PROGETTAZIONE e CODING

Daniele Grosso

daniele.grosso.genova@gmail.com

Pensiero computazionale e CODING

- "pensiero computazionale" processo mentale per la risoluzione di problemi
- "coding" implementazione (codifica in un linguaggio di programmazione)

Gauss a scuola, il suo insegnante ordinò agli alunni di calcolare **la somma di tutti i numeri da 1 a 100**

$$Somma = \frac{1+100}{2}100 = \frac{101}{2}100 = 5050$$

```
#include <stdio.h>
#include <stdlib.h>
int sumnum (int somma, int n);
main () {
int n, i, risult, somma;
somma=100;
risult= sumnum (n, somma);
printf ("Il risultato e' pari a %d\n", risult);
system ("pause");
}
int sum (int s, int n) {
int i;
s=0;
for (i=0; i<=n; i++) {
somma = i+ somma;
}
return s;
```

PROGETTAZIONE

Nascita UML

- Grady Booch
- Jim Rumbaugh
- Ivar Jacobson

«los amigos» danno origine ad UML raccogliendo le migliori prassi nel settore e definendo così uno standard

E' un linguaggio utilizzato per realizzare MODELLI basato sul paradigma della programmazione object oriented

linguaggio per specificare, visualizzare, e realizzare i prodotti di sistemi software, e anche per il business modeling. L'UML rappresenta una collezione di "best engineering practices" che si sono dimostrate utili nella progettazione di sistemi complessi e di grandi dimensioni

E' indipendente dal linguaggio utilizzato per il coding

Modelling

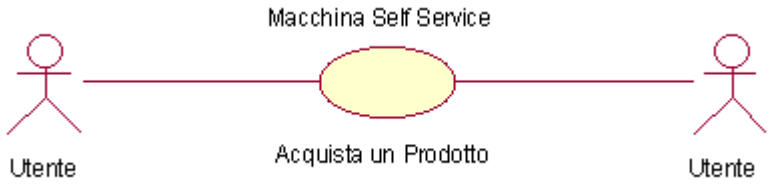
Il Sistema viene esaminato sotto 3 punti di vista, considereremo:

- aspetto funzionale
  use case diagram (comportamento visto dall' "utente")
- aspetto strutturale
  class diagram (oggetti, attribute, associazioni)
- aspetto dinamico
  sequence diagram (interazione tra attori e oggetti)
  activity diagram (workflow)
  statechart diagram (automi a stati finiti)

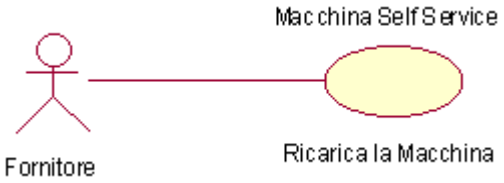
Principio fondamentale: **KISS** = Keep It Simple, Stupid

UML convenzioni

- I **diagrammi** sono grafi
  i nodi sono le entità
  gli **archi** sono relazioni

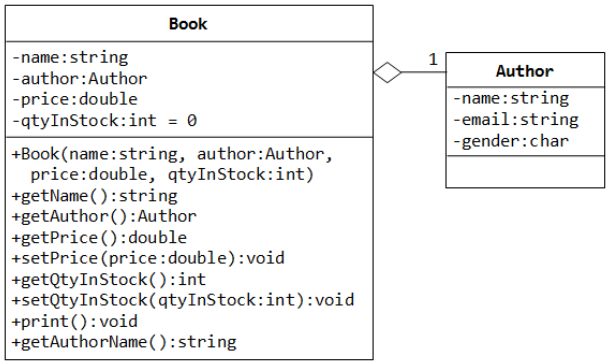


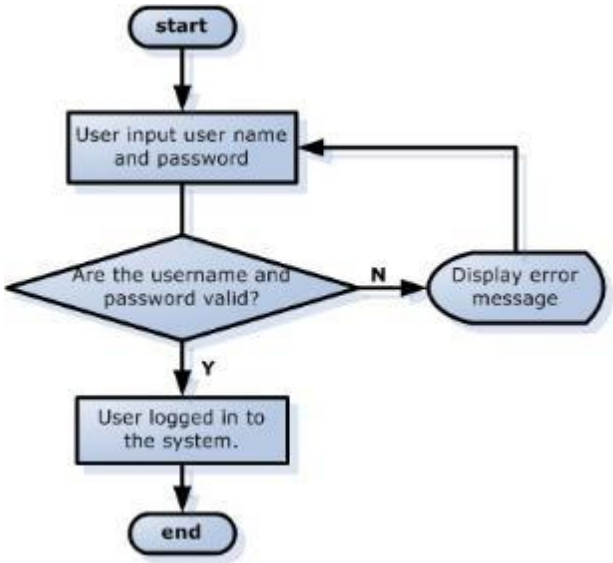
- i rettangoli sono classi o istanze
  le istanze sono hanno nome sottolineato
  i tipi non sono sottolineati



- gli ovali sono funzioni o case

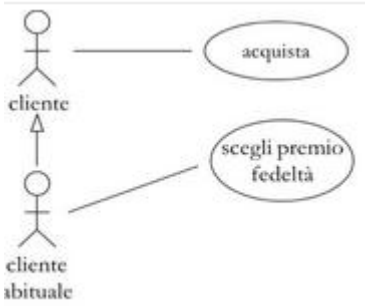
- gli omini stilizzati rappresentano gli actor

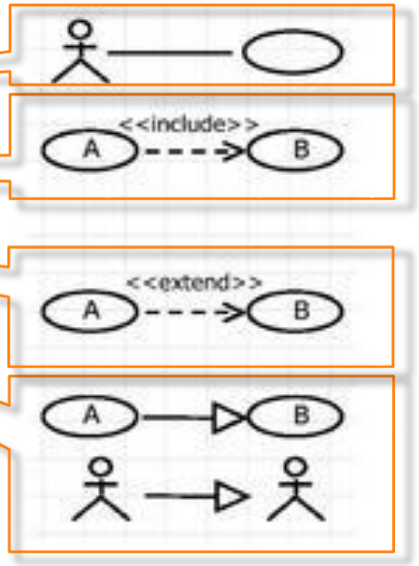




USE CASE DIAGRAM

- actor (esterni al sistema) – chi utilizza il sistema
- use case (interni al sistema) – le modalità di utilizzo
- relation ("connettono" entità) – le relazioni
  - associazione = "collegamento"
  - inclusion = A "include" B
  - estensione = A è variante di B
  - generalizzazione = A "eredita" da B

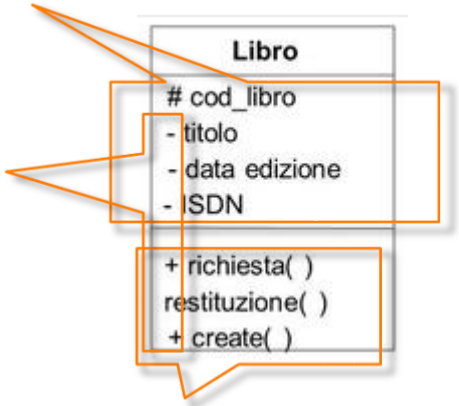




CLASS DIAGRAM

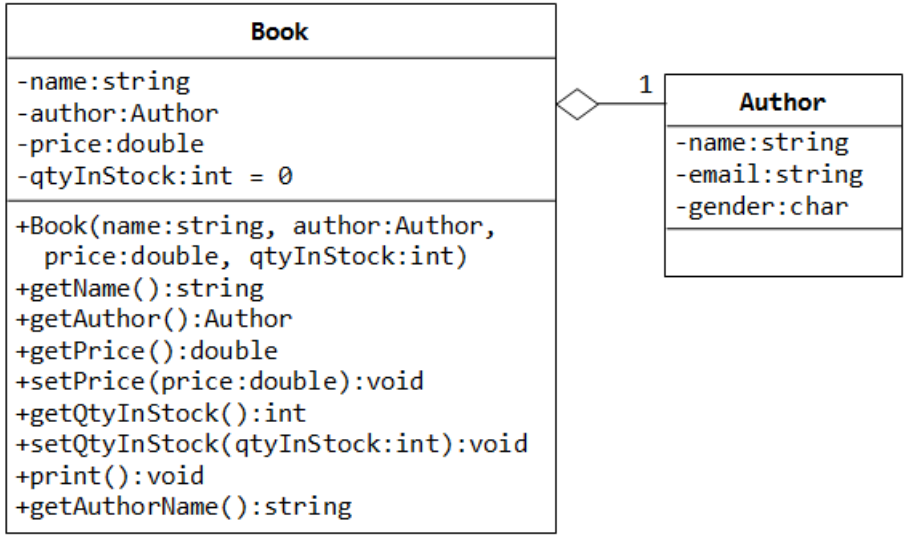
+ classi

attributi / proprietà

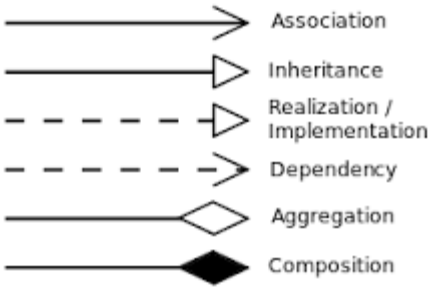


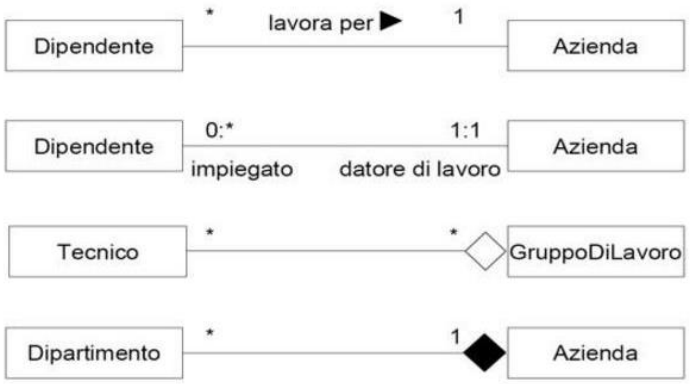
visibilità

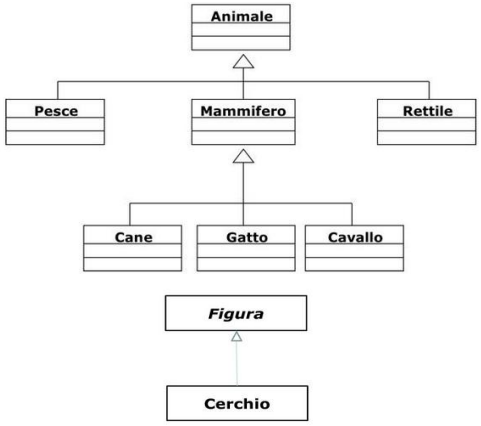
operazioni / metodi



+ relazioni



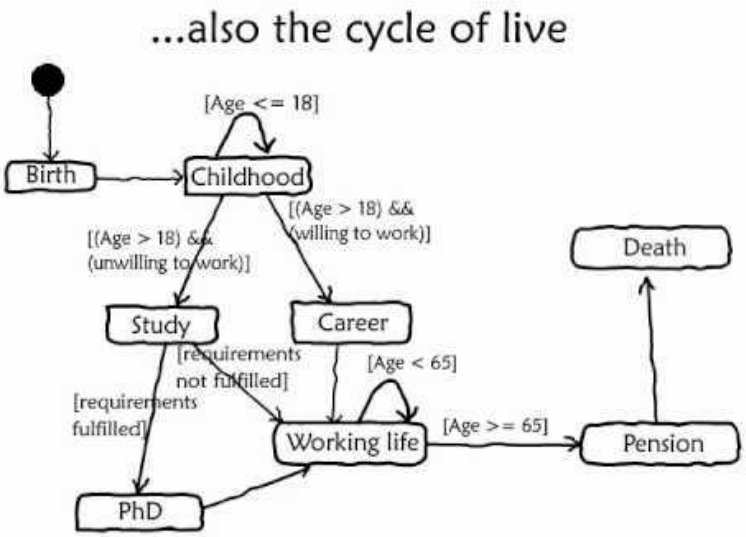


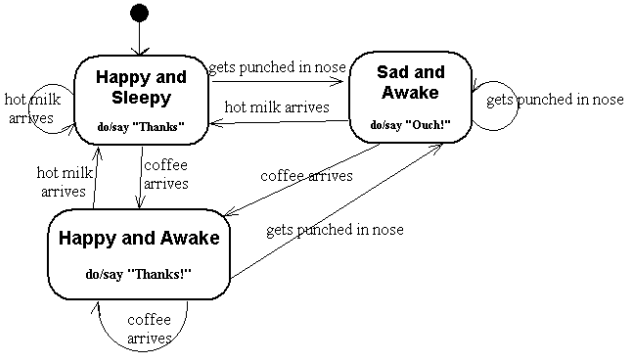


STATECHART diagrams

Descrivono il comportamento di USE CASES e CLASS

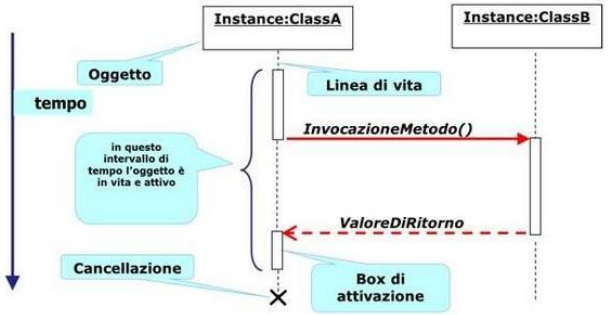
Un'entità il cui stato è descritto mediante valori dei parametri che la caratterizzano, subisce una transizione (cambia stato) in risposta ad un evento (**trigger**)

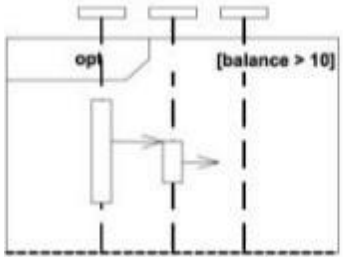


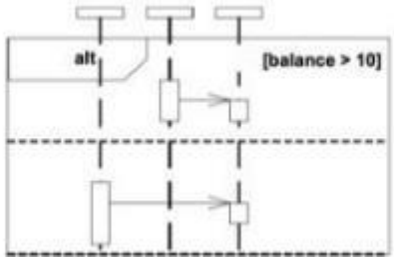


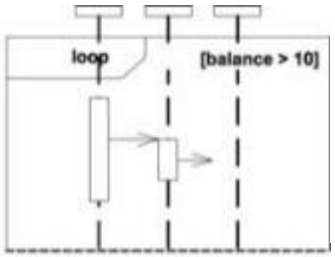
SEQUENCE diagram

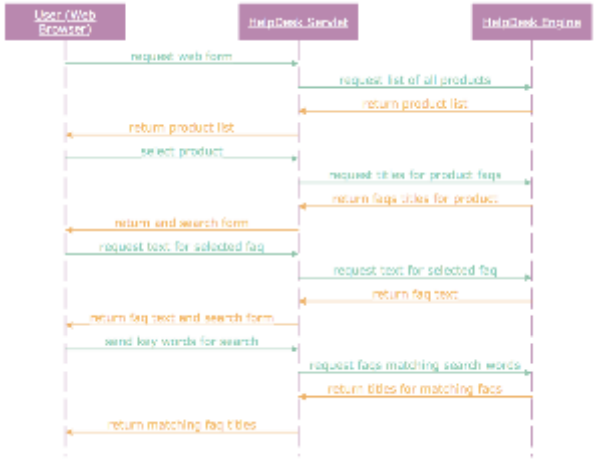
- messaggi
  sincrono – il mittente resta in attesa di un ritorno
  asincrono – il mittente procede con l'esecuzione
  ritorno – il destinatario restituisce un risultato



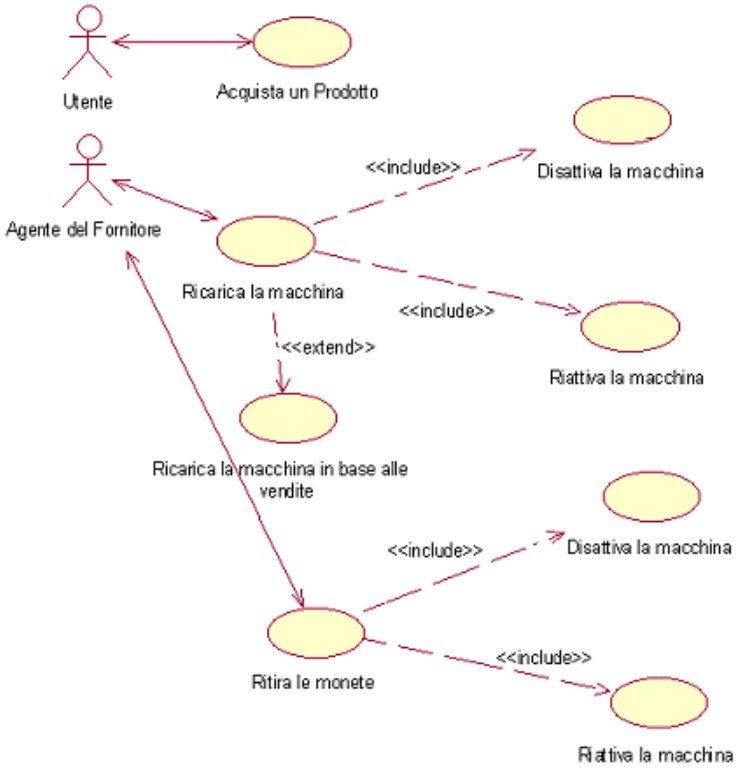


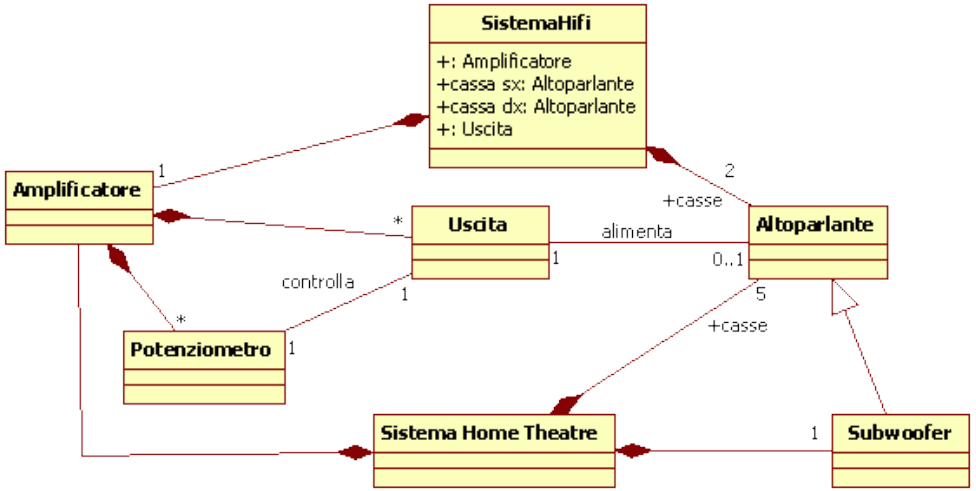


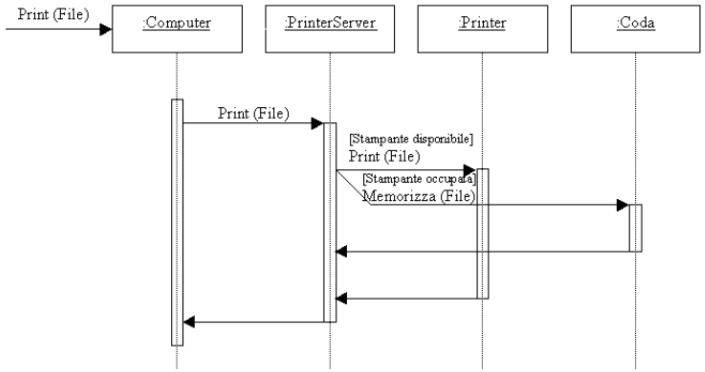




Esempi

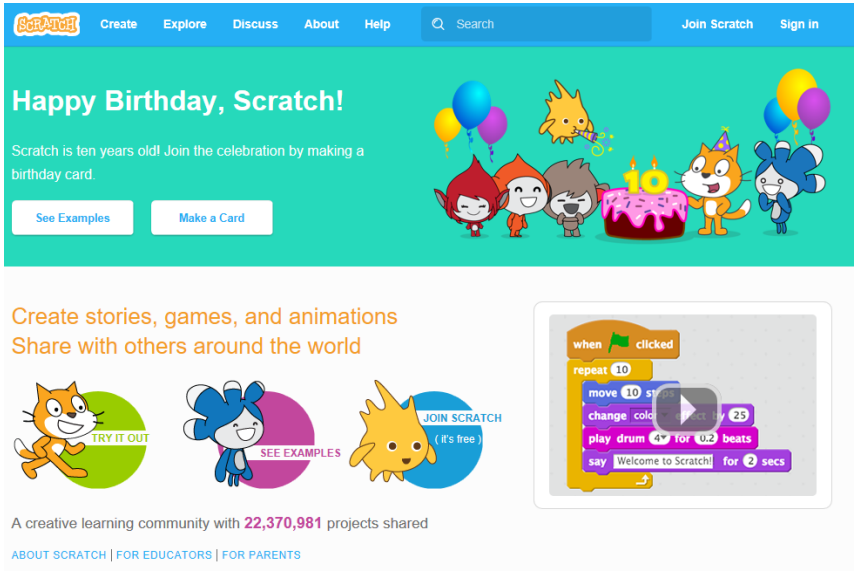

Utente
Acquista un Prodotto
Agente del Fornitore
<<include>>
Disattiva la macchina
Ricarica la macchina
<<include>>
Riattiva la macchina
<<extend>>
Ricarica la macchina in base alle vendite
<<include>>
Disattiva la macchina
Ritira le monete
<<include>>
Riattiva la macchina


SistemaHifi
+: Amplificatore
+cassa sx: Altoparlante
+cassa dx: Altoparlante
+: Uscita
Amplificatore
Uscita
Altoparlante
alimenta
+casse
controlla
Potenziometro
Sistema Home Theatre
Subwoofer


Print (File)
:Computer
:PrinterServer
:Printer
:Coda
Print (File)
[Stampante disponibile]
Print (File)
[Stampante occupata]
Memorizza (File)

CODING (with scratch) – hands on

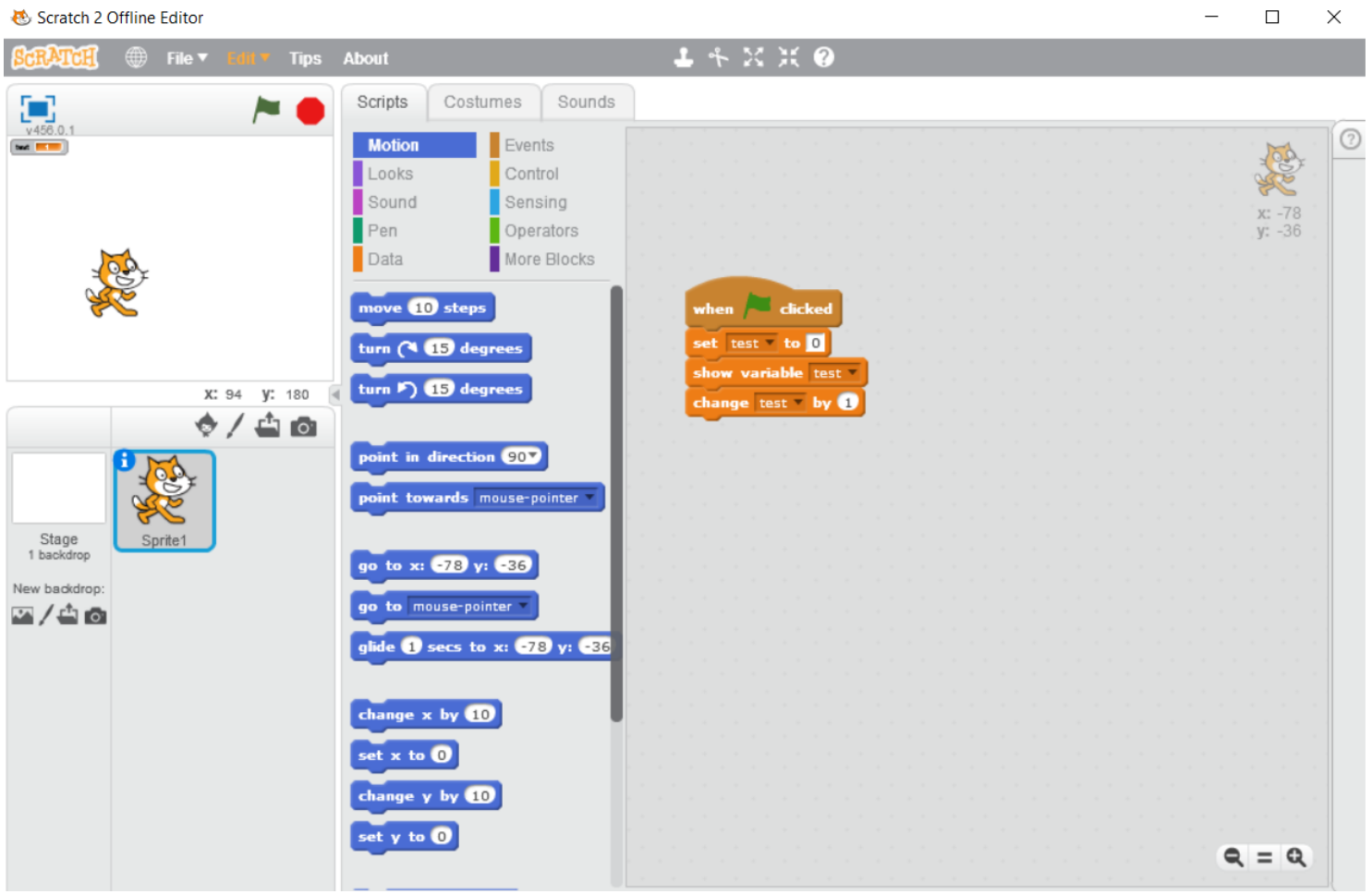
Scratch è un linguaggio di programmazione grafico realizzato al MIT che consente di realizzare **programmi multimediali** ed **animazioni interattive** (anche **lezioni**).
Scratch prevede un **approccio orientato agli oggetti** (denominati sprites)
Il linguaggio è **disponibile gratuitamente** su web e **multipiattaforma**
https://scratch.mit.edu/ - https://scratch.mit.edu/scratch2download/ **(offline editor)**



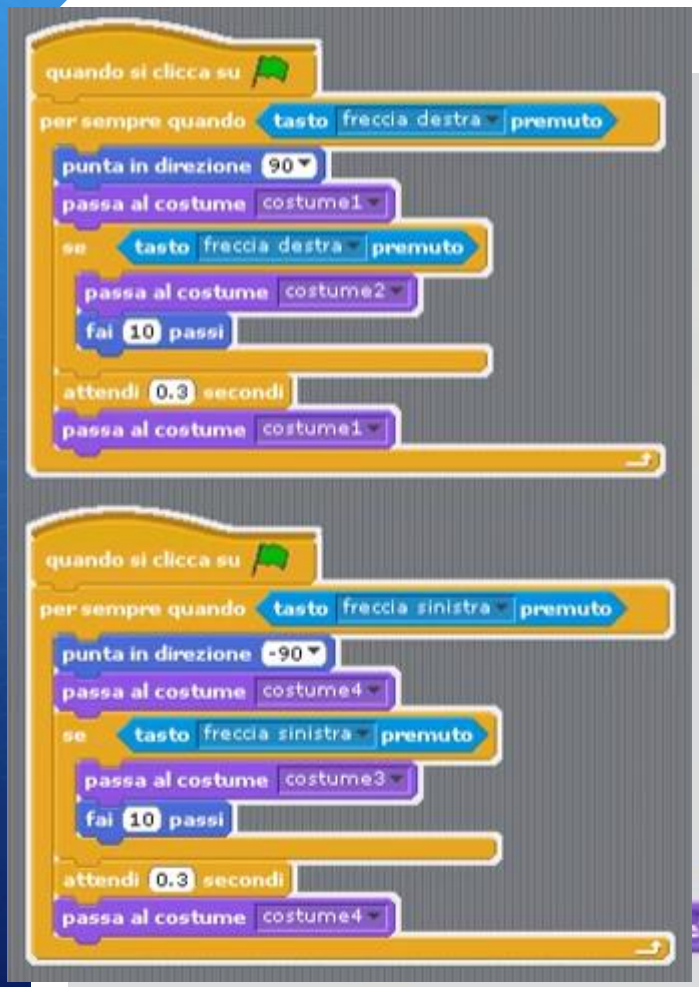
Categoria	Note
Movimento	Muove gli Sprite e cambia gli angoli
Aspetto	Controlla la visibilità, i costumi e l'output
Suono	Esegue brani audio e sequenze audio programmabili
Penna	Supporto al disegno e alla grafica
Variabili e liste	Uso di variabili e assegnazione di valori

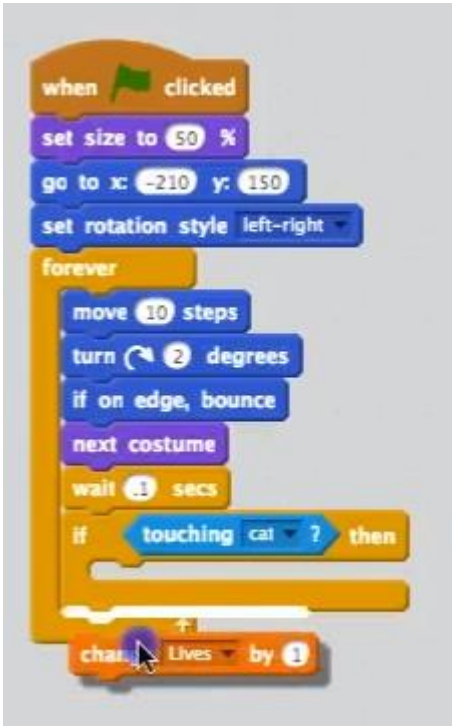
Categoria	Note
Situazioni	Blocchi di gestione degli eventi e da porre come testata.
Controllo	Istruzioni se e strutture ciclo/loop
Sensori	Sensori per gli Sprite e input utente
Operatori	Operatori matematici e booleani.
Altri blocchi	Procedure personalizzate (blocchi) e controllo di periferiche.

Scratch OFFLINE EDITOR

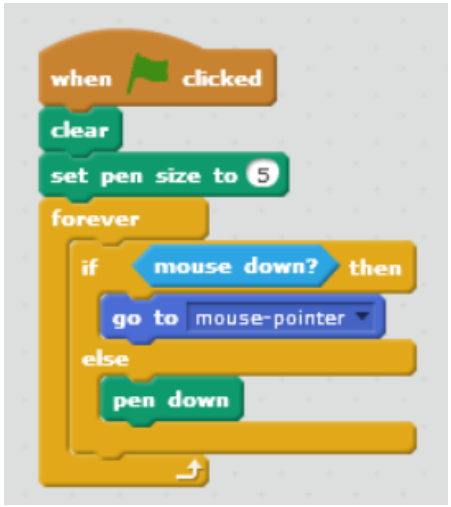


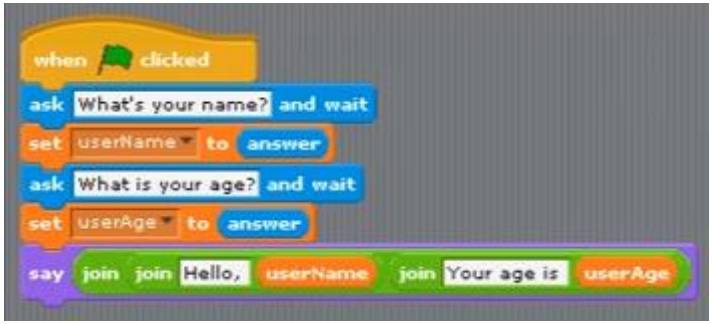
Scratch – movimento / collision detector





Scratch – IO / VARIABILI



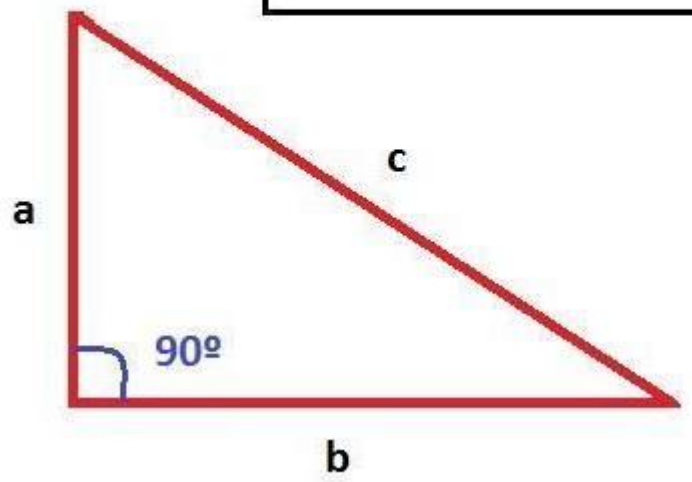


Alcune “sfide”

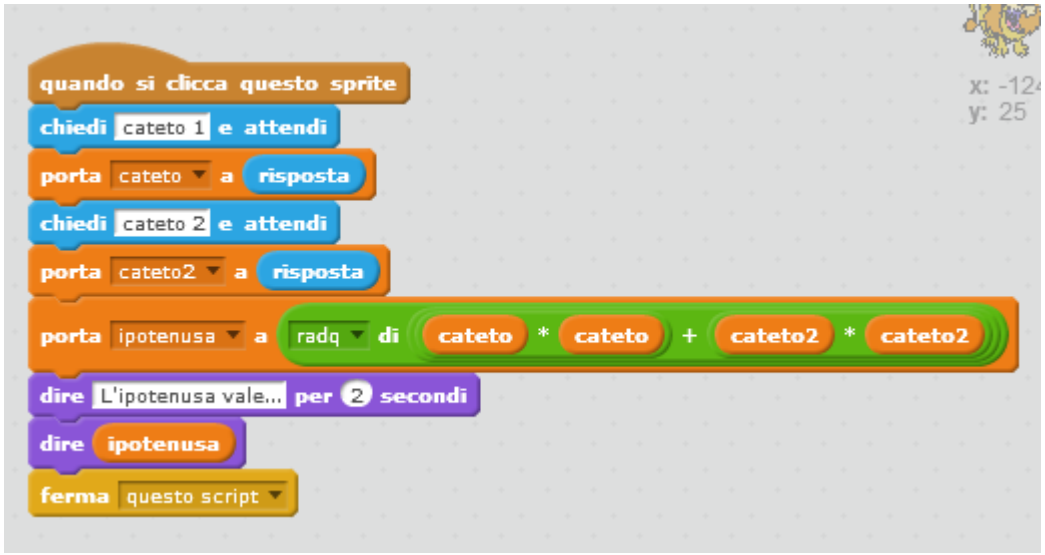
- Il teorema di Pitagora
- La successione di Fibonacci
- Ordinamento
  bubble sort
  insertion

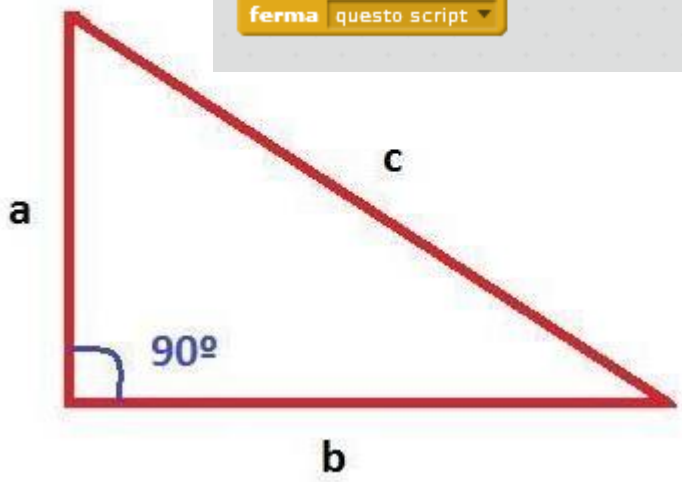
Scriviamo un programma che calcoli il valore dell'ipotenusa a partire dai cateti

$$c^2 = a^2 + b^2$$



Scriviamo un programma che calculi il valore dell'ipotenusa a partire dai cateti





La successione di Fibonacci

Come si evolve la popolazione di una colonia di conigli a partire da una coppia?

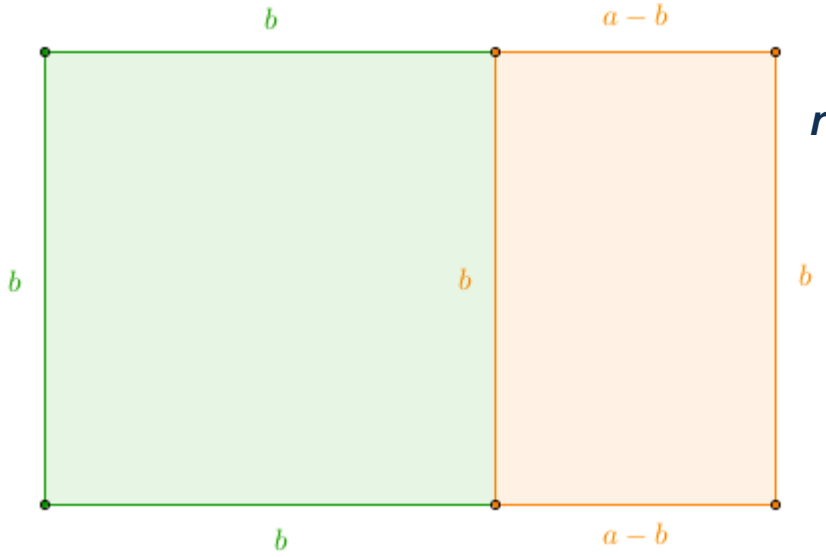
Le «regole» sono:

- ogni coppia di conigli genera solamente un'altra coppia ad ogni ciclo (1 mese)
- i conigli hanno bisogno di un mese di vita per diventare fertili
- una volta fertile ogni coppia continuerà a generare una coppia di conigli al mese

La successione di Fibonacci

1,1,2,3,5,8,13,21,…

$$\begin{cases} f_1 = 1; \\ f_2 = 1; \\ f_n = f_{n-1} + f_{n-2} \quad n \geq 3 \end{cases}$$



rettangolo aureo

$$\frac{a}{b} = \frac{b}{a-b} = \phi$$

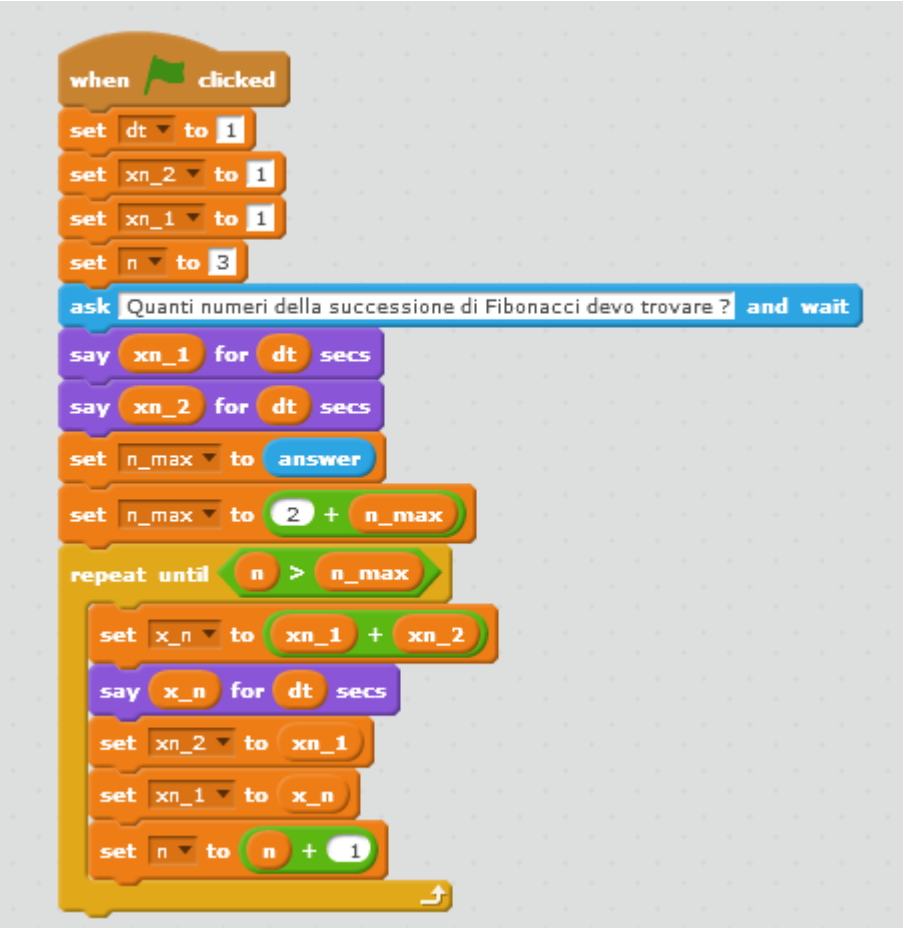
$$\frac{f_3}{f_2} = 2;$$
$$\frac{f_4}{f_3} = \frac{3}{2} = 1,5;$$
$$\frac{f_5}{f_4} = \frac{5}{3} = 1,\overline{6};$$
$$\frac{f_6}{f_5} = \frac{8}{5} = 1,6;$$
$$\frac{f_7}{f_6} = \frac{13}{8} = 1,625;$$
$$\frac{f_8}{f_7} = \frac{21}{13} = 1,\overline{615384};$$
$$\vdots$$

sezione aurea

ϕ=1,618033

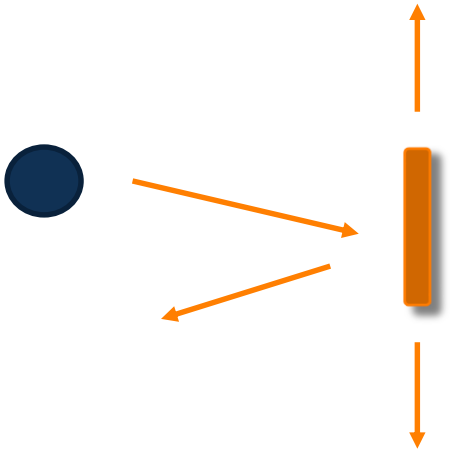
Fibonacci e scratch

Chiede quanti numeri calcolare e procede



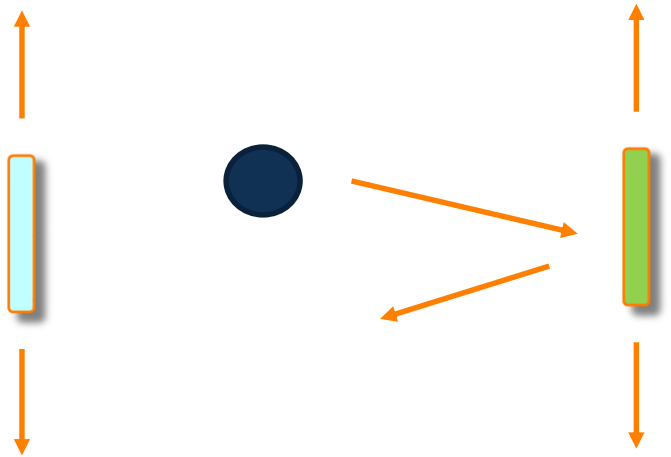
PONG 1

- Una pallina si muove di moto rettilineo uniforme
- Una "racchetta" si muove alla pressione di un tasto (su o giu)
- Quando la pallina urta la racchetta "rimbalza"
- Se "esce dallo schermo" si ferma il gioco …



PONG 2

- 1 pallina si muove di moto rettilineo uniforme
- 2"racchette" si muovono alla pressione di un tasto (su o giu)
- Quando la pallina urta una racchetta "rimbalza"
- Se "esce dallo schermo" si ferma il gioco …



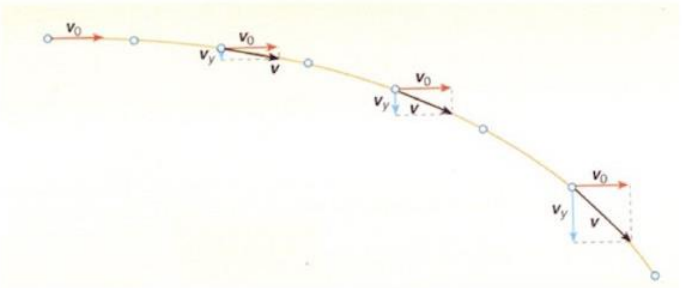
gravity

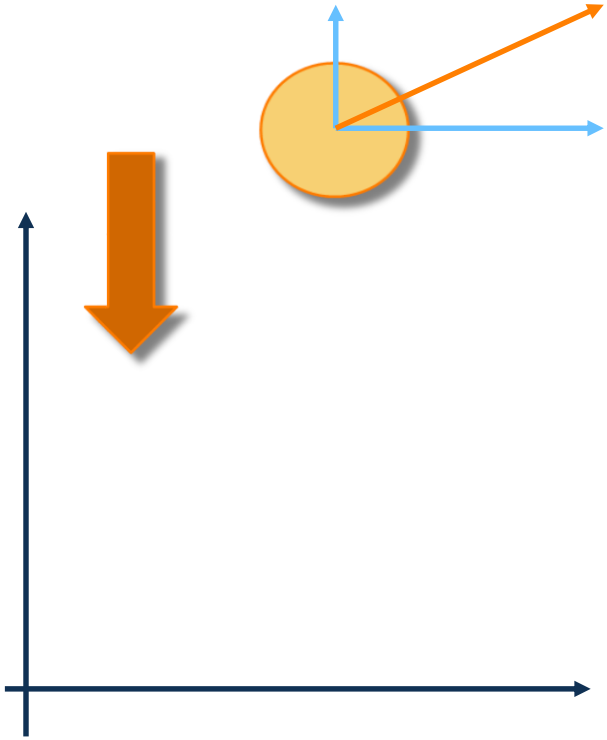
Caduta libera, balistica

$$\begin{cases} x = x_0 + v_{0x}t \\ y = -\frac{1}{2}gt^2 + v_{0y}t + y_0 \end{cases}$$

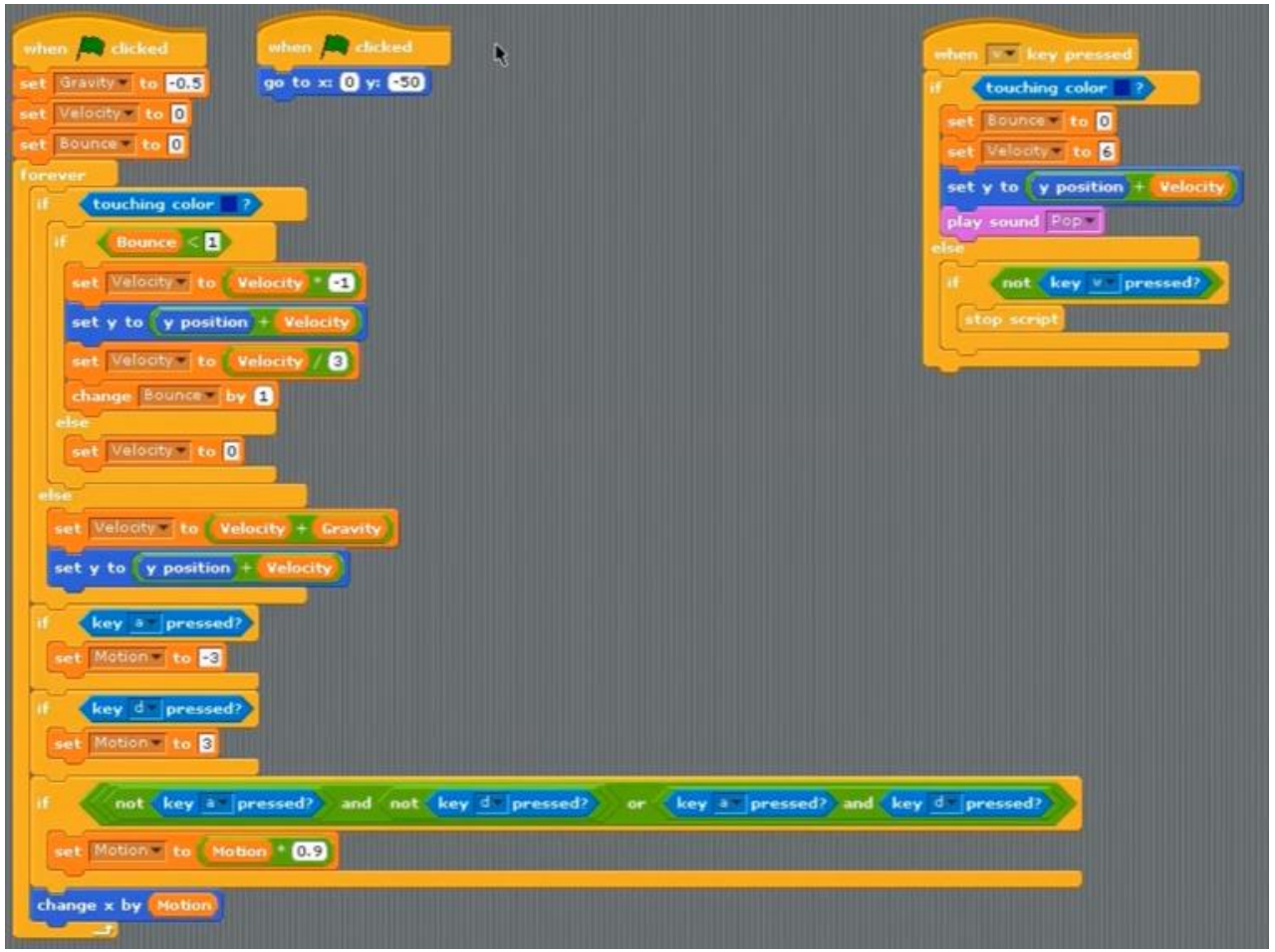
$$\begin{cases} v_{0x} = v_0 \cos(\alpha) \\ v_{0y} = v_0 \sin(\alpha) \end{cases}$$

$$V = \sqrt{V_{0x}^2 + V_y^2}$$





Scratch – gravity simulation



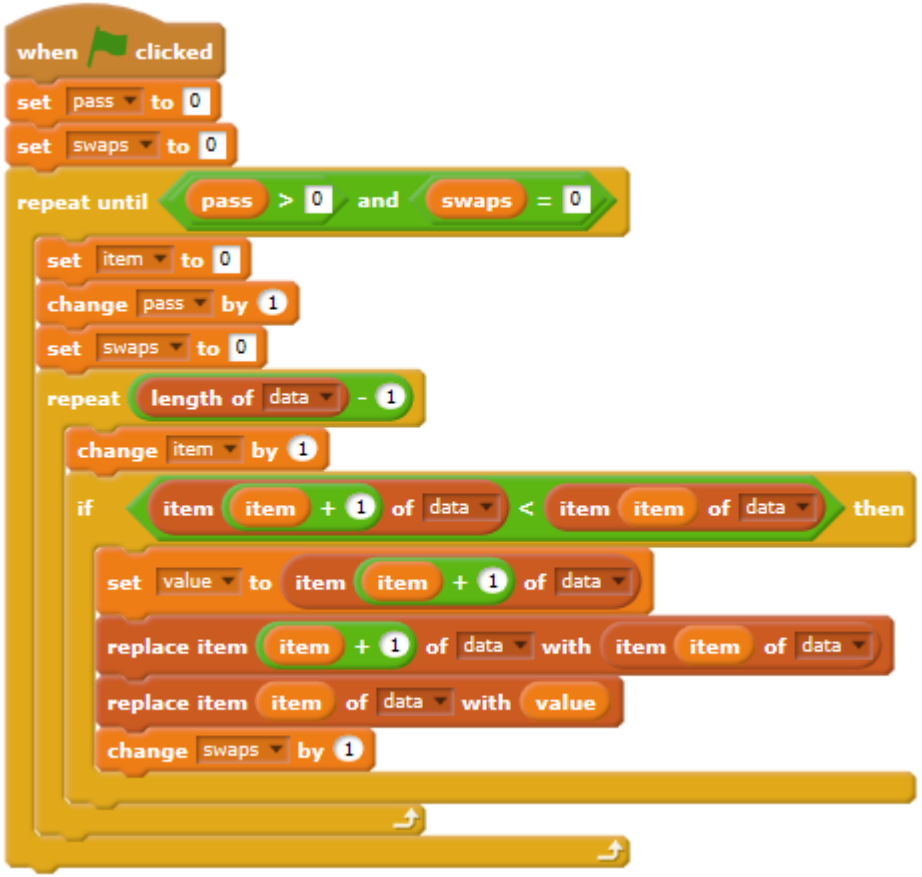
Ordinamento

Obbiettivo: ordinare una lista di numeri; “algoritmo”:

1. Porre la variable swap=1
2. se swap=1, porre swap=0 (altrimenti fine)
3. considerare una coppia di numeri “vicini”
4. se il secondo è minore del primo
   scambiare i numeri, porre swap=1
5. Procedere con la coppia successiva
6. Ricominciare da 2

Come ordinare una lista?
Bubble sort

Questo script riordina i dati nella lista “data”



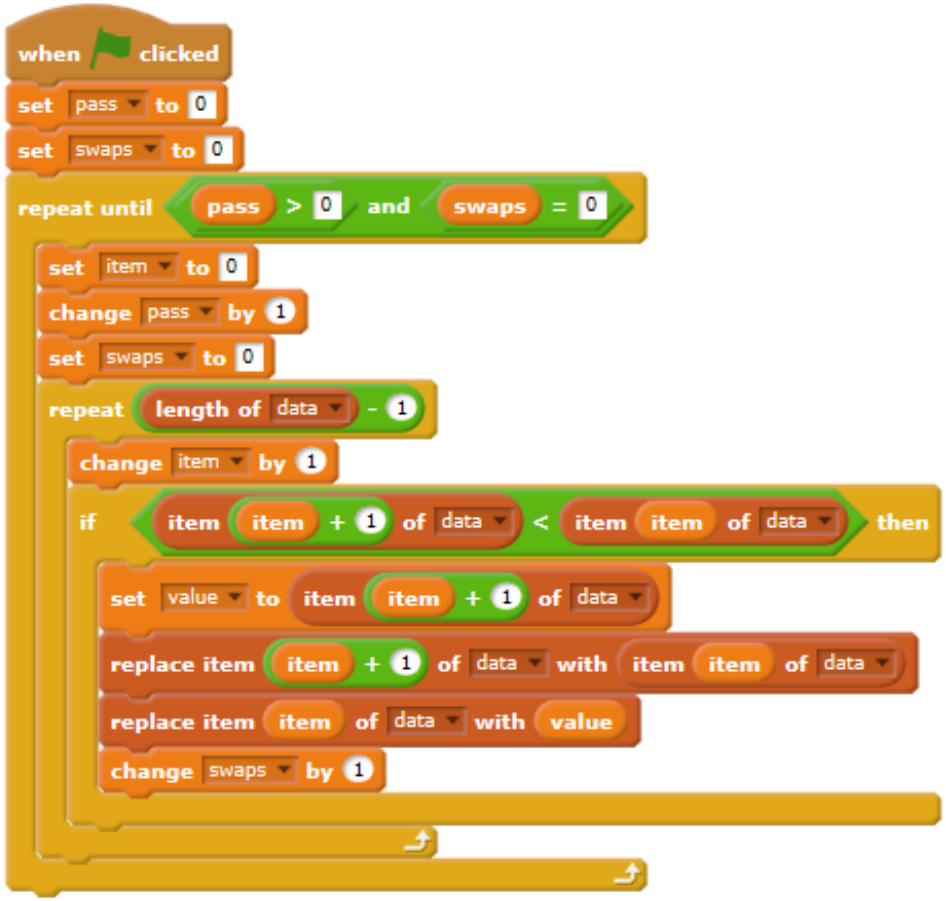
Come ordinare una lista? Insertion sort

Questo script riordina i dati nella lista “text”

La lista è divisa in 2 parti, uno ordinate una non ordinata

Un valore estratto dalla parte non ordinate viene inserito al proprio posto nella parte ordinata

Il processo continua fino a quando tutti I valori sono inseriti nella parte di lista ordinata



La fisica di PONG1 – in assenza di urti

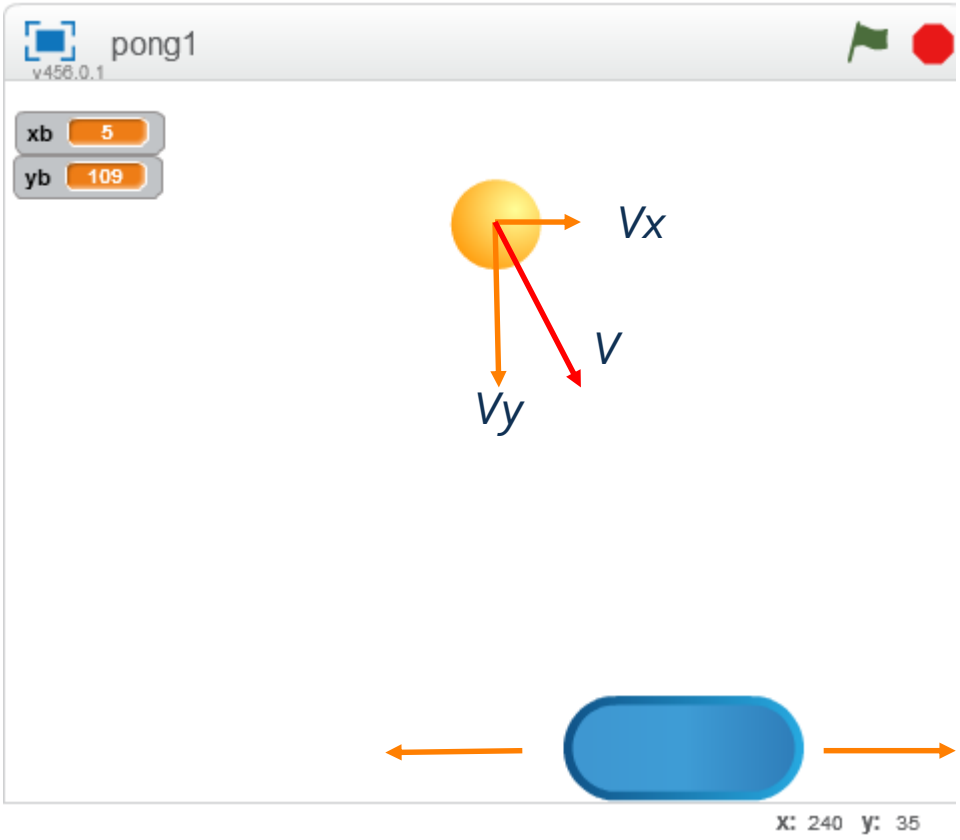
Problema:

Data la velocità lungo x ed y e la posizione x ed y, trovare la posizione successiva

$$\begin{cases} x_1 = x_0 + v_x * \Delta t \\ y_1 = y_0 + v_y * \Delta t \end{cases}$$

$$\begin{cases} x_2 = x_1 + v_x * \Delta t \\ y_2 = y_1 + v_y * \Delta t \end{cases}$$

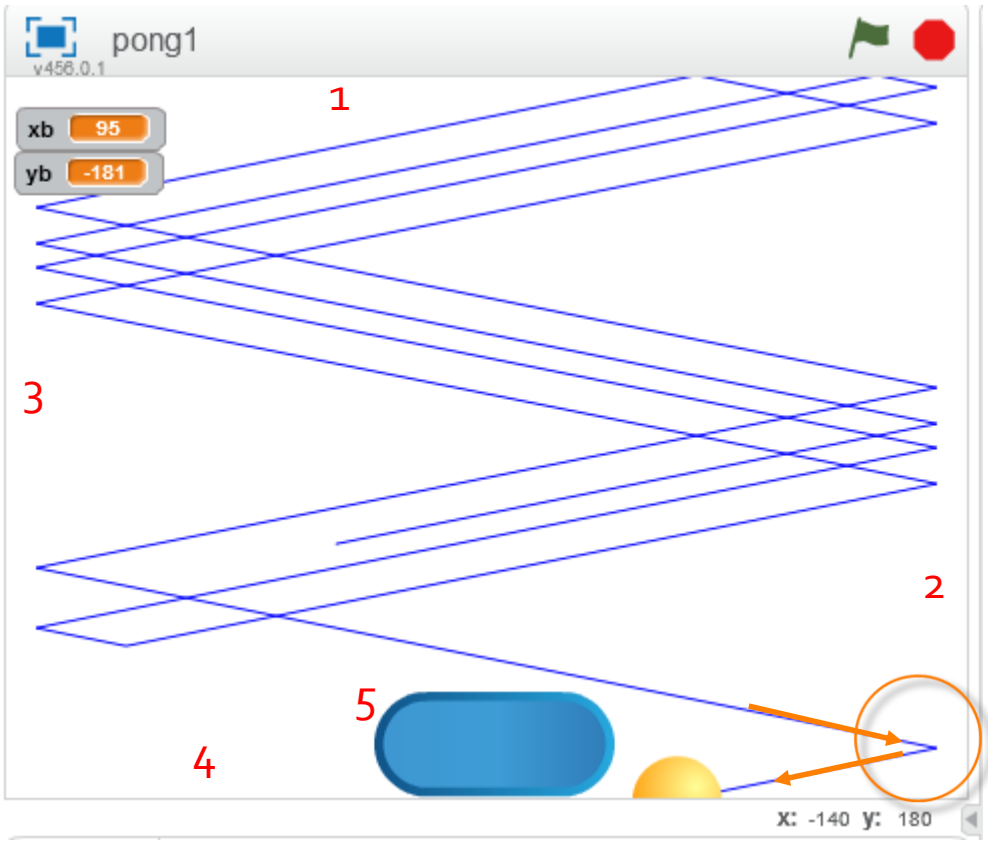
$$\begin{cases} x_{t+1} = x_t + v_x * \Delta t \\ y_{t+1} = y_t + v_y * \Delta t \end{cases}$$



La fisica di PONG1 – urti «elastici»

URTI ELASTICI
Durante un urto la velocità della pallina cambia direzione, non cambia il modulo di *v*

Per gestire correttamente gli urti dobbiamo decidere come «reagire» alle 5 condizioni che possono verificarsi:
-> 4 urti con le pareti (1,2,3,4) e
-> uno con la pad (5)

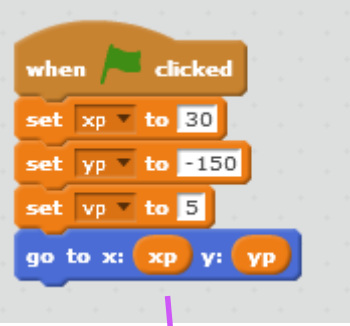


+ 1,5 la componente lungo y della velocità cambia segno
+ 2,3 la componente lungo x della velocità cambia segno
+ 4 fine della «partita»

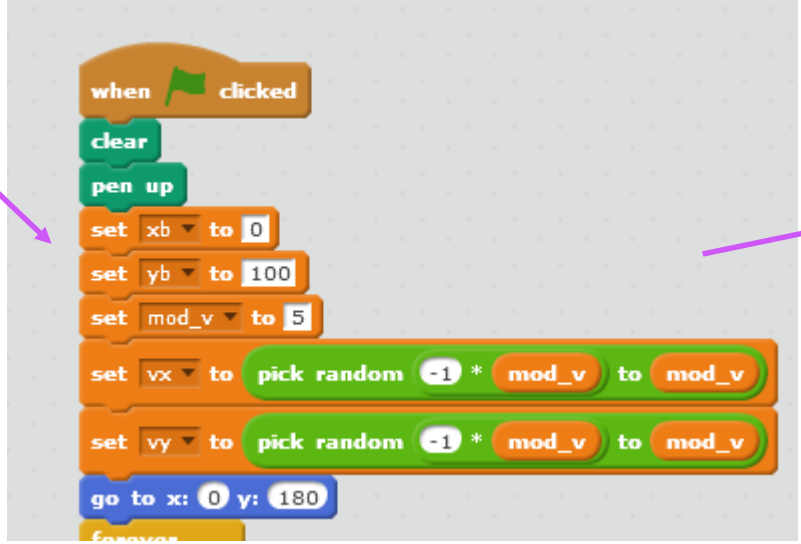
Inizializzazione: le condizioni iniziali di pallina e pad

+ PALLINA

+ PAD



Velocità nulla, posizione di partenza predefinita



Velocità iniziale casuale, posizione di partenza predefinita

… cosa accade se *Vy=o* ?

Il PAD è controllato dall'operatore che ne cambia la velocità quindi la posizione

Le equazioni del moto

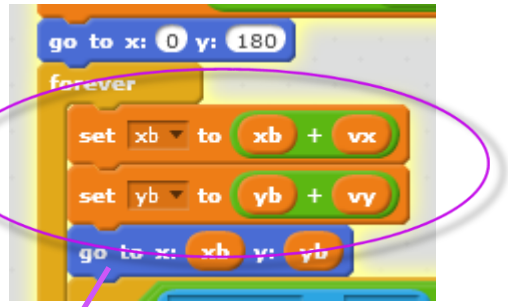
- Metodo di Eulero
  per risolvere «eq.differenziali»

$$\begin{cases} x_{t+1} = x_t + v_x * \Delta t \\ y_{t+1} = y_t + v_y * \Delta t \end{cases}$$

$\Delta t = 1$

$$\begin{cases} x_{t+1} = x_t + v_x \\ y_{t+1} = y_t + v_y \end{cases}$$

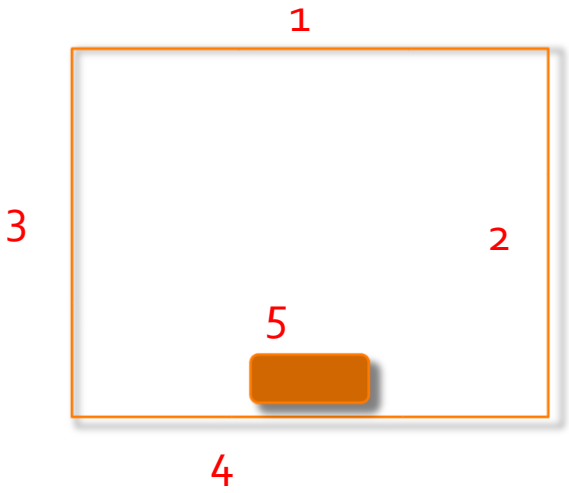
CODING !!!



Una volta calcolata la nuova posizione è sufficiente spostare la pallina

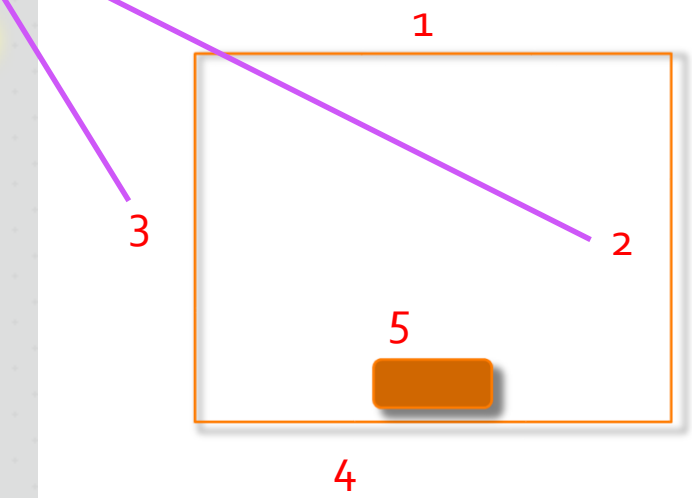
Come «gestire» gli urti ?

La trigonometria può complicare le cose… !



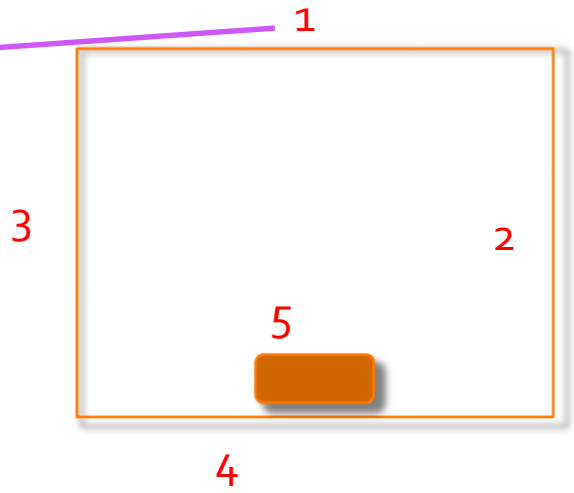
Come «gestire» gli urti ?

La trigonometria può complicare le cose… !



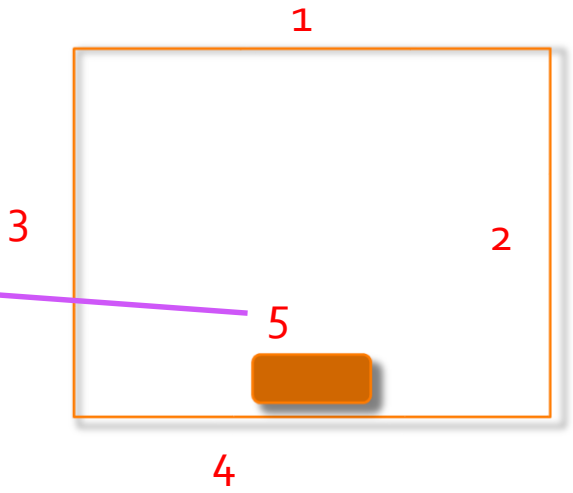
Come «gestire» gli urti ?

La trigonometria può complicare le cose… !



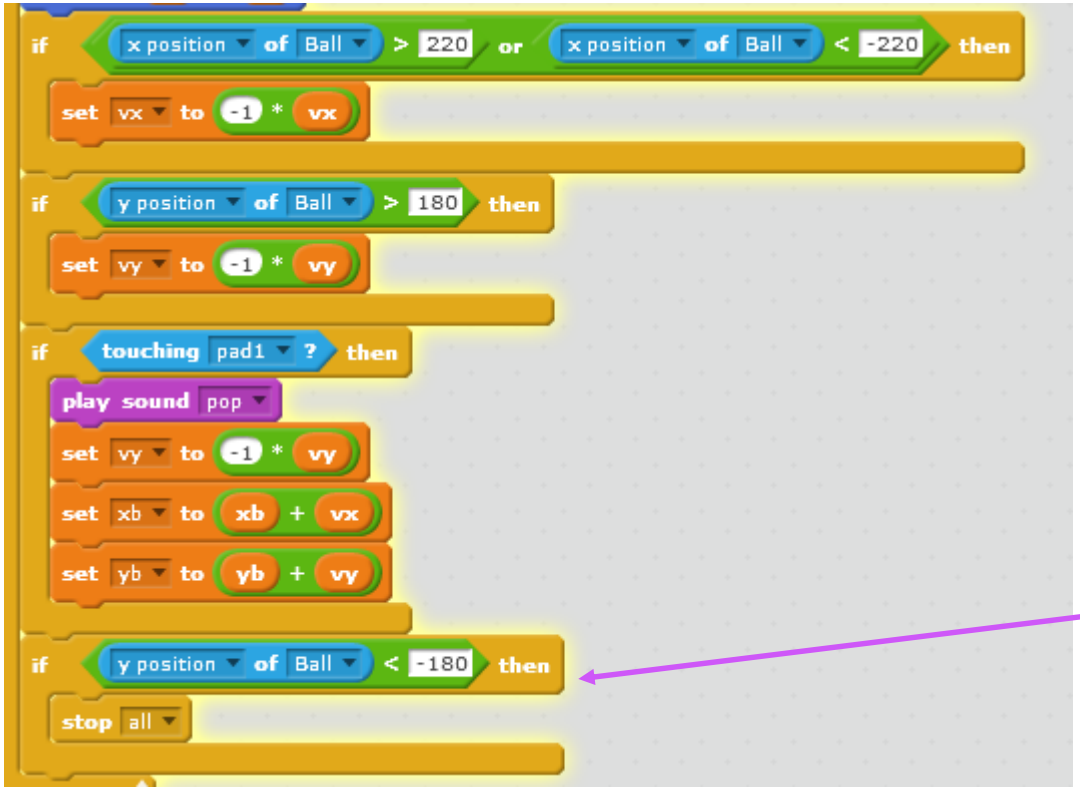
Come «gestire» gli urti ?

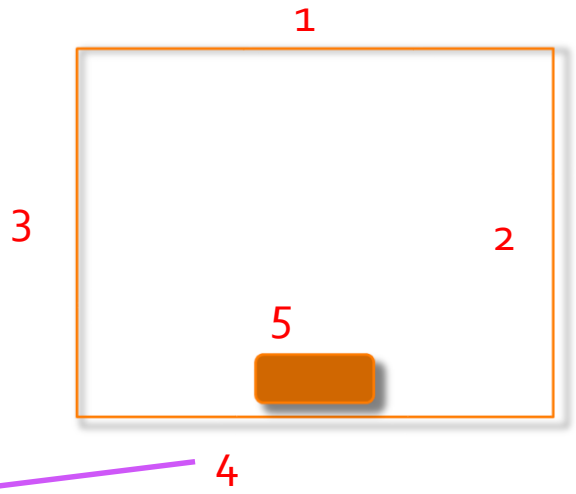
La trigonometria può complicare le cose… !



Come «gestire» gli urti ?

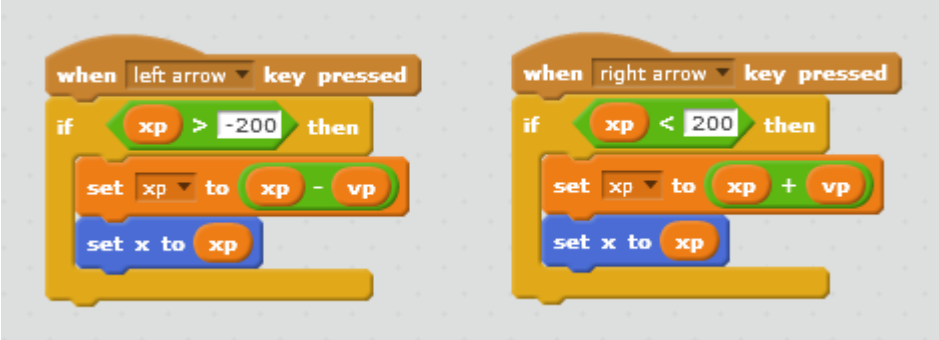
La trigonometria può complicare le cose… !





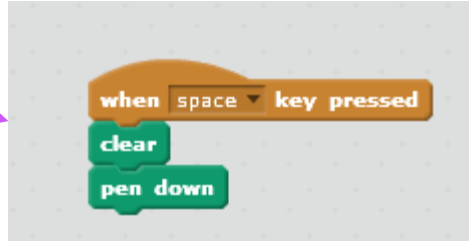
Interazione con l'utente
Il controllo del PAD, la track della pallina

Il PAD non si muove di moto rettilineo uniforme, è «controllato»



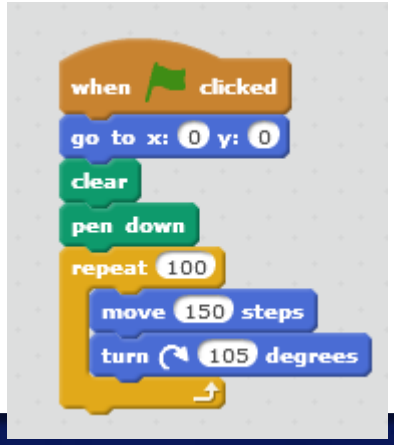
utilizziamo le equazioni precedenti,
le applichiamo solo in risposta ad un evento (pressione tasto)

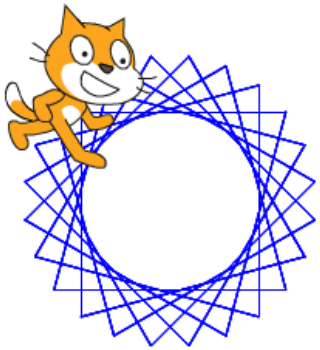
Possiamo realizzare un «esperimento» facendo in modo che la pallina lasci una traccia durante il suo percorso, per vedere come si comporta quando urta le pareti



Scratch 2 e turtle graphics

Il linguaggio LOGO fu ideato e realizzato negli anni '6o dal professor Seymour Papert del MIT per muovere un semplice robot, al quale si potevano dare comandi del tipo avanti, gira a destra etc.
Nelle successive implementazioni si ha la metafora di una tartaruga (rappresentata da un triangolo) in grado di lasciare una traccia sullo schermo

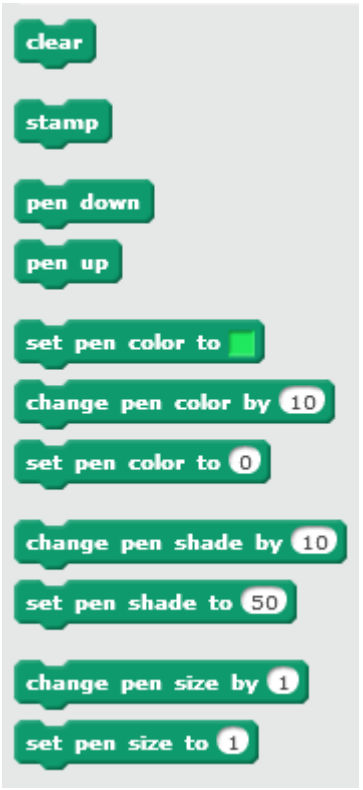




COMANDI PRINCIPALI TURLE GRAPHICS

Andare avanti
Andare indietro
Ruotare a destra
Ruotare a sinistra
Cancella il disegno
Torna al punto iniziale
Penna su
Penna giù
Colore penna
Colore riempimento ROSSO
Colore schermo
Dimensione penna

SCRATCH2 PEN



Conclusioni

scratch 2 / junior (il coding) come strumento didattico presenta

Vantaggi:

+ comprensione del contesto (analisi, modellazione) utilizzare UML per realizzare modelli
+ Comprensione delle variabili in gioco / rilevanti,
  impiego di un linguaggio formale (matematica, codice) «più preciso» del linguaggio naturale
+ Modellizzazione – realizzare un modello di un problema
  => simulazione di situazioni «impreviste» / difficili o impossibili da trattare analiticamente
+ Competenze «spendibili»

Svantaggi

+ .. altro lavoro per il docente, ma si può ridurre **condividendo**
+ rischio di «perdersi per strada» (dettagli di implementazione etc) ESPERIENZA!