Informatica II (Laboratorio)

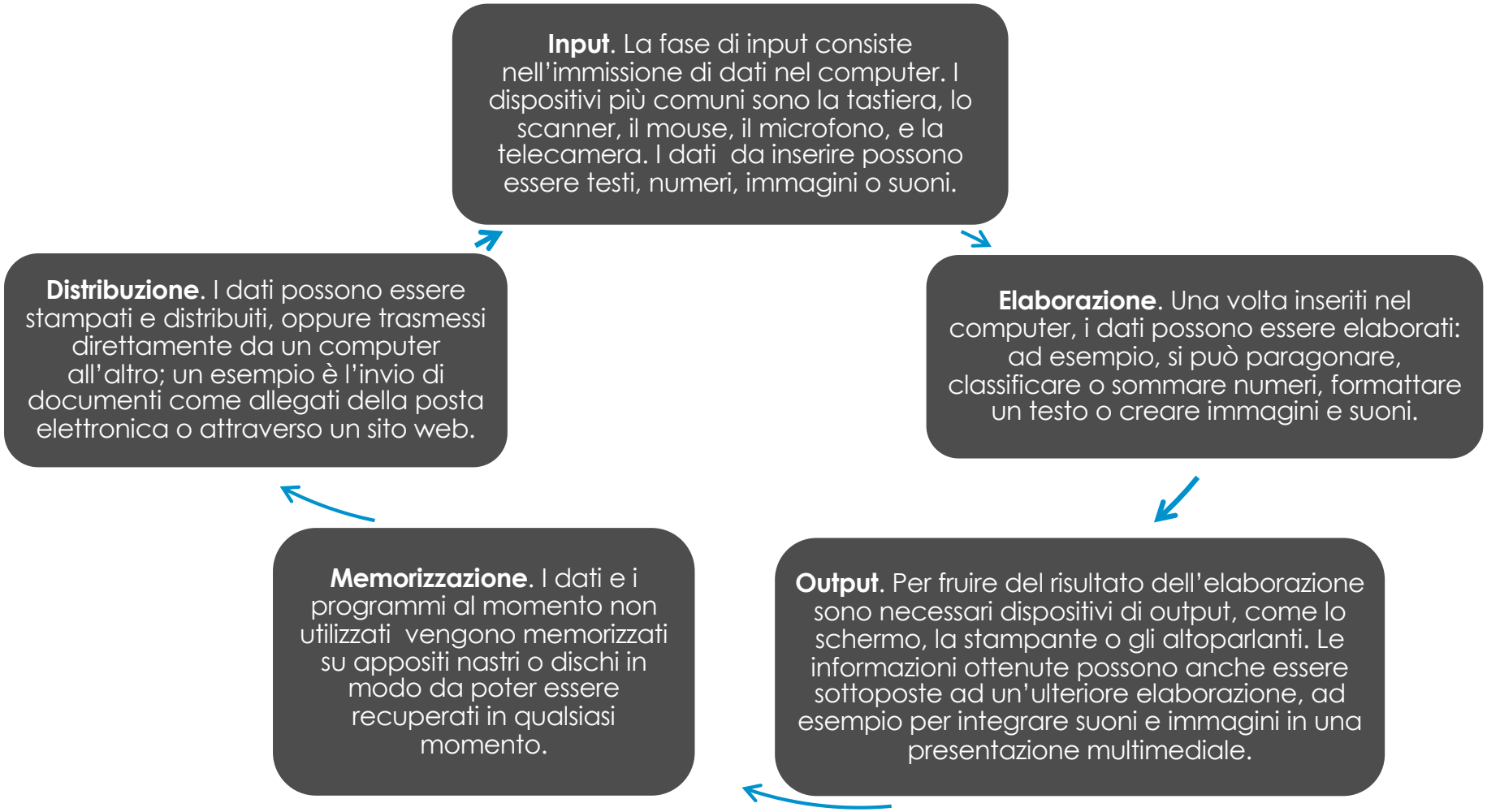
Modulo Algoritmi e Scratch

Andrea Bracciali

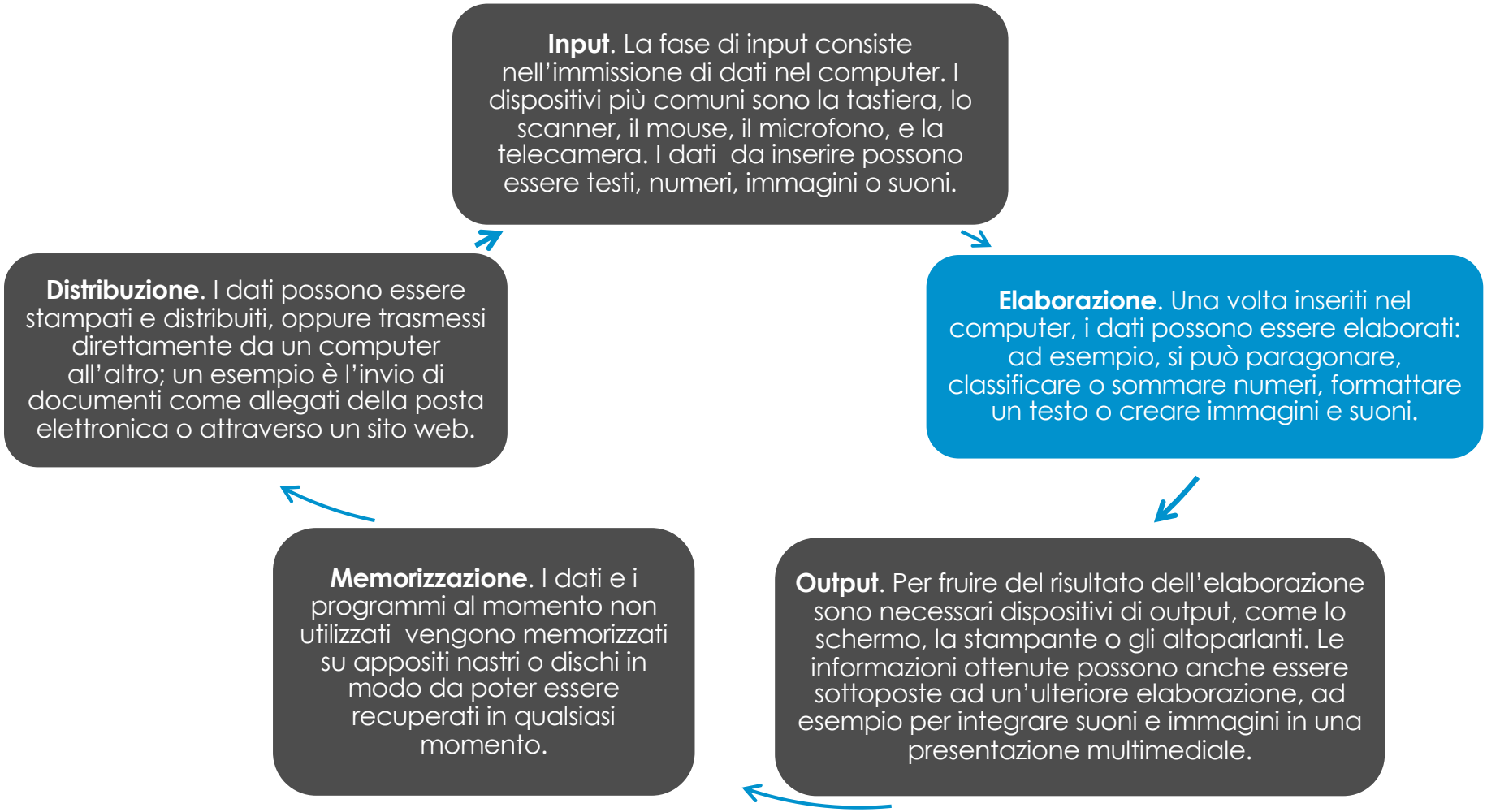
Argomenti

- L'elaborazione dell'informazione
  - Problema, Algoritmo, Programma
  - Scratch
    - Semplici algoritmi
    - Variabili, decisioni, cicli
    - Strutture dati: le liste

Ciclo di elaborazione dell'informazione (esempio)



Ciclo di elaborazione dell'informazione (esempio!)



Elaborazione dell'informazione - 1

- Cosa vuol dire elaborare dell'informazione?
  - Si tratta di un processo in cui un esecutore esegue un particolare insieme di azioni su un insieme di informazioni obbedendo ad una procedura definita, allo scopo di risolvere un problema.
- Perché elaborare l'informazione?
  - Per risolvere un problema!

Elaborazione dell'informazione - 2

- Cos'è un problema?
  - Un problema specifica in termini generali una relazione che intercorrere tra dei dati di ingresso (input) e dei dati di uscita (output).
    - Ad esempio:
      - Dato un insieme di numeri (input), trovare il massimo (output).
      - Dato un insieme di nomi (input), ordinarli alfabeticamente (output).

Elaborazione dell'informazione - 3

- Come si risolve un problema? Attraverso un algoritmo:
  - una procedura per la soluzione di un problema,
  - tramite una sequenza ordinata
  - di operazioni semplici e ben definite
  - eseguibili da un esecutore (calcolatore, per es)
  - in tempo finito – la singola operazione
  - che può non terminare, e.g. 10 div 3 = 3.3333
  - Ad esempio:
    - Un ricetta, se ben definita, altrimenti NO - e.g. non può contenere, «q.b.», «pizzico di sale»...
    - Le istruzioni per assemblare un mobile.

Elaborazione dell'informazione - 4

- Le operazioni semplici (e i corrispettivi costrutti nei linguaggi di programmazione) si possono tipicamente dividere in
  - Espressioni, e.g. «restituisci un valore di un dato tipo»
    - un valore, una somma, una condizione, 3+3+5, x+3, z<5. ....
  - Comandi, e.g. «fai qualcosa»:
    - stampa un messaggio, x=x+3 (assegnamento) ...
  - Istruzioni di controllo: e.g. «reagisci a una condizione»:
    - se cotto spegni il fuoco, ripeti queste istruzioni 3 volte, if (z<5) then alarm ...

Elaborazione dell'informazione - 5

- Tipicamente, un algoritmo è una procedura computazionale che prende un valore in ingresso (input) e restituisce un valore in uscita (output).

› L'esecuzione di un algoritmo riguarda:

- La specifica dei dati da elaborare;
- La sequenza di azioni da compiere;
- La specifica dei controlli che determinano l'ordine in cui eseguire le azioni.

Elaborazione dell'informazione - 6

- Un buon algoritmo garantisce le seguenti proprietà:
  - Correttezza: l'algoritmo perviene alla soluzione del problema dato.
  - Finitezza (desiderabile, ma...): Il numero di azioni che fanno parte di un algoritmo è finito e l'algoritmo stesso perviene alla soluzione in un tempo finito.
  - Efficienza: l'algoritmo perviene alla soluzione utilizzando la minore quantità di risorse (tempo, spazio) possibile.
  - Generalità (desiderabile): L'algoritmo si occupa della risoluzione di una famiglia di problemi, quindi, compatibilmente con alcuni vincoli, non dipende dallo specifico input.

Elaborazione dell'informazione - 7

- Come si può far eseguire un algoritmo ad un computer?
  - Attraverso un programma

    programma <> algoritmo !

  - Un programma è la trascrizione (implementazione) di un algoritmo in un linguaggio formale di programmazione comprensibile da un computer (elaboratore/esecutore).
  - Un programma è quindi un insieme di frasi (istruzioni) che specificano le azioni da compiere in un linguaggio di programmazione, in accordo alla sintassi e alla semantica di tale linguaggio.

Elaborazione dell'informazione - 8

- Nota bene: l'algoritmo rimane lo stesso indipendentemente da:
  - Lo specifico linguaggio di programmazione utilizzato per scrivere il programma (es Java, PHP, C++)
  - Le caratteristiche del computer che lo esegue (es. computer con sistema operativo Mac OS o Windows)
  - I dati elaborati (es., per un algoritmo che trova il massimo, sequenza di numeri "1, 2, 3" o "3, 2, 1").

Introduzione alla programmazione con Scratch

Il progetto Scratch - 1

- Scratch è un progetto del Lifelong Kindergarten Group@MIT, Boston (http://scratch.mit.edu//)
- Si tratta di un ambiente gratuito che incoraggia la programmazione creativa.
- Progettato per i bambini di 8-16 anni, è usato da persone di tutte le età con obiettivi e in situazione diverse (biblioteche, scuole,...).
- Grazie a Scratch si possono creare giochi interattivi, musica, progetti multimediali, simulazioni… che si possono condividere.

Il progetto Scratch - 2

- Il nome deriva dalla *scratching technique* usata dai disk jockey: mixare, comporre in maniera creativa.
- Si impara a programmare mentre si creano storie interattive, giochi, progetti multimediali.
- Non serve saper programmare:
  - *starting from scratch* → partire da zero
  - La programmazione è completamente grafica: non occorre sapere nulla di programmazione formale.

L'ambiente Scratch - 1

- 3S – Sprite(s), Stage and Script
  - Sprite (agenti) – figure bidimensionali che svolgono le azioni, spesso in risposta a vari eventi. Il gatto si chiama Scratchy.
  - Stage – la zona dove vediamo quello che abbiamo creato – è la casa di Scratchy e degli altri sprites.
  - Script – insieme di comandi o blocchi.

L'ambiente Scratch - 1

Collegarsi a

https://scratch.mit.edu/

https://scratch.mit.edu/projects/editor/?tutorial=getStarted

(raggiungibile da Create – in alto nella pagina)

Eventualmente si può creare un account per la persistenza dei propri dati/programmi.

L'ambiente Scratch - 2

- I comandi di base si chiamano "blocchi".
  - Assomigliano ai mattoncini LEGO – sono colorati e si possono fare varie costruzioni mettendoli insieme.
- Sequenze di più blocchi messi insieme si chiamano "script".
  - Gli script si usano per controllare gli sprite.

L'ambiente Scratch - 3

- Pagina di aiuto
  - https://scratch.mit.edu/ideas
- FAQ
  - https://scratch.mit.edu/info/faq
- Per iniziare
  - https://scratch.mit.edu/projects/editor/?tutorial=getStarted

Blocchi di base

- Grazie ai blocchi, è possibile svolgere varie azioni:
  - Movimento (fai 10 passi, ruota di 45 gradi,…)
  - Aspetto (dire “Hello”, pensa “Hmmm”,…)
  - Suono (suonare il tamburo per 0.25 battute,…)
  - Eventi (quando si clicca questo sprite,…)
  - Controllo (ripeti 10 volte, per sempre,…)
  - Sensori (sta toccando il colore blu, x del mouse,…)
  - Operatori (operatori aritmetici (+,-,…), logici (<,>,…), di testo)
  - Variabili e liste (crea una variabile, crea una lista)
  - Altri blocchi

Un semplice algoritmo - 1

- Gli algoritmi più semplici sono composti da una sequenza di azioni, ad esempio:
  - Facciamo muovere Scratchy 10 passi a destra
  - Facciamo suonare il tamburo 1 per 0,25 battute
  - Facciamo ruotare Scratchy di 180 gradi
  - Facciamo aspettare Scratchy per 1 secondo
  - Facciamo muovere Scratchy 10 passi a sinistra
  - Facciamo suonare il tamburo 1 per 0,25 battute

Un semplice algoritmo - 2

- In molti casi reali l'esecuzione di un algoritmo è guidata dagli eventi: ciò significa che determinate azioni devono venire eseguite quando si verifica un evento.
  - Facciamo iniziare la sequenza di azioni di Scratchy quando viene cliccata la bandiera verde.

Un semplice algoritmo - 3

Iterazione - 1

- In algoritmi più complessi può essere utile ripetere le stesse istruzioni per un determinato numero di volte, o finché non si verifica una certa condizione, ad esempio:
  - Facciamo in modo che Scratchy ripeta il suo movimento 4 volte (usiamo il blocco “ripeti” nella categoria “controllo”).

Un semplice algoritmo - 1.1

- Scrivere il programma per il seguente algoritmo (sequenza di passi elementari):
  - Facciamo muovere Scratchy 10 passi a destra
  - Facciamo suonare il tamburo 1 per 0,25 battute
  - Facciamo ruotare Scratchy di 180 gradi
  - Facciamo aspettare Scratchy per 1 secondo
  - Facciamo muovere Scratchy 10 passi a sinistra
  - Facciamo suonare il tamburo 1 per 0,25 battute

Un semplice algoritmo - 1.1

- ... provare a scrivere un programma Scratch che implementa l'algoritmo della slide precedente.

- NOTA: Non abbiamo visto forse tutti i costrutti necessari ma Scratch è proprio pensato per essere molto intuitivo e favorire in un certo senso l'autoapprendimento: si può imparare per tentativi...

Iterazione - 2

Variabili - 1

- Spesso è utile poter ricordare le informazioni che si stanno elaborando, come se le si annotasse su un foglio di carta.
  - Per ricordare le informazioni si utilizzano le variabili. Una variabile è un "contenitore" di dati identificato in modo univoco dal suo nome. Il contenuto di una variabile è definito "valore".

Variabili - 2

- Le variabili possono essere richiamate per utilizzarne il valore/contenuto.
- Le variabili sono contenitori temporanei, quindi vengono cancellate al termine dell'esecuzione dell'algoritmo/programma.

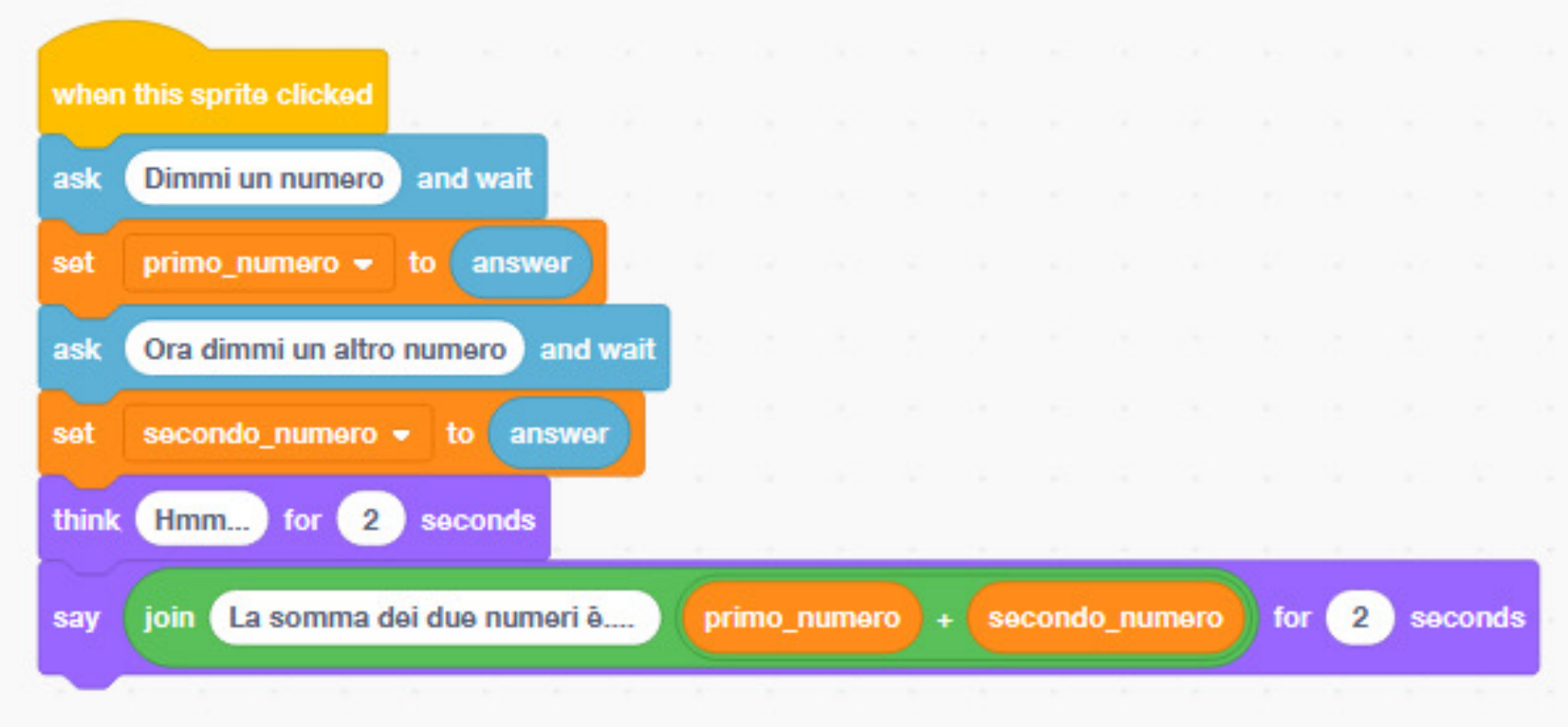
Variabili - 3

- Proviamo ad utilizzare le variabili:
  - Creiamo due variabili, “primo_numero” e “secondo_numero” (usiamo il comando “crea una variabile” nella sezione “Variabili”).
  - Facciamo chiedere a Scratchy un numero.
  - Memorizziamo il numero nella variabile “primo_numero” .
  - Facciamo chiedere a Scratchy un secondo numero.
  - Memorizziamo il numero nella variabile “secondo_numero” .
  - Facciamo pensare Scratchy per due secondi.
  - Facciamo dire a Scratchy quanto fa la somma dei due numeri.
  - Facciamo iniziare la sequenza di azioni quando viene cliccato lo sprite che rappresenta Scratchy.

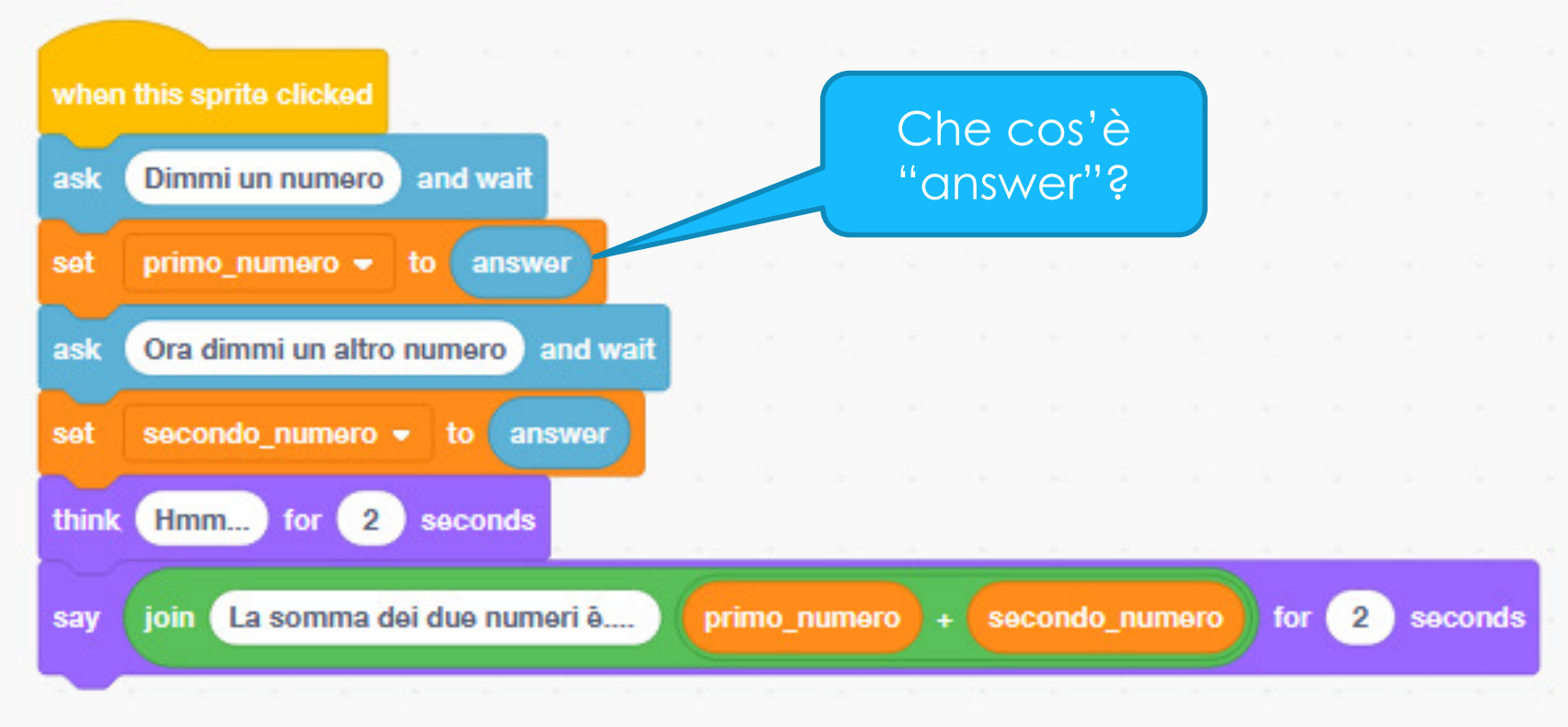
Variabili - 3

- Proviamo ad utilizzare le variabili: ...
  - ....PROVARE a implementare tutti i passi della slide precedente
  - si tratta di cercare di capire come possono essere usati i costrutti del linguaggio per questo scopo ... trials and errors

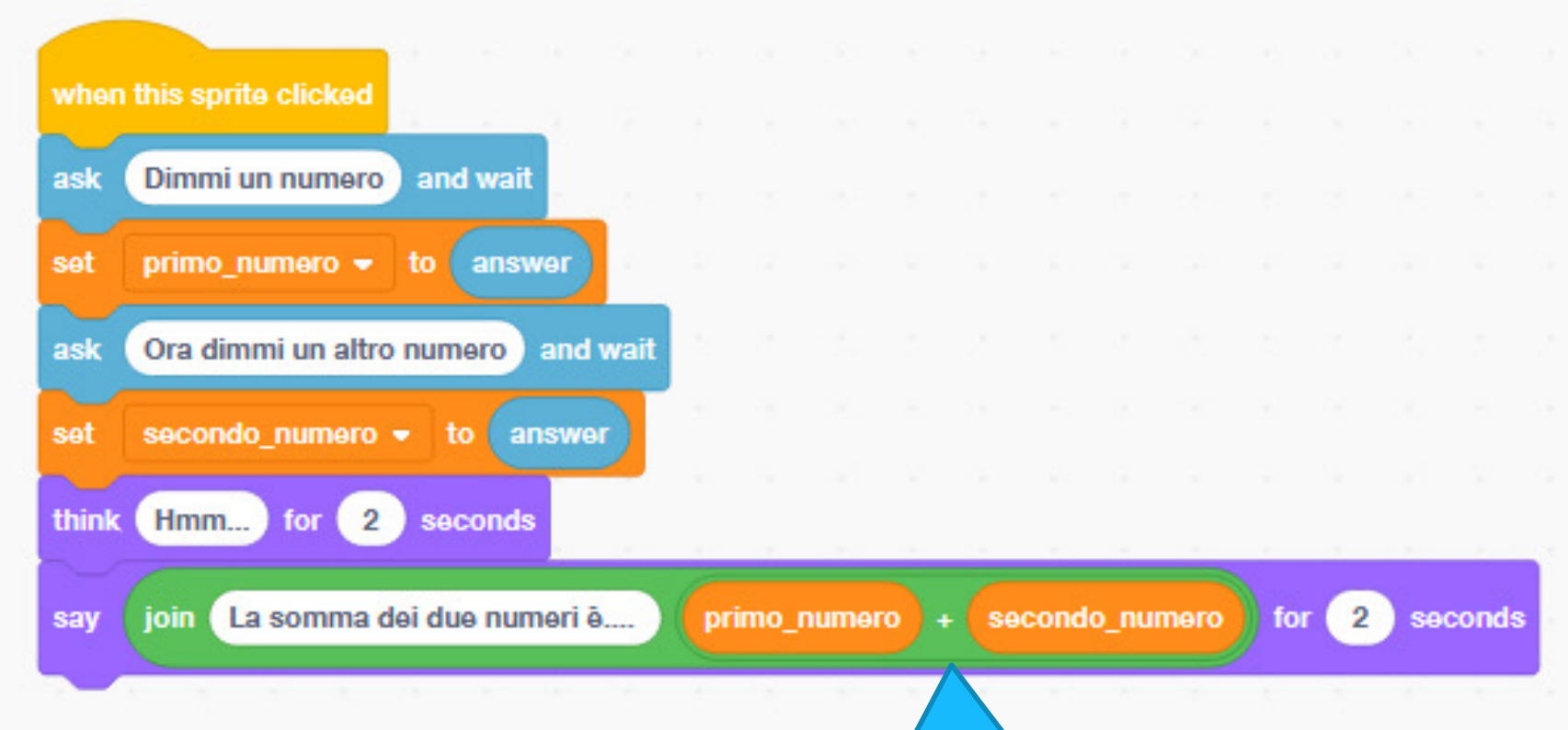
Variabili - 4


when this sprite clicked
ask Dimmi un numero and wait
set primo_numero to answer
ask Ora dimmi un altro numero and wait
set secondo_numero to answer
think Hmm... for 2 seconds
say join La somma dei due numeri è.... primo_numero + secondo_numero for 2 seconds

Variabili - 4

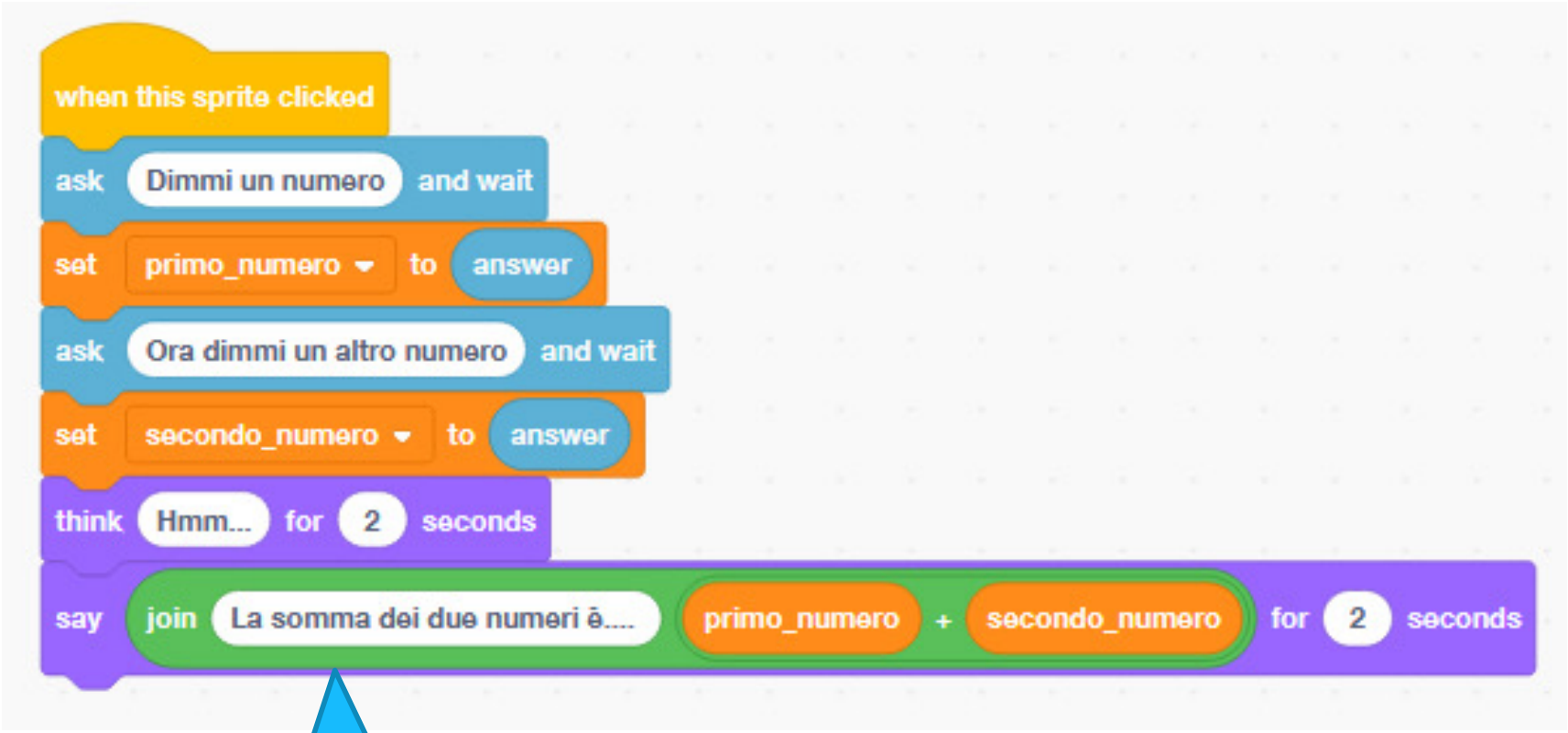

when this sprite clicked
ask Dimmi un numero and wait
set primo_numero to answer
ask Ora dimmi un altro numero and wait
set secondo_numero to answer
think Hmm... for 2 seconds
say join La somma dei due numeri è.... primo_numero + secondo_numero for 2 seconds
Che cos'è "answer"?

Variabili - 4



Che cosa rappresenta questo blocco?

Variabili - 4


when this sprite clicked
ask Dimmi un numero and wait
set primo_numero to answer
ask Ora dimmi un altro numero and wait
set secondo_numero to answer
think Hmm... for 2 seconds
say join La somma dei due numeri è.... primo_numero + secondo_numero for 2 seconds
E questo?

Condizioni - 1

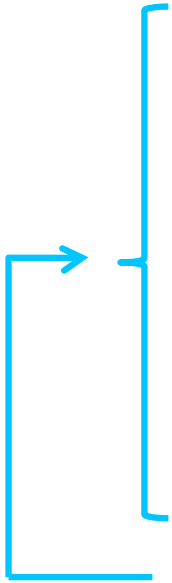
- In alcuni casi è utile che un algoritmo possa "prendere delle decisioni" e richiedere l'esecuzione di insiemi alternativi di istruzioni sulla base del verificarsi (o meno) di una certa condizione.
  - Le condizioni prevedono un confronto tra variabili e valori. Il risultato del confronto sarà sempre o "vero" o "falso".
  - Ad esempio, possiamo fare in modo che Scratchy chieda quanti passi debba fare, ma si sposti solo se il numero di passi richiesti è pari.

Condizioni - 2

- Proviamo ad utilizzare le condizioni:
  - Facciamo chiedere a Scratchy il numero di passi da fare.
  - Verifichiamo se il numero di passi sia pari o dispari (utilizziamo il blocco "modulo" della sezione "Operatori" per verificare la condizione e il blocco "se … /allora…/ altrimenti…" della sezione "Controllo" per gestire il flusso di esecuzione):
    - Se il numero di passi è pari, facciamo fare a Scratchy il numero di passi stabilito.
    - Altrimenti, facciamo in modo che Scratchy pensi "Numero dispari" per 2 secondi.
  - Facciamo chiedere a Scratchy il nuovo numero di passi da fare.
  - Facciamo in modo che Scratchy ripeta le ultime due operazioni finché la risposta dell'utente non è "0".
  - Facciamo iniziare la sequenza di azioni di quando viene cliccato lo sprite che rappresenta Scratchy.

Condizioni - 2

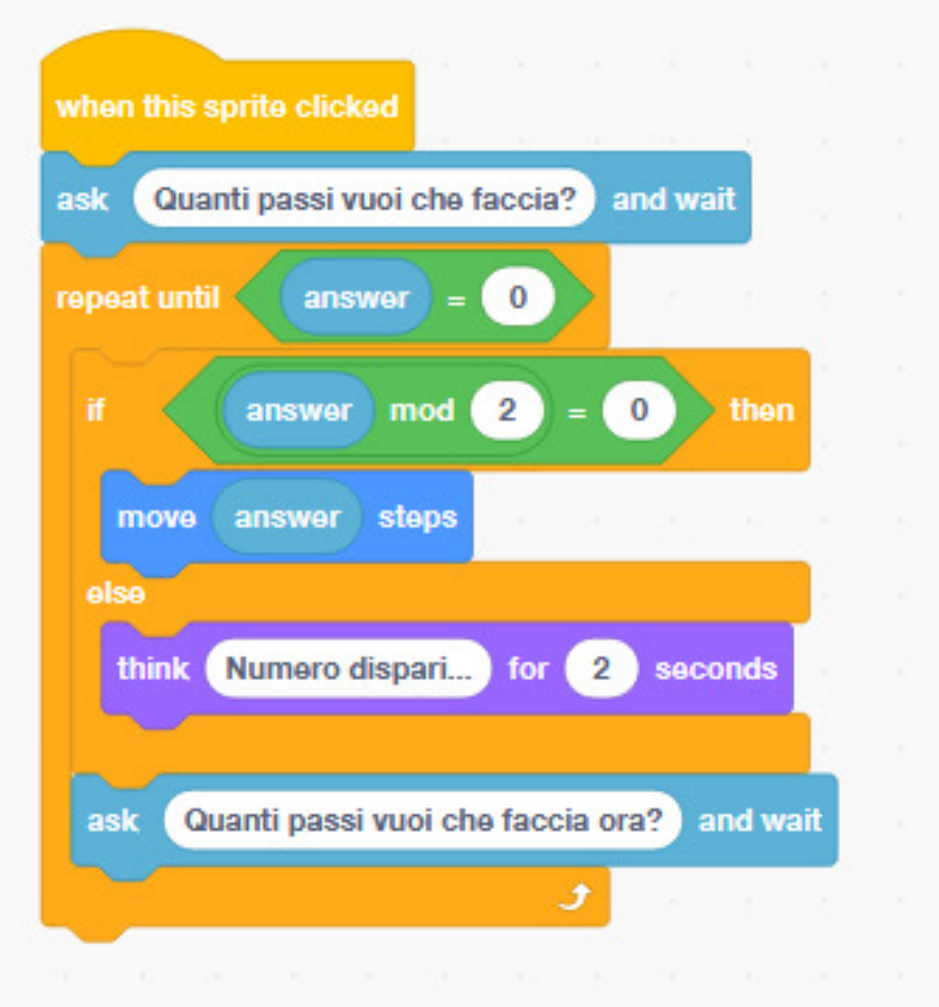
- Proviamo ad utilizzare le condizioni: ....



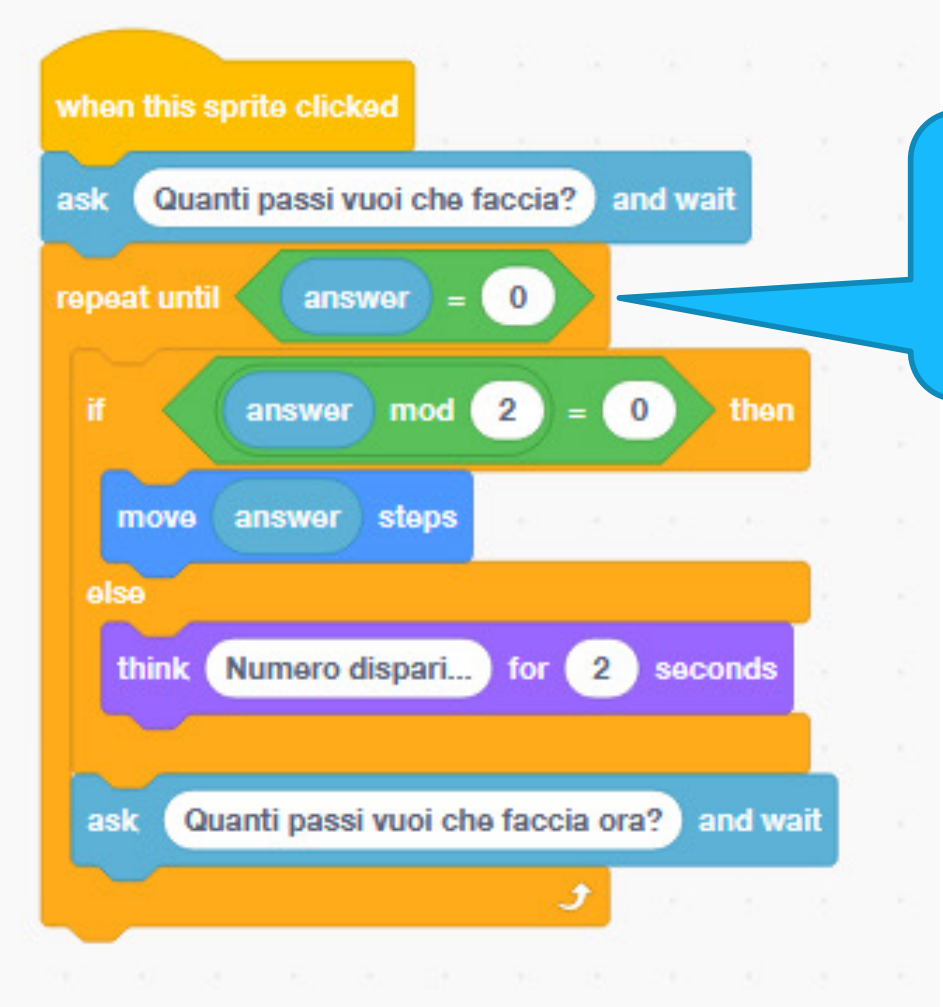
- ....PROVARE a implementare tutti i passi della slide precedente
- si tratta di cercare di capire come possono essere usati i costrutti del linguaggio per questo scopo ...
- fino a quando non funziona tutto

trials and errors!

Condizioni - 3



Condizioni - 3



Su quale base si decide se ripetere o meno la sequenza di azioni che segue?

Coordinamento - 1

- Gli sprite si possono attivare in risposta a svariati eventi, ad esempio:
  - Quando si clicca sulla bandiera verde
  - Quando si clicca sullo sprite stesso
  - Quando si preme un tasto
  - Quando lo sprite viene toccato da un altro sprite
- Inoltre, gli sprite possono coordinarsi tra di loro attraverso lo scambio di messaggi:
  - Uno sprite può "passare la parola" inviando un messaggio
  - Uno sprite può attivarsi quando riceve un particolare messaggio

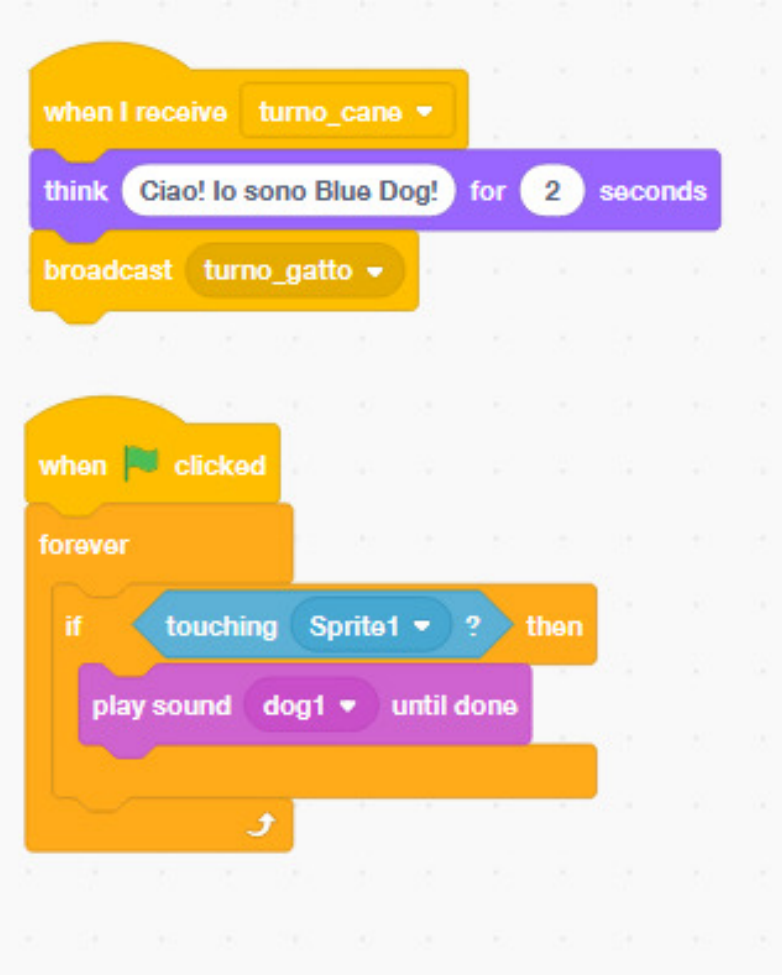
Coordinamento - 2

- Facciamo parlare tra loro due sprite, coordinando il dialogo:
  - Inseriamo due sprite: Scratchy e Blue Dog.
  - Facciamo parlare Scratchy per primo. Quando finisce, facciamo in modo che Scratchy mandi un messaggio al cane per "passargli la parola".
  - Facciamo in modo che Blue Dog parli quando riceve il messaggio di Scratchy. Quando finisce, facciamo in modo che Blue Dog mandi un messaggio al gatto per "ri-passargli la parola".
  - Facciamo in modo che Scratchy riprenda la parola quando riceve il messaggio di Blue Dog.
  - Inoltre:
    - Facciamo in modo che Scratchy si muova in modo casuale sullo stage (utilizziamo il blocco "vai a posizione casuale" della sezione "Movimento").
    - Facciamo in modo che il cane abbai se viene toccato dal gatto.

Coordinamento - 3

Script di Scratchy

Coordinamento - 4



Script di Blue Dog

Esercizio 8

- Scrivere uno script che rappresenti e risolva il seguente problema:
  - Cinque ragazzi vanno al cinema approfittando della seguente promozione: **ogni 5 persone un biglietto è gratis** (quindi pagheranno solo 4 biglietti). Ogni biglietto ha il costo di 8 €. I ragazzi divideranno tra loro la spesa in modo equo.
    - Si vuole calcolare quanto pagherà ciascun ragazzo.
    - Si vuole chiedere all'utente di provare ad indovinare il risultato e confrontare la sua risposta con quanto calcolato.

Esercizio 8

- Scrivere uno script che rappresenti e risolva il seguente problema: ....
  - Impegnativo e divertente!
  - Da fare a gruppi di 2.

Esercizio 9

- Scrivere uno script che rappresenti e risolva il seguente problema:
- Si vogliono proporre **10 tabelline** random. L'utente darà la risposta e, se corretta, avanzerà di un punto, altrimenti non gli sarà assegnato nulla. Quando termineranno le 10 domande si controllerà il punteggio. Se l'utente totalizza:
  - **Meno** di **6 punti**, gli suggeriamo di ripassare le tabelline;
  - **6 o 7 punti**, gli diamo come premio una **ciambella**;
  - **8 o 9 punti** gli diamo come premio un **pallone**;
  - **10 punti**, gli diamo come premio una **macchinina**.

Strutture dati: le liste

Liste - 1

- Quando si vogliono memorizzare molti valori con lo stesso significato (ad esempio, la temperatura media in diversi giorni dell'anno) può non essere conveniente utilizzare delle variabili, specie se il numero di valori è sconosciuto a priori.
  - Le liste consentono di associare più valori ad un unico nome di variabile.
    - Possiamo immaginare le liste come variabili "più capienti".
    - Le liste sono strutture dati ordinate (si tiene traccia della posizione in cui è memorizzato un valore).

Liste -2

- Variabile

<nome>

<valore>

numero

45

- Lista

<nome>

1	<valore>
2	<valore>
…	<valore>
N	<valore>

studenti

1	Maria
2	Gianni
…	Alfio
N	Sandra

Liste - 2

- Proviamo a memorizzare una sequenza di numeri e poi a ripeterla:
  - Creiamo una lista “numeri” e assicuriamoci di svuotarla (usiamo il comando “crea una lista” e il blocco “cancella tutto” nella sezione “Variabili”).
  - Facciamo in modo che Scratchy chieda all’utente di dire un numero (oppure “fine” per terminare l’interazione).
  - Facciamo in modo che Scratchy ripeta le seguenti operazioni finché l’utente non dice “fine”:
    - Aggiungere la risposta dell’utente alla lista.
    - Chiedere all’utente se voglia aggiungere altri numeri.
  - Facciamo in modo che Scratchy “annunci” la lettura della lista.
  - Creiamo una variabile “contatore” per tenere il conto delle posizioni della lista che leggeremo mano a mano.

Liste - 3

- Sapendo che le posizioni della lista sono numerate da 1 fino ad un valore pari alla lunghezza della lista:
  - Impostiamo la variabile contatore a 1.
  - Facciamo in modo che Scratchy ripeta le seguenti operazioni finché il valore della variabile contatore non eccede la lunghezza della lista:
    - Leggere dalla lista il valore che si trova nella posizione pari al valore del contatore.
    - Aumentare di uno il valore del contatore.

Liste - 4


when clicked
delete all of numeri
ask Dimmi un numero, oppure scrivi "fine" per terminare and wait
repeat until answer = fine
add answer to numeri
ask Altro? and wait
say Ora leggerò i numeri che mi hai detto... for 2 seconds
set contatore to 1
repeat until contatore > length of numeri
say item contatore of numeri for 2 seconds
change contatore by 1