione, che richiederà che venga fissato un sistema di riferimento, sotto forma di coordinate (x_0, y_0) dell'origine di un riferimento cartesiano, e una posizione iniziale (x, y) del corpo. Qui verrà presa una decisione anch'essa di natura fisica: essendosi posti nella condizione di studiare le orbite di singoli corpi intorno a un pianeta molto più grande, potremo assumere che questo stia fermo e sarà quindi naturale far coincidere il centro (x_0, y_0) del sistema con il baricentro di questo. Così facendo avremo determinato la condizione iniziale del vettore posizione \mathbf{r} attraverso le due coppie di coordinate (x_0, y_0) e (x, y) . Poi dobbiamo decidere la dimensione del passo e in che direzione compierlo. Questo lo possiamo fare se sappiamo la velocità nel punto in cui ci troviamo. Potrebbe essere zero se immaginiamo semplicemente di lasciar cadere il corpo verso la terra, ma sarà invece un valore preciso se si tratterà di imprimere la giusta velocità iniziale a un satellite o a una cometa, affinché questi percorrano le orbite che ci aspettiamo. La velocità nel punto del primo passo è la velocità iniziale e la direzione sarà determinata attraverso la reciproca dimensione delle due componenti, lungo x e y , rispettivamente. Dopodiché, la dimensione del passo verrà determinata a partire dalla relazione $spazio = velocità \times tempo$, che caratterizza il concetto di velocità. In pratica, dal punto di vista computazionale, ovvero di istruzioni da impartire a un computer, dovremo scrivere qualcosa del genere

$$\begin{aligned} XPOS &= XPOS + XVEL * Dt \\ YPOS &= YPOS + YVEL * Dt \end{aligned} \tag{11.10}$$

Dove XVEL e YVEL rappresentano le componenti della velocità e Dt l'intervallo di tempo su cui abbiamo deciso di aggiustare il passo. Prima di commentare quest'ultimo elemento, per coloro che non avessero mai visto una scrittura del genere, diciamo che questo è uno dei modi per alterare – incrementare in questo caso – una variabile. In generale, scrivere $a = a + b$ significa prendere la somma dei valori di a e b e attribuire questo risultato alla variabile a . È un modo quindi per aggiornare il valore di una variabile: a destra del segno di $=$ si usa il vecchio valore di a , a sinistra c'è quello nuovo attribuito in base all'operazione fatta. Detto questo, finiamo di commentare il fattore Dt. Questo rappresenta l'intervallo di tempo nel quale viene percorso il tratto a quella velocità. Deve essere piccolo perché senon non si può ragionevolmente accettare che la velocità sia costante in quel tratto. Piccolo quanto?

11.2.4 The art of scientific computing

La risposta onesta è: bisogna vedere... Non esattamente soddisfacente come risposta per una scienza (un tempo) ritenuta "esatta". Esiste un testo fondamentale per chiunque si sia occupato o si occupi di calcolo scientifico, pubblicato la prima volta nel 1986: Numerical Recipes – The Art of Scientific Computing [14]. Quello del calcolo scientifico non è un mondo esatto. Sembra un paradosso: laddove le scienze esatte sposano la tecnologia si parla di "arte del calcolo scientifico". In parte perché ci sono metodi matematici estremamente sensibili ad ogni piccola variazione dei dati, che in certi contesti conducono all'impossibilità di risolvere problema, e in parte perché la rappresentazione in bit dei numeri è solo un'approssimazione della loro nozione matematica, questa sì esatta. Non solo, lo stesso identico calcolo eseguito su computer diversi può dare risultati differenti, come abbiamo visto nel capitolo precedente, perché non è identico l'insieme dei processi con i quali i computer manipolano l'imperfetta rappresentazione dei numeri.

Tornando alla questione dell'intervallino di tempo Dt , anch'essa dipende molto dal contesto. Il compromesso è un po' questo: se si fanno passi lunghi allora il calcolo procede veloce ma si rischia di sbagliare molto, in particolare quando la velocità cambia apprezzabilmente fra un passo e l'altro; per sbagliare meno occorre scegliere passi più brevi e quanto più questi sono brevi, tanto più il calcolo sarà accurato. Però anche qui fino a un certo punto, perché se gli intervalli diventano brevissimi si possono avere problemi di precisione nella rappresentazione dei numeri e quindi i conti possono nuovamente sbagliare, oltre a rischiare di dilatare inutilmente i tempi di calcolo, se non a renderli ingestibili.

Perché tutte queste considerazioni che possono parere anche esageratamente tecniche per il nostro contesto? Da un lato per tentare di rimettere al centro la complessità e la ricchezza del pensiero computazionale, che finisce con l'essere polverizzato in un discorso pubblico banale e partigiano, strapazzato nelle diatribe fra fazioni di apocalittici e integrati, o di "umanisti" e "scientifici". A proposito di quest'ultima dicotomia vorrei dire che chi vi crede tradisce allo stesso tempo sia la visione umanistica che quella scientifica, perché una vera visione umanistica certamente comprende e valorizza il pensiero scientifico. Il pensiero scientifico è umanistico, invece v'è ne poco in giro. Dall'altro perché questa complessità fornisce un terreno adatto per un approccio alla conoscenza di natura laboratoriale, che è quello che di fatto vivono il ricercatore, l'artista o l'artigiano, quando si misurano con la realtà nei rispettivi modi e con i rispettivi strumenti. Le incertezze del calcolo scientifico non devono certamente essere rovesciate sugli allievi come si fa nell'approccio disciplinare classico, bensì devono essere tenute presenti per essere eventualmente approfondite, in misura adeguata, qualora i loro effetti si presentino nel corso delle esplorazione.

Il codice nella formula 11.10, se rieseguito ripetutamente, disegna la traiettoria del corpo per incrementi successivi, che è proprio quello che desideriamo. Il problema tuttavia è che ci occorre conoscere la velocità in ogni punto della traiettoria ma noi questa informazione non ce l'abbiamo; soprattutto non abbiamo niente che leghi la grandezza velocità alla legge di gravitazione (Eq. 11.5), che è tutto ciò che sappiamo. In realtà nell'equazione 11.5 abbiamo l'accelerazione, che rappresenta la variazione della velocità. Quindi quello che ci occorre è un passaggio simile a quello che nel codice 11.10 ci è servito per ricostruire il percorso. Dobbiamo cioè ricostruire l'andamento della velocità per incrementi

successivi:

$$\begin{aligned} XVEL &= XVEL + XACC * Dt \\ YVEL &= YVEL + YACC * Dt \end{aligned} \quad (11.11)$$

Dove XACC e YACC rappresentano le componenti dell'accelerazione lungo le direzioni degli assi cartesiani X e Y. Questa ce l'abbiamo perché è proprio la legge di gravitazione 11.5 a darcela, che scissa nelle due componenti risulta:

$$\begin{aligned} a_x &= -\frac{GM}{r^2} \frac{r_x}{r} \\ a_y &= -\frac{GM}{r^2} \frac{r_y}{r} \end{aligned} \quad (11.12)$$

che rappresentiamo nel codice LibreLogo (vedi dopo) in questa forma:

$$\begin{aligned} XACC &= GG/R2 * DX/R \\ YACC &= GG/R2 * DY/R \end{aligned} \quad (11.13)$$

Letto così può parere complesso rispetto a quanto abbiamo visto ora. Certo, non stiamo facendo un corso di fisica ma non è questo il punto. L'importante è riconoscere i blocchi principali per afferrare i concetti importanti, intravedendo il nesso con il caso (apparentemente) più semplice del cerchio che abbiamo raccontato al *low floor*.

Prima descriviamo l'algoritmo con un pseudocodice, ovvero un codice informale che non può essere eseguito nel computer ma che serve a facilitare la comprensione degli algoritmi, scritto tenendo conto solo dei passaggi fondamentali e omettendo tutti i particolari che invece sono necessari alla macchina per potere eseguire il programma. Rammentiamo che i programmi in LibreLogo si scrivono anteponendo le procedure, in modo che, ogni volta che una di queste sia invocata, il suo codice sia già stato "letto" dalla tartaruga. Quindi il pseudocodice¹ che segue si riferisce al cosiddetto programma principale che, nel listato completo successivo inizia a partire dall'istruzione n. 134.

Data: Dati del sole e del corpo orbitante, condizioni iniziali

Result: Orbita del corpo

while *orbita non è chiusa* **do**

calcola prossimo punto con legge di gravitazione;

disegna prossimo punto;

scrive su file coordinate, velocità, accelerazioni nel prossimo punto

end

Algorithm 1: Pseudocodice per il calcolo dell'orbita di un corpo intorno al sole

¹Il pseudocodice è una sorta di linguaggio che sta à metà fra un vero linguaggio di programmazione, che può essere interpretato dal computer, e il linguaggio naturale. Le sue istruzioni non possono essere eseguite nella macchina ma servono a dare un'idea della struttura fondamentale di un algoritmo. In questi esempi abbiamo lasciato le pseudostrutture in inglese, considerato che in tutto questo ambito la lingua di riferimento è l'inglese. Del resto il significato è intuitivo. Aggiungiamo solo che il costrutto "**while** condizione **do** istruzioni **end**" esprime un ciclo, dove le istruzioni comprese fra **do** e **end** vengono ripetute fintantoché risulta vera la condizione espressa fra **while** e **do**, che nel nostro caso è "orbita non è chiusa". Alla prima ripetizione in cui l'orbita risulta chiusa il ciclo si interrompe.

92CAPITOLO 11. GIRANDO IN TONDO: DAL CERCHIO ALL'ORBITA DI HALLEY

Messo così non siamo molto lontani dal codice che esprimeva il cerchio alla Papert:

```
REPEAT [ FORWARD 1 RIGHT 1 ]
```

che in sostanza diceva²

```
while do
  | disegna prossimo punto;
end
```

Algorithm 2: Pseudocodice per il calcolo della circonferenza

Cos'hanno in comune di importante il codice per disegnare l'orbita di un corpo celeste e quello per disegnare il cerchio? Il fatto di farlo esclusivamente in base a valutazioni locali, senza impiegare la conoscenza esplicita dell'ellisse in un caso e del cerchio nell'altro, ovvero senza utilizzare le equazioni delle rispettive curve. In termini matematici, questa "valutazione locale" non vuol dire altro che risolvere l'equazione differenziale che descrive un certo fenomeno. Torneremo dopo su questo, ora era necessario solo cercare di cogliere questa corrispondenza matematica, prima di tuffarsi un poco nei particolari del codice per l'orbita di un corpo reale, che per la sua particolare natura richiede una serie di accorgimenti, i quali spostano considerevolmente in alto l'*high ceiling*. Forse anche troppo, in media, per una scuola secondaria, ma non è affatto escluso, che in una situazione laboratoriale, un insegnante e degli studenti in gamba vi si possano arrampicare. L'esempio è comunque utile per mettere in luce vari aspetti che mostrano come l'approccio computazionale consenta di avvicinarsi in maniera molto più concreta allo studio dei fenomeni fisici, sotto vari aspetti, e anche come il calcolo scientifico sia ben lontano da quel banale determinismo che taluni, intrappolati in una miope visione dicotomica fra *humanities* e scienza, vorrebbero far credere. Tant'è che l'espressione "*art of scientific computing*" è comune e pertinente. In ultimo, l'esempio è istruttivo anche per mostrare come non siano necessari strumenti sofisticati per realizzare compiti che possono essere anche considerevolmente complessi, dal punto di vista scientifico. Insomma Logo non è limitato al disegno delle casette. E per questo che abbiamo aggiunto, ad esempio, la scrittura su di un file dei dati calcolati nel processo – posizioni velocità e accelerazioni – per mostrare che LibreLogo può anche essere considerato un sistema per eseguire calcolo, vero e proprio, e non solo per fare un po' di grafica.

Nelle pagine successive riportiamo il listato del codice, ampiamente commentato, per il caso della cometa di Halley che orbita intorno al sole. Aggiungiamo poi qualche considerazione per approfondire alcuni aspetti più rilevanti. La sor-

²Qui abbiamo usato la variante "**while do** istruzioni **end**" senza condizione, che produce una ripetizione del ciclo all'infinito.

gente del codice, in formato ODT, la mettiamo a disposizione in un file a parte³, in maniera che possa essere eseguito da tutti.

```

1 ; Programma distribuito alle condizioni della GNU General Public License
2
3 ; Questo programma è Software Libero (Free Software): può essere
4 ; ridistribuito e modificato nei termini della GNU General Public
5 ; License pubblicata dalla Free Software Foundation, nella versione 3 o
6 ; una delle successive. Il testo della licenza è accessibile in
7 ; <https://www.gnu.org/licenses/licenses.it.html>.
8
9
10 ; Calcolo dell'orbita di un corpo celeste intorno al sole mediante
11 ; integrazione numerica delle equazioni del moto governate dalla legge
12 ; di gravitazione di Newton. Il problema è posto in due dimensioni e
13 ; assume che non vi siano perturbazioni da parte di altri corpi.
14 ; Il codice è aggiustato per risolvere il caso di un'orbita fortemente
15 ; eccentrica come quella della cometa di Halley.
16
17 ; La Tartaruga gioca il ruolo della cometa Il pianeta sta al centro,
18 ; che in LibreLogo ha coordinate [297.89,421.11] dove la unità di
19 ; misura è il "punto" (p). Lo spazio della pagina è
20 ; (pagg. 63–65 del Piccolo Manuale di LibreLogo)
21
22 ;      [0,  0] ----- [596,  0]
23 ;      |           |
24 ;      |           |
25 ;      |           |
26 ;      |           |
27 ;      |           |
28 ;      [0, 842] ----- [596, 842]
29
30 ; dove i numeri sono espressi nell'unità di misura del foglio, qui
31 ; denominata "punto". In Writer di LibreOffice il punto vale 1/2.83 mm
32 ; (vedi nota 54 a pagina 64). Nei calcoli di questo algoritmo tutte
33 ; le misure di lunghezza sono riportate in "punti", in ultima analisi.
34
35 ; Dichiarazione delle variabili globali, ovvero che sono "visibili"
36 ; sia nel codice principale che all'interno di ogni procedura, NEWTON,
37 ; STEP ecc.
38
39
40 GLOBAL GG, Dt, DX, DY, XPOS0, YPOS0, XPOS, YPOS, XVEL, YVEL, XACC, YACC
41
42 ; Qui seguono le procedure nelle quali abbiamo incapsulato alcune
43 ; funzionalità specifiche: NEWTON che calcola l'accelerazione in un
44 ; dato punto, STEP che valuta il prossimo passo del percorso,
45 ; WRITEPOINT scrive gli estremi dei punti via via calcolati su un
46 ; file...
47
48 ; ****
49 ; procedura NEWTON: calcola l'accelerazione nel punto di coordinate
50 ; X, Y – il risultato consiste nelle due componenti ACCX e ACCY
51 ; del'accelerazione. È qui che "c'è" la legge di gravitazione.
52
53 TO NEWTON X Y
54   GLOBAL GG, Dt, DX, DY, XPOS0, YPOS0, XPOS, YPOS, XVEL, YVEL, ~
55   XACC, YACC
56   DX = (X-XPOS0)
57   DY = (Y-YPOS0)
58   R2 = (DX**2 + DY**2)
59   R = SQRT(R2)
60   XACC = - GG / R2 * DX / R
61   YACC = - GG / R2 * DY / R
62 END
63
64 ; ****

```

³Il codice è accessibile presso l'URL <http://iamarf.ch/unifi/Halley-RK-4-AU-90-distribuibile.odt>

94CAPITOLO 11. GIRANDO IN TONDO: DAL CERCHIO ALL'ORBITA DI HALLEY

```

65 ; procedura STEP: calcola il prossimo passo con l'interpolazione di
66 ; Runge-Kutta del IV ordine. Questa interpolazione rappresenta un
67 ; modo abbastanza sofisticato per calcolare i vari punti della
68 ; traiettoria; serve a ridurre gli errori di approssimazione dovuti al
69 ; fatto di calcolare una funzione continua in un numero finito di
70 ; punti. L'algoritmo è tratto da W.H. Press et al., Numerical Recipes
71 ; - The Art of Scientific Computing, Cambridge university Press, 1992,
72 ; pp. 704-708. Il risultato consiste nelle coordinate della nuova
73 ; posizione, XPOS e YPOS, e della velocità in quel punto, XVEL e YVEL.
74
75 TO STEP
76     GLOBAL GG, Dt, DX, DY, XPOS0, YPOS0, XPOS, YPOS, XVEL, YVEL, ~
77         XACC, YACC
78
79     NEWTON XPOS YPOS
80     KX1 = Dt * XACC
81     XVEL1 = XVEL +KX1 / 2.
82     KY1 = Dt * YACC
83     YVEL1 = YVEL +KY1 / 2.
84     XPOST = XPOS +XVEL1 * Dt / 2.
85     YPOST = YPOS +YVEL1 * Dt / 2.
86
87     NEWTON XPOST YPOST
88     KX2 = Dt * XACC
89     XVEL2 = XVEL +KX2
90     KY2 = Dt * YACC
91     YVEL2= YVEL +KY2
92     XPOST= XPOS +XVEL2 * Dt / 2.
93     YPOST= YPOS +YVEL2 * Dt / 2.
94
95     NEWTON XPOST YPOST
96     KX3 = Dt * XACC
97     XVEL3 = XVEL +KX3
98     KY3 = Dt * YACC
99     YVEL3= YVEL +KY3
100    XPOST= XPOS +XVEL3 * Dt / 2.
101    YPOST= YPOS +YVEL3 * Dt / 2.
102
103    NEWTON XPOST YPOST
104    KX4 = Dt * XACC
105    XVEL4 = XVEL + KX4
106    KY4 = Dt * YACC
107    YVEL4 = YVEL +KY4
108
109    XVEL= XVEL + (KX1 + 2 * KX2 + 2 * KX3 + KX4) / 6.
110    YVEL= YVEL + (KY1 + 2 * KY2 + 2 * KY3 + KY4) / 6.
111
112    XPOS= XPOS + (XVEL1 + 2 * XVEL2 + 2 * XVEL3 + XVEL4) * Dt / 6.
113    YPOS= YPOS + (YVEL1 + 2 * YVEL2 + 2 * YVEL3 + YVEL4) * Dt / 6.
114
115 END
116
117 ; *****
118 ; procedura WRITEPOINT. Fa due cose:
119
120 ; 1) invia la tartaruga nel prossimo punto della traiettoria mediante
121 ; un'istruzione POSITION - è così che si crea il disegno
122
123 ; 2) scrive in un file tutti i valori rilevanti di ciascuno punto della
124 ; traiettoria calcolato, in particolare le due componenti della
125 ; posizione, della velocità e dell'accelerazione, ovvero tutto
126 ; ciò che viene detto soluzione dell'equazione del moto. Questi
127 ; dati possono così essere ripresi per creare rappresentazioni
128 ; grafiche con altri tipi di software o per ulteriori elaborazioni.
129
130 TO WRITEPOINT f REPCOUNT
131     GLOBAL GG, Dt, DX, DY, XPOS0, YPOS0, XPOS, YPOS, XVEL, YVEL, ~
132         XACC, YACC
133
134     POSITION [XPOS, YPOS]
135     f.write( repr(REPCOUNT) + ',' + repr(GG) + ',' + repr(Dt) + ',' ~
136         + repr(DX) + ',' + repr(DY) + ',' + repr(XPOS0) ~
137         + ',' + repr(YPOS0) + ',' + repr(XPOS) + ',' + repr(YPOS) ~
138         + ',' + repr(XVEL) + ',' + repr(YVEL) + ',' + repr(XACC) ~

```

```

139      + ', ' + repr(YACC) + '\n')
140 END
141
142 ; ****
143 ; ****
144 ; Qui inizia il programma vero e proprio , quello che abbiamo chiamato
145 ; il codice principale
146
147 ; Prima di tutto si calcolano le costanti fisiche che entrano in gioco
148 ; nella determinazione del moto. I valori sono calcolati nel sistema
149 ; di misura M.K.S (Meter , Kilogram , Second), con l'eccezione delle
150 ; distanze delle orbite perché qui, dati gli enormi valori è più
151 ; usuale l'AU (Astronomical Unit), dove 1 AU =  $1.495978707 \times 10^{11}$ 
152 ; metri, che corrisponde alla distanza media fra la terra e il sole.
153 ; Quindi, ad esempio, quando si trova che l'afelio della cometa di
154 ; Halley vale 35.08 AU, significa che la distanza massima della cometa
155 ; dal sole è pari a circa 35 volte la distanza fra la terra e il sole.
156 ; Alla fine tuttavia , tutte le misure di distanza sono trasformate
157 ; in "punti" della pagina ai fini della rappresentazione grafica .
158
159 G = 6.67E-11          ; (N*m^2/Kg^2) Costante di gravitazione
160 Ms = 1.99E30          ; (Kg) Massa del sole
161
162 Dp = 200.0            ; Afelio espresso in punti del foglio
163 rAf = 35.08            ; Afelio (AU)
164 Dt = 0.001             ; Intervallo di integrazione (ha la
165 ; dimensione del tempo)
166
167 K = Dp/rAf            ; fattore di scala: numero di punti/AU
168 GAU = G / 1.496E11**2
169 Gp = GAU * K**2        ; (N*p^2*Kg^2)
170 GG = Gp * Ms           ; Cost. inclusiva della massa solare (per
171 ; ridurre il numero di moltiplicazioni nei
172 ; cicli di integrazione)
173 eps = 0.967            ; Eccentricità dell'orbita Halley
174
175 ; Qui si apre il file dove verranno scritti gli estremi dei punti della
176 ; traiettoria via via calcolata. Si scrive subito anche la prima riga di
177 ; intestazione della tabella.
178
179 f = open('/home/arf/Didattica//CODING/Logo/orbite/halley-dt-0.001-20', ~
180      'w')
181 f.write( 'REPCOUNT, GG, Dt, DX, DY, XPOS0, YPOS0, ~
182           XPOS, YPOS, XVEL, YVEL, XACC, YACC, \n')
183
184 CLEARSEREN
185 HOME
186
187 FILLCOLOR "skyblue"
188 PENCOLOR "blue"
189 CIRCLE 400              ; Si disegna un cerchio di raggio 200 punti
190 ; nel quale verrà disegnata la traiettoria
191 ; del corpo celeste; qualora si usi un
192 ; valore di eccentricità dell'orbita pari
193 ; a 0 la traiettoria risulterà
194 ; perfettamente circolare e coinciderà con
195 ; il perimetro di questo cerchio.
196
197 FILLCOLOR "yellow"       ; colore del sole
198 PENCOLOR "yellow"
199 CIRCLE 5                 ; Date le dimensioni dell'orbita della
200 ; cometa il sole non può essere in scala
201
202 ; Qui memorizziamo le coordinate del centro della pagina, che
203 ; assumiamo come origine del sistema di riferimento e che
204 ; facciamo coincidere con la posizione del sole.
205
206 XPOS0 = POSITION[0] ; coordinate origine (centro pagina)
207 YPOS0 = POSITION[1]
208
209 HIDETURTLE               ; nasconde la Tartaruga perché assorbe
210 ; troppe risorse
211
212 ; Determinazione delle condizioni iniziali del problema

```

96CAPITOLO 11. GIRANDO IN TONDO: DAL CERCHIO ALL'ORBITA DI HALLEY

```

213 ; Posizione iniziale dove portiamo il satellite in quota
214
215 PENUM
216 POSITION [XPOS0 + rAf*K, YPOS0]
217 PENDOWN
218 XPOS = POSITION[0]
219 YPOS = POSITION[1]
220
221 ; Velocità iniziale che imprimiamo al satellite
222
223 XVEL = 0.0
224 YVEL = sqrt(GG/(rAf*K)*(1-eps)) ; Grazie alla II legge di Keplero la
225 ; velocità, ad esempio all'afelio,
226 ; risulta determinata dall'eccentricità
227 ; dell'orbita oltre che dalla
228 ; massa del sole
229
230 PENCOLOR "blue"
231 PENSIZE 1
232
233 ; Ora inizia il ciclo che disegna la traiettoria del corpo celeste.
234 ; I punti della traiettoria sono calcolati con la subroutine STEP e
235 ; vengono disegnati con la procedura WRITEPOINT. Quest'ultima
236 ; provvede anche a scaricare gli estremi di ciascuno punto
237 ; (posizione, velocità, accelerazione) in un file. Tuttavia
238 ; WRITEPOINT viene invocata solo in un numero limitato di punti,
239 ; perché la traiettoria, affinché possa essere calcolata in
240 ; maniera sufficientemente accurata, risulta composta da un
241 ; numero esagerato di punti, ai fini del disegno. In questa
242 ; implementazione, aggiustata per riprodurre orbite fortemente
243 ; eccentriche, come quella della cometa di Halley, il codice
244 ; usa solo un punto su 10000 per disegnare traiettoria, e questo
245 ; comportamento viene controllato mediante il contatore nWrite.
246 ; Si usa poi un flag (variabile di tipo sì/no), yIsNegative,
247 ; per controllare che l'orbita non venga disegnata più di una volta.
248
249 nWrite = 0
250 yIsNegative = FALSE ; Flag controllo completamento orbita:
251
252 WHILE NOT ( yIsNegative AND (YPOS-YPOS0) > 0 ) [
253   nWrite = nWrite + 1
254   IF NOT yIsNegative AND (YPOS-YPOS0) < 0 [ yIsNegative = TRUE ]
255   STEP
256   IF nWrite = 1 [
257     WRITEPOINT f, REPCOUNT
258   ]
259   IF nWrite = 10000 [ nWrite = 0 ]
260 ]
261
262 PRINT "Fatto!"

```

La prima cosa che si incontra è la licenza di software libero. L'abbiamo fatto più che altro per poterla "toccare con mano". Poi, la prima istruzione che troviamo è GLOBAL (n. 40). È buona pratica rendere il codice modulare, riducendolo a un insieme di blocchi indipendenti specializzati nell'esecuzione di singole operazioni. Questi blocchi devono poi essere in grado di comunicare fra loro mediante appositi meccanismi. Alla base di questi sta lo scambio di dati che servono a compiere le operazioni. Tali blocchi si chiamano procedure, subroutine, metodi, funzioni, in dipendenza di diversità fra linguaggi e contesti che qui non ci serve approfondire. Serve sapere comunque che se voglio far fare a una procedura (usiamo qui questo nome) ciò che sa fare occorre che le passi i dati necessari e che poi lei mi renda in qualche maniera i risultati dell'operazione. Il meccanismo normale è quello di aggiungere i nomi di tali

dati nell'istruzione che "chiama" la procedura. È quello che avevamo visto per esempio scrivendo in LibreLogo una procedura TO CASA L, dove L ad esempio potrebbe essere la misura del quadrato che forma la casa. Normalmente ciascuna procedura ha un suo spazio privato per le variabili che usa, nel senso che quando questa ha finito il suo compito le variabili per così dire vengono perse. È per questo motivo che è necessario "passarle" nella chiamata. In certi contesti è tuttavia possibile anche definire alcune variabili "globali", che significa visibili da ogni parte del programma, anche da dentro le procedure. In tal caso non c'è bisogno che queste vengano "passate". Ogni procedura ci lavora e i cambiamenti restano a disposizione di altre parti del programma. Questa non è considerata una buona pratica, specialmente quando i progetti divengono complessi, ma nel caso di codici relativamente compatti questo tipo di variabile può tornare comodo. È il nostro caso, dove abbiamo resi globali la costante di gravitazione universale inclusiva della massa del sole (GG), il passo di integrazione, ovvero quanto finemente si vogliono calcolare i successivi punti (Dt), le coordinate espresse rispetto all'origine, ovvero il centro della pagina (DX, DY), le coordinate dell'origine (XPOS0, YPOS0), le coordinate del punto corrente della traiettoria (XPOS, YPOS), le componenti della velocità (XVEL, YVEL) e quelle dell'accelerazione (XACC, YACC).

Poi vengono i codici delle varie procedure, iniziando da quella che non ne chiama nessun'altra, NEWTON per il calcolo dell'accelerazione, STEP per il calcolo dei passi della traiettoria, WRITEPOINT per il disegno e la memorizzazione dei punti su file. I commenti nel listato dovrebbero essere sufficienti, per comprendere il comportamento, eccetto che nel caso della procedura STEP dove occorre qualche precisazione. In realtà l'operazione compiuta da STEP potrebbe ridursi all'applicazione delle istruzioni 11.10 e 11.11 , che riscriviamo qui di seguito

$$\begin{aligned} XVEL &= XVEL + XACC * Dt \\ YVEL &= YVEL + YACC * Dt \end{aligned} \quad (11.14)$$

e

$$\begin{aligned} XPOS &= XPOS + XVEL * Dt \\ YPOS &= YPOS + YVEL * Dt \end{aligned} \quad (11.15)$$

Immaginando un intervento didattico, è utile partire proprio da questa versione perché consente di capire facilmente il meccanismo con cui ogni valore viene aggiornato in base al valore del precedente e in funzione di una cosa che va calcolata – l'accelerazione in un caso, la velocità nell'altro - giusto in quel punto. In questa versione si capisce bene anche l'essenza del concetto di approssimazione, perché diviene palese la seguente ambiguità: il valore ad esempio dell'accelerazione a quale punto si riferisce? Al precedente o al successivo? La risposta è che il passo fra l'uno e l'altro deve essere abbastanza breve da rendere inessenziale la differenza. Di fatto si tratta di un'approssimazione e non è affatto banale stabilire una regola valida per tutte le situazioni. Ecco dove compare l'arte nel calcolo scientifico, o, se vogliamo, dove necessita la mano dell'artigiano, che comprende in base all'esperienza e all'intuito come regolarsi. Le operazioni descritte sopra prendono il nome di interpolazione di Eulero, che concettualmente è perfettamente valida, ma all'atto dell'applicazione concreta, molto facilmente produce soluzioni distorte perché troppo sensibile agli errori di approssimazione.

Per capire bene il meccanismo immaginiamo di far fare ai bambini un cerchio nella sabbia come avevamo visto al *low floor* ma schiacciato, dicendo loro di curvare quando meno quando più, in maniera da riprodurre qualcosa di simile a un'ellisse. E proviamo a esagerare, che tanto ai bambini piace, facendo un cerchio molto schiacciato. È facile rendersi conto che se lungo i tratti per così dire piatti possiamo fare anche dei passi piuttosto lunghi, quando invece arriviamo in prossimità delle curve strette dobbiamo per forza accorciare il passo, per poter seguire l'andamento della traiettoria. L'interpolazione di Runge-Kutta del IV ordine, realizzata attraverso la procedura STEP, migliora la scelta della direzione del passo facendo come dei sondaggi aggiuntivi rispetto al semplice calcolo unico dell'interpolazione di Eulero, espresso dalle relazioni 11.14 e 11.15, tre in più, per la precisione. Noi l'abbiamo introdotta perché sennò l'orbita della cometa di Halley non viene calcolata correttamente, essendo molto eccentrica, quindi con curvatura molto accentuata in prossimità dell'afelio e del perielio. Ma per capire l'architettura generale possiamo benissimo fare riferimento all'approssimazione di Eulero.

A partire dall'istruzione 144 inizia il programma principale, dove, come prima cosa, si definiscono le costanti fisiche e geometriche in ballo nel problema. Sono tutte spiegate nel listato. Ovviamente si può giocare variandole a piacimento per riflettere sugli effetti. La prima cosa da fare è lavorare sull'eccentricità dell'orbita, attualmente fissato sul valore della cometa di Halley, pari a 0.967. Prossimo al valore limite di 1, dove l'ellisse si trasforma in una parabola e oltre 1 diventa un'iperbole. Per valori di eccentricità decrescenti l'ellisse diventa sempre meno schiacciata fino a 0 che vale per il cerchio.

Successivamente vengono determinate le condizioni iniziali, che è un passo fondamentale per la risoluzione di qualsiasi problema di fisica. Se gli allievi uscissero dalle scuole superiori con questo concetto ben chiaro in testa sarebbe già un gran risultato. L'impostazione computazionale dei problemi di fisica consentirebbe di realizzare questo obiettivo. Nel caso della soluzione di un problema di moto le condizioni iniziali si determinano attraverso la posizione e la velocità iniziali, in valore (modulo) e direzione. Nel nostro codice posizione e dimensioni dell'orbita sono aggiustate in maniera da accomodarsi adeguatamente nella pagina. L'eccentricità invece viene determinata attraverso la II legge di Keplero che lega questa alla velocità del corpo celeste nella sua orbita. In virtù di tale legge, la velocità di un corpo celeste è legata alla distanza r dal sole mediante questa relazione

$$v = \sqrt{GM\left(\frac{2}{r} - \frac{1}{a}\right)} \quad (11.16)$$

dove G la costante di gravitazione universale, M è la massa del sole e a è il semiasse maggiore dell'orbita ellittica. Per stabilire la nostra condizione iniziale assumiamo che l'orbita ellittica debba stare sdraiata con l'asse maggiore orizzontale, con il fuoco sinistro coincidente con il centro della pagina e che la cometa "parta" dall'afelio, ovvero nel punto di massima distanza, che così ci ritroviamo a destra, e sia diretta verso il basso. La posizione è determinata mediante le istruzioni 206 e 207. Per quanto riguarda la velocità usiamo il fatto che, quando la distanza dal sole coincide con l'afelio, $r = r_a$, vale la seguente relazione fra la distanza all'afelio r_a , il semiasse maggiore a e l'eccentricità ϵ :

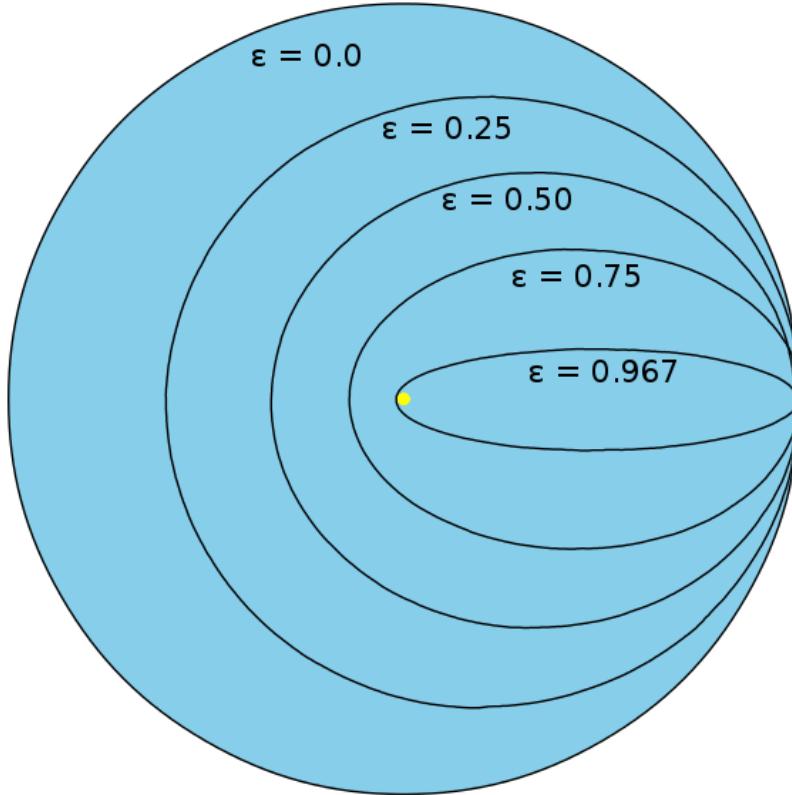


Figura 11.2: Orbite di un corpo celeste intorno al sole con diversi valori di eccentricità ϵ . Il valore $\epsilon = 0.967$ corrisponde a quello della cometa di Halley..

$$r_a = a(1 + \epsilon) \quad (11.17)$$

Questa, combinata con l'equazione 11.16 fornisce

$$v = \sqrt{\frac{GM}{r_a}(1 - \epsilon)} \quad (11.18)$$

che è esattamente l'equazione codificata nell'istruzione 225 del listato, per la componente lungo y della velocità, essendo la componente lungo x pari a 0 perché appunto assumiamo che la cometa in quel punto sia diretta esattamente in basso.

Si comprende qui come, per variare la forma dell'orbita, si possa agire sulla velocità: velocità iniziali maggiori comportano valori di ϵ inferiori, i quali a loro volta descrivono orbite sempre più circolari. Qui abbiamo dunque la possibilità apprezzare come si possa riflettere sul significato di una formula manipolando il codice software che la esprime, in una maniera che assai difficilmente può verificarsi attraverso la semplice memorizzazione della formula.

Infine, fra l'istruzione 252 e 260 si svolge il ciclo sui punti della traiettoria, dove vengono invocate successivamente le procedure STEP e WRITEPOINT, la prima che calcola il prossimo passo e la seconda che disegna e memorizza i punti calcolati. La variabile yIsNegative, usata come *flag*, ovvero come semaforo sì-no, serve a controllare che l'orbita non venga ripetuta una seconda volta. Invece la variabile nWrite, usata come contatore, serve a far sì che i punti vengano disegnati e memorizzati sempre ma solo uno ogni tanto, precisamente uno ogni 10000 in questa versione! Ciò è dovuto al fatto che affinché l'orbita venga disegnata con ragionevole precisione, questa venga calcolata con un campionamento molto fitto: in questa implementazione la Tartaruga calcola più di 14 milioni di punti! Non avrebbe senso farle scrivere tutti questi valori e il sistema grafico andrebbe nel pallone perché non possiede questa precisione. Pur tenendo traccia di un punto su diecimila, il numero di punti totale che vengono effettivamente rappresentati e memorizzati è di 1466. Noi ci siamo fermati qui. In realtà sarebbe possibile aggiungere un altro perfezionamento. Ora il campionamento è svolto in modo regolare, con un passo Dt pari a 0.001, che produce oltre 14 milioni di punti, come abbiamo visto. Il risultato ai nostri fini è soddisfacente: l'orbita viene abbastanza regolare e il calcolo si svolge in pochi minuti, anche su un computer moderatamente veloce. Ma non è detto che non si presentino problemi provando altre configurazioni. In tal caso, potremmo provare a rendere l'interpolazione adattiva, aggiustando il passo Dt in funzione della curvatura della traiettoria: dove questa è più rettilinea la Tartaruga potrebbe procedere spedita, facendo passi più lunghi, mentre dove è più curva i passi potrebbero stringersi progressivamente.

E a proposito di altre configurazioni, lo spazio per le esplorazioni è sconfinato. Ad esempio, perché non cogliere l'occasione di conoscere Ardusat, iniziando a simularne l'orbita, anche per farsi un'idea di come "volino" i satelliti intorno alla terra – una cosa ben diversa dal caso della cometa! Ardusat⁴ è un cosiddetto "nanosatellite", un cubo di 10 cm di lato che pesa un chilo – un cartone di latte più o meno – equipaggiato con una particolare versione di Arduino, basata su 16 microprocessori e uno di controllo più una serie di 25 sensori diversi, fra cui una videocamera uno spettrometro ottico, un contatore Geiger, un sensore a infrarossi ecc. È concepito per essere usato da studenti, insegnanti o appassionati che possono progettare esperimenti, applicazioni che impiegano dati spaziali o addirittura giochi. Gli utenti possono progettare il software delle loro applicazioni sulle normali schede Arduino per poi sperimentarlo su un clone terrestre di Ardusat. Quando tutto va bene il software viene inviato a Ardusat per eseguire le operazioni previste, scaricando i dati sul browser degli utenti, quando il satellite è visibile dalle apposite stazioni terrestri. Al termine del periodo di lavoro, che può durare fino a una settimana, tutti dati vengono scaricati e inviati all'utente. In un programma di scienze, potrebbe essere estremamente stimolante esplorare Ardusat, partendo per esempio dalla simulazione della sua orbita con Logo.

Tutte considerazioni che possono essere estremamente istruttive e che possono essere svolte per dare una corretta valutazione di cosa sia oggi la scienza, in particolare di quanto e come gli aspetti computazionali influiscano su di essa. In realtà la riflessione è possibile di un ulteriore e significativo approfondimen-

⁴Informazioni dettagliate si trovano ad esempio http://www.dk3wn.info/sat/afu/sat_ardusat.shtml e <https://en.wikipedia.org/wiki/ArduSat>.

to, secondo la linea di pensiero illustrata da B. Sherin in un articolo intitolato *"A comparison of programming languages and algebraic notation as expressive languages for physics"* [10].

Sherin nel suo articolo affianca – si badi bene, non contrappone – la *programming-physics* all'*algebra-physics*. L'impianto teorico del suo lavoro si fonda sul ruolo giocato dalle rappresentazioni strumentali e simboliche che supportano la conoscenza nella formazione stessa di tale conoscenza – concetti sviluppati da altri autori, ad esempio da Weintrop e Wilensky [8]. L'autore documenta i propri argomenti con alcuni esercizi di fisica presentati sia nella forma algebrica convenzionale che nella forma computazionale. Per quest'ultima riferisce di esperienze svolte con l'ambiente Boxer, sviluppato da Di Sessa e altri autori [6], sulla base di Logo. Con questo lavoro l'autore sostiene la tesi che sistemi di rappresentazione diversi influiscono in modo differente sui meccanismi del pensiero e possono indurre un diverso tipo di comprensione dei medesimi fenomeni. In capo a un'analisi minuziosa di esperimenti didattici, condotti su gruppi di studenti sia mediante l'*algebra-physics* che la *programming-physics*, giunge alla conclusione per cui con la conoscenza algebrica si tende a enfatizzare gli equilibri mentre con quella computazionale si è portati a comprendere meglio gli aspetti dinamici. È estremamente interessante la prospettiva nella quale Sherin pone questa conclusione. Non si tratta, dice, di giudicare l'effetto di un metodo o dell'altro secondo una singola metrica e di confrontarli sulla base di tale metrica – ovvero non si tratta di stabilire quale sia "meglio" - bensì di accettare, comprendere e utilizzare proficuamente il fatto che il nuovo paradigma offre una mutata visione della conoscenza degli stessi fenomeni e di come, in ultima analisi, la cosa più sensata da fare sia quella di affiancare questa nuova forma di conoscenza a quelle preesistenti. E non si può evitare di osservare che la nuova prospettiva computazionale – qui nel senso della *programming-physics* di Sherin, possa essere di grande giovamento per la comprensione dei fenomeni fisici. Infatti lo strumento matematico costituisce indubbiamente il fondamento imprescindibile delle scienze di base – il linguaggio che consente di porre domande alla natura, per dirla con Galileo - ma il processo con il quale un giovane giunge a creare senso compiuto a partire da un linguaggio formale è molto complesso e faticoso. Pochi studenti arrivano ad apprezzare il formalismo matematico come uno strumento utile per comprendere e esprimere pensieri sul mondo fisico o altro. Per la grande maggioranza i formalismi matematici rappresentano al più una quantità di regole da applicare a memoria negli specifici contesti creati dalla scuola: qual era la formula da usare qui...? A questo proposito è interessante ricordare un noto articolo scritto da Enrico Persico [9], maestro di Enrico Fermi, dove ci si domandava cosa non andasse con quella studentessa che procedeva come una locomotiva quando sciorinava le equazioni di Maxwell alla lavagna ma che non sapeva dire perché, con quel certo valore di corrente, una lampadina si sarebbe fulminata – non a caso Sherin rileva esattamente lo stesso problema a pag. 43 del suo lavoro, e proprio a proposito delle equazioni di Maxwell. La questione della comprensione dei fenomeni attraverso il linguaggio matematico non è, e non da ora, semplice. È esattamente qui che il "nuovo" approccio computazionale, nel quale peraltro vengono declinati settori sempre più ampi della fisica e delle altre scienze, può venire in aiuto. Infatti, l'approccio computazionale induce ad analizzare e scomporre i fenomeni fisici nella dimensione temporale, enfatizzandone così la natura dinamica, spesso più accessibile all'intuizione. Non solo, l'analisi computazionale costringe ad utilizzare precisi valori numerici

102CAPITOLO 11. GIRANDO IN TONDO: DAL CERCHIO ALL'ORBITA DI HALLEY

da assegnare ai parametri fisici coinvolti e questa è una pratica che induce più facilmente gli studenti a ricavare un senso da ciò che studiano.

Ora che abbiamo tirato in ballo l'articolo di Sherin vale la pena di specificare che è proprio grazie a questo lavoro che siamo giunti dal "cerchio alla Papert" alle orbite dei corpi celesti. Sì perché all'inizio ci eravamo messi a riprodurre in Logo gli esercizi esemplificati da Sherin, nel modo più semplice possibile. Riportiamo la progressione qui di seguito.

```
1 ; Caduta di un grave con accelerazione di gravità costante
2
3 CLEARSCREEN
4 HOME
5
6 PENUP
7 FORWARD 350
8 RIGHT 180
9 PENDOWN
10
11 XPOS = 0.0
12 VEL = 0.0
13 ACC = 9.8
14 DVEL = ACC
15
16 REPEAT 10 [
17   VEL = VEL + DVEL
18   XPOS = XPOS + VEL
19   PENDOWN
20   CIRCLE 5
21   PENUP
22   FORWARD VEL
23 ]
24
25 HIDEURTLE
```

```
1 ; Caduta di un grave con accelerazione di gravità costante
2 ; in presenza di resistenza dell'aria
3
4 CLEARSCREEN
5 HOME
6
7 XPOS = 0.0
8 VEL = 0.0
9 ACC = 9.8
10 RES = 0.0
11
12 G = 9.8
13 M = 10.0
14 K = 2.0
15
16 PENUP
17 FORWARD 350
18 RIGHT 180
19 PENDOWN
20
21 REPEAT 20 [
22   RES = VEL * K
23   ACC = G - RES / M
24   VEL = VEL + ACC
25   XPOS = XPOS + VEL
26   PENDOWN
27   CIRCLE 5
28   PENUP
29   FORWARD VEL
```

```

30      ]
31
32 HIDETURTLE

```

```

1 ; Andamento nel tempo del moto di un grave appeso a una molla
2 ; (asse x: tempo)
3
4 CLEARSEREN
5 HOME
6
7 XPOS = 0.0
8 VEL = 0.0
9 ACC = 9.8
10 RES = 0.0
11
12 G = 9.8
13 M = 10.0
14 KA = 0.5
15 K = 1.0
16
17 PENUP
18 FORWARD 250
19 RIGHT 180
20 PENDOWN
21
22 REPEAT [
23   RES = KA * VEL
24   SPR = K * XPOS
25   ACC = G - RES / M - SPR / M
26   VEL = VEL + ACC
27   XPOS = XPOS + VEL
28   PENDOWN
29   CIRCLE 5
30   PENUP
31   FORWARD VEL
32   LEFT 90 FORWARD 3 RIGHT 90
33   ]
34
35 HIDETURTLE

```

```

1 ; Caduta di un grave dallo spazio
2
3 CLEARSEREN
4 HOME
5
6 FILLCOLOR "BLUE"
7 CIRCLE 70
8 FILLCOLOR "LIME"
9 XPOS0 = POSITION[0]
10 YPOS0 = POSITION[1]
11
12 PENUP
13 FORWARD 350
14 RIGHT 180
15 PENDOWN
16 HIDETURTLE
17
18 XPOS = POSITION[0]
19 YPOS = POSITION[1]
20 VEL = 0.0
21 KG = 9800
22
23 REPEAT [
24   DY = YPOS-YPOS0
25   VEL = VEL + KG / DY**2

```

104CAPITOLO 11. GIRANDO IN TONDO: DAL CERCHIO ALL'ORBITA DI HALLEY

```

26   YPOS = YPOS + VEL
27   PENDOWN
28   CIRCLE 5
29   IF YPOS0-YPOS < 10 [ BREAK ]
30   PENUP
31   POSITION [XPOS,YPOS]
32   ;FORWARD (VEL)
33   ]
34   FILLCOLOR "RED"
35   CIRCLE 10
36   HEADING 0
37   FORWARD 5
38   LABEL "
39
40 HIDETURTLE

```

AAAAAAAHH...!!!!"

```

1 ; Legge gravitazionale di Newton in 2D
2 ; È un esempio in germe. Per una corretta e proficua realizzazione
3 ; didattica occorre entrare nel merito dei valori delle costanti
4 ; fisiche implicate, masse, distanze, costante gravitazionale,
5 ; fattori di scala per la rappresentazione grafica. In questo
6 ; esempio ho utilizzato valori numerici arbitrari. Interessava
7 ; vedere come può funzionare...
8
9 ; La Tartaruga gioca il ruolo del satellite
10 ; Il pianeta sta al centro, che in LibreLogo ha coordinate
11 ; [ 297.89, 421.11 ]
12
13 CLEARSCREEN
14 HOME
15
16 FILLCOLOR "BLUE"
17 CIRCLE 20
18 FILLCOLOR "LIME"
19 XPOS0 = POSITION [0]
20 YPOS0 = POSITION [1]
21 SHOWTURTLE
22
23 ; Determinazione delle condizioni iniziali
24
25 ; Posizione iniziale...
26 ; con queste istruzioni la Tartaruga-satellite si dirige nella
27 ; posizione iniziale lo fa alla Papert, con comandi "sintonici":
28 ; sale di 100 e poi si mette a testa in giù (fatto inessenziale
29 ; perché poi la dirigiamo con comandi diretti tramite POSITION)
30
31 PENUP
32 HEADING 9h
33 FORWARD 0 ; qui si potrebbe spostare lateralmente
34 HEADING 0
35 FORWARD 100 ; qui sale di quota
36 RIGHT 180
37 PENDOWN
38
39 XPOS = POSITION [0]
40 YPOS = POSITION [1]
41
42 ; Velocità iniziale...
43 ; dove aggiustando componenti lungo X e Y decidiamo in che
44 ; direzione lanciare la Tartaruga-satellite (molto divertente
45 ; giocare con questo...)
46
47 ; Con XVEL= 10.0 e YVEL=0.0 si azzecca un'orbita circolare
48 ; Con XVEL= 5.0 e YVEL=0.0 viene l'orbita di tipo caotico mostrata
49 ; nel post https://iamarf.org/2017/04/19/dove-la-tartaruga-di-seymour-papert-impara-la-legge-di-gravitazionale-di-newton/
50
51 XVEL = 10.0
52 YVEL = 0.0

```

```

55 KG = 9800      ; qui dentro ci sarebbero le masse e la costante
56 ; gravitazionale...
58
59 REPEAT 50 [
60   DX = (XPOS-XPOS0)
61   DY = (YPOS-YPOS0)
62   R2 = (DX**2 + DY**2)
63   R = SQRT(R2)
64   ACC = KG / R2
65   XVEL = XVEL - ACC * DX / R
66   YVEL = YVEL - ACC * DY / R
67   XPOS = XPOS + XVEL
68   YPOS = YPOS + YVEL
69
70 PENDOWN
71 CIRCLE 5
72 IF SQRT((XPOS0-XPOS)**2 + (YPOS0-YPOS)**2) < 10 [ BREAK ]
73 PENUP
74 POSITION [XPOS, YPOS]
75 ]
76 FILLCOLOR "RED"
77 CIRCLE 10
78
79 HEADING 0
80 FORWARD 5
81 LABEL "          AHI!!!"
82
83 HIDETURTLE

```

Sono codici grezzi, con quasi nessun commento, eccetto l'ultimo, e senza un'accurata identificazione di ciascuna variabile e delle sue unità di misura. Li mettiamo a disposizione affinché possano essere completati e variati dal lettore, un po' come i disegni da colorare per i bambini...

11.3 Conclusione

In questo capitolo la dimensione verticale è ancora più estesa rispetto a quella del capitolo precedente, svolto intorno all'esplorazione di Marta. Qui siamo partiti proprio da un'esperienza che ha avuto luogo in una scuola primaria, dove la maestra Antonella ha coinvolto i suoi bambini in un'esplorazione del cerchio ispirata al pensiero di Papert. Le foto che Antonella ci ha regalato ci hanno come portati in classe. Ci siamo poi serviti delle parole di Papert stesso per commentare come in tale attività fisica si annidi un concetto matematico importante, una "idea potente", quella del calcolo differenziale. Un modo matematico per sviluppare una forma geometrica (in realtà anche molto più di questo) utilizzando solamente "informazioni locali". Ed è proprio Papert che in quel brano collega la questione a quella del calcolo dei movimenti dei corpi celesti, secondo la legge di gravitazione universale di Newton. Nella seconda parte del capitolo ci siamo quindi mossi per esplorare quanto alto possa essere il cielo di Logo, anche in una versione semplificata come quella di LibreLogo. Ed abbiamo preso così sul serio il proposito di andare direttamente nello spazio, provando a calcolare l'orbita di un corpo celeste e scegliendo quella della cometa di Halley, un po' per poterci meravigliare di andare così lontano, seppur nello spazio di un foglio di carta, e un po' per metterci alla prova, perché le orbite delle comete sono molto schiacciate, ovvero eccentriche, e quindi più insidiose da calcolare.

Nel fare tutto ciò abbiamo potuto esplorare aspetti computazionali e di calcolo scientifico niente affatto banali. Anzi, diciamo anche arditi, se li vogliamo pensare calati in una classe di scuola secondaria superiore. Effettivamente, un esercizio del genere non starebbe male nel corso di fisica generale al primo anno di università. Ma questo non significa che non vi possano essere ragazzi più giovani in grado di comprendere un percorso del genere, se adeguatamente seguiti dall'insegnante. Ho conosciuto un ragazzo al IV anno del liceo scientifico, molto vicino a me, al quale davo piccoli incarichi di calcolo scientifico che servivano nel mio lavoro di ricerca. Ebbene, questo ragazzo risolveva questi problemi prima e meglio dei miei giovani collaboratori, assegnisti e dottorandi. La scuola dovrebbe essere in grado di dare sostegno a tutti coloro che esulano dalle medie statistiche, superando il concetto "monodimensionale" e banale di performance, sul raggiungimento di determinate competenze o altro, ma occupandosi delle diversità e mettendole a frutto. Non è infatti raro che ingegni particolarmente creativi finiscano, in questa scuola ancora largamente tayloristica, fra i "meno bravi".

Infine, abbiamo commentato l'aspetto computazionale dell'esercizio rifacendoci al lavoro di Sherin [10], autore di un interessante approfondimento sulle differenze fra quella che chiama fisica algebrica, insegnata usualmente, e fisica computazionale. In sostanza, l'approccio computazionale ha il vantaggio di avvicinare maggiormente i ragazzi ai fenomeni, obbligandoli a seguirne gli aspetti dinamici, far capire l'importante concetto di condizione al contorno, che qualsiasi problema di fisica richiede di affrontare, rendersi maggiormente consapevoli della realtà delle grandezze fisiche in gioco, essendo obbligati a mettervi dei numeri. Abbiamo poi chiuso il capitolo riportando alcuni codici tratti dagli esempi presentati da Sherin nel suo lavoro, in modo che chiunque possa svilupparli e rifinirli, in maniera analoga a quanto abbiamo fatto con il calcolo dell'orbita della cometa di Halley.

Questo capitolo è anche una risposta ai "detrattori del pensiero computazionale", che duellano nei social network con gli altrettanto sgradevoli entusiasti del "codice facile". Le polemiche che si sviluppano in questi contesti si centrano sovente su una visione automatizzata delle pratiche di coding, come se queste si riducessero a banali sequenze di comandi, inducendo gli allievi all'esecuzione passiva di procedure automatizzate. Ora, io non sono in grado di affermare che le pratiche di coding possano essere applicate con risultati significativi anche in contesti umanistici - penso di sì ma non ho gli elementi per fare affermazioni certe - ma sicuramente, per qualsiasi giovane che sia interessato all'ambito delle discipline STEM, oggi il coding è uno strumento fondamentale. E lo è non solo dal punto di vista strumentale ma profondamente culturale, perché non vi è dominio scientifico che, da più di mezzo secolo a questa parte, non si sia ampiamente esteso grazie alla prospettiva computazionale. Nascondere questo agli studenti che andranno a iscriversi ai corsi di laurea scientifici è una grave omissione da parte della scuola.

Capitolo 12

Appendice

Qui raccogliamo i listati dei codici usati per costruire alcune delle figure usate nel manuale. Naturalmente, questi codici talvolta possono contenere dei costrutti che non sono stati ancora affrontati nel punto del testo cui appare la figura corrispondente. Non importa se non si capisce tutto subito, il lettore può sempre tornarci successivamente, quando avrà sarà più esperto. Li mettiamo a disposizione perché può essere interessante e utile per vedere che cosa si può fare in pratica.

I listati sono commentati. In LibreLogo i commenti si ottengono preponendo un punto e virgola: in qualsiasi riga, tutto quello che segue il punto e virgola non viene eseguito ma serve solo a rendere più facile da leggere il codice. Questo significa che se salviamo il codice di una di queste figure, così com'è, con tutti commenti, questo può essere seguito per produrre la figura.

È molto importante inserire nel codice codice commenti chiari e accurati, sia per rileggerlo più facilmente molto tempo dopo, sia per facilitare la collaborazione con altre persone. Scrivendo codice, specie quando si è acquisita una certa confidenza, è facile farsi prendere la mano, cercando di arrivare quanto più velocemente al risultato desiderato. È bene invece controllarsi, imponendosi di documentare adeguatamente i lavori man mano che si procede. Tanto più si aspetta quanto più sarà faticoso andare a commentare il lavoro fatto, sia per la mole che per la maggiore difficoltà a ricordare i particolari. Naturalmente questo vale per il codice destinato durare un certo tempo e ad essere condiviso, come potrebbe essere il caso di quello scritto per costruire alcune delle figure di questo manuale. Non certo per piccoli frammenti estemporanei. Il codice riportato nelle seguenti pagine è un po' al limite. Ma lo abbiamo commentato per mostrare la buona pratica, ivi inclusa l'apposizione di un'intestazione che riporti anche il nome dell'autore, il numero della versione e la data. Intestazione che andrà aggiornata con eventuali successive modifiche.

```
1 ; FIGURA 1 (torna alla figura)
2 ; Le coordinate della patoria alla figuragina
3 ; Versione 1.
4 ; A.R. Formiconi
5 ; 25 luglio 2016
6
7
8 ; Versione scritta come viene, senza particolari ottimizzazioni
```

```

9
10 ; Predisponde tutto
11
12 CLEARSEREN
13 HOME
14 A = 200 ; lato corto del rettangolo che
15 ; rappresenta la pagina
16 B = 282 ; lato lungo
17 FILLCOLOR [230, 230, 230] ; fissa a grigio chiaro il colore
18 ; di riempimento
19 PENUP
20 FORWARD B/2 ; si dirige verso l'angolo in alto
21 ; a sinistra
22 HEADING 9h
23 FORWARD A/2
24 HEADING 3h ; e punta verso destra (ore 3)
25 PENDOWN
26 ; disegna il rettangolo, ruotando in senso orario
27
28 FORWARD A
29 RIGHT 90
30 FORWARD B
31 RIGHT 90
32 FORWARD A
33 FILLCOLOR [230, 230, 230] ; riempie il rettangolo
34 PENDOWN
35 PENUP
36 RIGHT 90
37 FORWARD B ; si posiziona nell'angolo in alto a
38 ; sinistra del foglio
39 ; ripete il giro marcando gli angoli e ponendo le scritte
40
41 HEADING 3h ; angolo alto destro
42 FILLCOLOR [50, 50, 50] ; fissa un grigio più scuro per i
43 ; cerchietti
44 FORWARD A
45 PENDOWN
46 CIRCLE 5
47 PENUP
48 FORWARD 10 LEFT 90 FORWARD 15
49 LABEL "[PAGESIZE[0], 0]"
50 BACK 15 RIGHT 90 BACK 10
51
52 RIGHT 90 ; angolo basso destro
53 FORWARD B
54 PENDOWN
55 CIRCLE 5
56 PENUP
57 FORWARD 15 LEFT 90 FORWARD 30 LEFT 90
58 LABEL "[PAGESIZE[0], PAGESIZE[1]]"
59 LEFT 90 FORWARD 30 RIGHT 90 FORWARD 15
60 RIGHT 180
61
62 RIGHT 90 ; angolo basso sinistro
63 FORWARD A
64 PENDOWN
65 CIRCLE 5
66 PENUP
67 RIGHT 90
68 FORWARD 15
69 BACK 15
70 LABEL "[0, PAGESIZE[1]]"
71 FORWARD 15
72
73 FORWARD B ; angolo altro sinistro
74 PENDOWN
75 CIRCLE 5
76 PENUP
77 FORWARD 15
78 LEFT 90 FORWARD 10 RIGHT 90
79 LABEL "[0, 0]"
80 BACK 15
81 PENDOWN
82

```

83 HIDETURTLE	; mando la tartaruga a dormire
-----------------	--------------------------------

```

1 ; FIGURA 2 (Torna alla figura)
2 ; L'effetto dell'istruzione POSITION e le coordinate della pagina
3 ; Versione 1.
4 ; A.R. Formiconi
5 ; 25 luglio 2016
6
7 ; Versione scritta come viene, senza particolari ottimizzazioni
8 ; Predispone tutto
9
10 CLEARSCREEN
11 HOME
12 A = 200 ; lato corto del rettangolo che
13 ; rappresenta la pagina
14 B = 282 ; lato lungo
15 FILLCOLOR [230, 230, 230] ; fissa a grigio chiaro il colore di
16 ; riempimento
17 PENUP
18 FORWARD B/2 ; si dirige verso l'angolo in alto
19 ; a sinistra
20 HEADING 9h
21 FORWARD A/2
22 HEADING 3h ; e punta verso destra (ore 3)
23 PENDOWN
24
25 ; disegna il rettangolo, ruotando in senso orario
26
27 FORWARD A
28 RIGHT 90
29 FORWARD B
30 RIGHT 90
31 FORWARD A
32 FILL ; riempie il rettangolo
33 PENUP
34 FORWARD B ; si posiziona nell'angolo in alto
35 ; a sinistra del foglio
36
37 ; ripete il giro marcando gli angoli e ponendo le scritte
38
39 HEADING 3h ; angolo alto destro
40 FILLCOLOR [50, 50, 50] ; fissa un grigio più scuro per i
41 ; cerchietti
42 FORWARD A
43 PENDOWN
44 CIRCLE 5
45 PENUP
46 FORWARD 10 LEFT 90 FORWARD 15
47 LABEL "[PAGESIZE[0], 0]"
48 BACK 15 RIGHT 90 BACK 10
49
50 RIGHT 90 ; angolo basso destro
51 FORWARD B
52 PENDOWN
53 CIRCLE 5
54 PENUP
55 FORWARD 15 LEFT 90 FORWARD 30 LEFT 90
56 LABEL "[PAGESIZE[0], PAGESIZE[1]]"
57 LEFT 90 FORWARD 30 RIGHT 90 FORWARD 15
58 RIGHT 180
59
60 RIGHT 90 ; angolo basso sinistro
61 FORWARD A
62 PENDOWN
63 CIRCLE 5
64 RIGHT 90
65 PENUP
66
67 | PENDOWN
68

```

```

69 | BACK 15
70 | LABEL "[0, PAGESIZE[1]]"
71 | FORWARD 15
72
73 | FORWARD B ; angolo altro sinistro
74 | PENDOWN
75 | CIRCLE 5
76 | PENUP
77 | FORWARD 15
78 | LEFT 90 FORWARD 10 RIGHT 90
79 | LABEL "[0, 0]"
80 | BACK 15
81 | PENDOWN
82
83 ; disegna il percorso della tartaruga in seguito all'istruzione
84 ; \POSITION [350,320]
85
86 HOME ; mi posiziono al centro
87 PENUP ; senza disegnare...
88 POSITION [298, 430] ; vado dove voglio piazzare...
89 HEADING 0 ; l'etichetta con le...
90 LABEL "[298, 421]" ; coordinate del centro
91 HOME ; torno al centro e...
92 PENDOWN ; disegno...
93 CIRCLE 5 ; il cerchietto centrale
94 FILLCOLOR "green" ; fisso il colore verde per la
95 POSITION [350,320] ; tartaruga, che mostrerà
96 P = POSITION ; applico POSITION [350,320]
97 H = HEADING ; memorizzo tale posizione...
98 PENUP ; e direzione
99 POSITION [300,320] ; alzo la penna
100 HEADING 0 ; mi sposto un po' per piazzare...
101 POSITION P ; orientata correttamente
102 LABEL "[350, 320]" ; l'etichetta
103 HEADING H ; ritorno alla posizione e
104 ; direzione...
105 ; per lasciarvi visibile la
106 ; tartaruga

```

```

1 ; FIGURA 3 (Torna alla figura)
2 ; I riferimenti per l'istruzione HEADING
3 ; Versione 1.
4 ; A.R. Formiconi
5 ; 25 luglio 2016
6
7
8 ; Versione scritta in maniera più ordinata e strutturata, facendo
9 ; uso delle "procedure". La strutturazione del software di questo
10 ; tipo consente di rendere il codice più modulare e più mantenibile.
11
12 ; Prima vengono le subroutine e alla fine il programma vero e
13 ; proprio. Prima vanno messi i frammenti più elementari, in maniera
14 ; che LibreLogo li legga per primi. Questo serve perché quando nel
15 ; codice viene citata una subroutine, questa deve essere già stata
16 ; analizzata da LibreLogo.
17
18
19 ; Subroutine BR per disegnare un segmento lungo 10 pt dalla
20 ; posizione e lungo la direzione corrente senza muoversi
21 ; (come risultato finale)
22
23 ; Parametri:
24 ;     P: posizione corrente
25 ;     H: direzione corrente
26
27 TO BR P H
28     FORWARD 10
29     POSITION P
30     HEADING H
31 END

```

```

32 ; Subroutine TARROW per disegnare la punta di una freccia
33
34
35 TO TARROW
36     P = POSITION
37     H = HEADING
38     LEFT 160
39     BR P H
40     RIGHT 160
41     BR P H
42 END
43
44 ; Subroutine LBA scrivere il testo contenuto in T in vetta al segmento
45 ; di lunghezza L e ruotato dell'angolo A
46 ; Parametri:
47 ;     L: lunghezza del segmento
48 ;     A: angolo di rotazione del segmento
49 ;     T: testo da scrivere nell'etichetta
50
51 TO LBA L A T
52     PENUP
53     FORWARD L/10 + L/5 * SIN A*PI/180
54     H = HEADING
55     HEADING 0
56     LABEL T
57     PENDOWN
58 END
59
60 ; Subroutine LB scrivere il testo contenuto in T in una posizione
61 ; determinata in coordinate polari rispetto alla posizione corrente,
62 ; mediante la distanza L l'angolo A.
63
64 ; Parametri:
65 ;     L: distanza
66 ;     A: angolo
67 ;     T: testo da scrivere nell'etichetta
68
69 TO LB L A T
70     P0 = POSITION
71     PENUP
72     S = SIN A*PI/180
73     C = COS A*PI/180
74     POSITION [P0[0] + L * C, P0[1] - L * S]
75     HEADING 0
76     LABEL T
77     POSITION P0
78     HEADING 0
79     PENDOWN
80 END
81
82 ; Subroutine ARROW per disegnare, a partire dalla posizione corrente,
83 ; una freccia di lunghezza L, inclinata di un angolo A, e in vetta una
84 ; etichetta con il testo contenuto in T
85 ; Parametri:
86 ;     L: lunghezza della freccia
87 ;     A: angolo di rotazione della freccia
88 ;     T: testo da scrivere nell'etichetta
89
90 TO ARROW P0 A0 L A T
91     PENDOWN
92     HEADING A
93     FORWARD L
94     TARROW
95     LBA L A T
96     PENUP
97     POSITION P0
98     HEADING A0
99     PENDOWN
100 END
101
102 ; Questo è il programma vero e proprio che, come si vede, grazie al
103 ; ricorso alle procedure, è abbastanza conciso.
104
105 CLEARSCREEN           ; cancella il foglio

```

```

106 HOME ; mando a casa la tartaruga
107 HIDE TURTLE ; faccio il disegno senza vedere i
108 ; la tartaruga
109 FILLCOLOR [230, 230, 230] ; fisso il riempimento a un grigio scuro
110 CIRCLE 5 ; disegno un cerchietto nella posizione
111 ; centrale
112 P0 = POSITION ; memorizzo tale posizione iniziale...
113 A0 = HEADING ; e anche la direzione iniziale
114 L = 150 ; faccio la freccia lunga 150 pt
115 A = 60 ; e la voglio inclinata di 60 gradi
116
117 PENSIZE 1 ARROW P0 A0 L 0 "HEADING 0" ; freccia verticale
118 PENSIZE 0.5 ARROW P0 A0 L A "HEADING 30" ; freccia a 60 gradi
119 PENSIZE 1 ARROW P0 A0 L 90 "HEADING 90" ; freccia orizzontale
120 PENSIZE 0.5
121 ELLIPSE [L/3, L/3, 0, A, 3] ; arco di cerchio piccolo
122 LB L/4 A "30 gradi" ; etichetta "30 gradi"
123 ELLIPSE [L*2, L*2, 0, 90, 3] ; arco di cerchio grande

```

```

1 ; FIGURA 4 (Torna alla figura)
2 ; Esempi di spessore del tratto
3 ; Versione 1.
4 ; A.R. Formiconi
5 ; 25 luglio 2016
6
7 ; In questo esempio si illustra l'impiego dell'istruzione REPEAT,
8 ; per realizzare i "cicli" ("loop")
9
10 CLEARSCREEN
11 HOME
12 RIGHT 90
13 PENCOLOR "BLACK"
14
15 REPEAT 10 [
16   PENWIDTH REPCOUNT
17   FORWARD 100
18   PENUP FORWARD 50
19   HEADING 0h
20   LABEL "PENWIDTH " + STR REPCOUNT-1
21   HEADING 3h
22   BACK 50
23   PENUP
24   RIGHT 90
25   FORWARD 20
26   RIGHT 90
27   FORWARD 100
28   LEFT 180
29   PENDOWN
30 ]
31 PENWIDTH 0
32 HIDE TURTLE

```

Bibliografia

- [1] Munari Bruno. *Fantasia*. Laterza, Roma, 28 edition, 2017.
- [2] Reggini Horacio C. *Logo: ali per la mente*. Mondadori, Milano, 1984.
- [3] Emma Castelnuovo. *L'officina matematica – ragionare con i materiali*. Edizioni La Meridiana, Molfetta, 2 edition, 2008.
- [4] Lewis Colleen. How programming environment shapes perception, learning and goals: Logo vs. scratch. <http://ims.mii.lt/ims/konferenciju-medziaga/SIGCSE'10/docs/p346.pdf>, 2015. Accessed: 2017-08-13.
- [5] Weintrop David. Comparing block-based, text-based, and hybrid block-s/text programming environments in introductory computer science classes. <http://dweintrop.github.io/papers/Weintrop-diss-4pager.pdf>. Accessed: 2017-08-13.
- [6] Abelson Harold e Ploger D. DiSessa Andrea A. An overview of boxer. *Journal of Mathematical Behavior*, 10:3–15, 1991.
- [7] Weintrop David e Wilensky U. To block or not to block, that is the question: Students' perceptions of blocks-based programming. http://dweintrop.github.io/papers/Weintrop_Wilensky_ICER_2015.pdf, 2015. Accessed: 2017-08-13.
- [8] Weintrop David e Wilensky U. Using commutative assessments to compare conceptual understanding in blocks based and text based programs. http://dweintrop.github.io/papers/Weintrop_Wilensky_ICER_2015.pdf, 2015. Accessed: 2017-08-13.
- [9] Persico Enrico. Che cos'è che non va? *Giornale di Fisica*, 1:64–67, 1956.
- [10] Sherin Bruce L. A comparison of programming languages and algebraic notation as expressive languages for physics. *Int. Journal of Computers for Mathematical Learning*, 6:1–61, 2001.
- [11] László Németh. Esempi di uso librelogo (ungherese). <http://www.numbertext.org/logo/logofuzet.pdf>. Accessed: 2017-08-13.
- [12] László Németh. Manuale di comandi di librelogo. https://help.libreoffice.org/Writer/LibreLogo_Toolbar/it. Accessed: 2017-08-13.

- [13] Seymour Papert. *Mindstorms, Children, Computers, and Powerful Ideas.* Basic books, New York, 2 edition, 1993.
- [14] Vetterling William T. e Flannery Brian P. Press William H., Teukolsky Saul A. *Numerical Recipes – The Art of Scientific Computing.* Cambridge University Press, Cambridge, Masssachusetts, 2007.
- [15] Lakó Viktória. Manuale di comandi di librelogo. http://szabadszoftver.kormany.hu/wp-content/uploads/librelogo_oktatasi_segedanyag_v4.pdf. Accessed: 2017-08-13.