

Promuovere la creatività con attività di programmazione

La prima parte di quanto segue è derivata dal materiale preparato da Martina Kabátová (kabatova@fmph.uniba.sk) e Katarína Mikolajová (mikolajova@fmph.uniba.sk) - Dept of Informatics Education, Comenius University, Bratislava – per lo Scratch WORKSHOP “Fostering creativity through programming” durante ISSEP 2011 del 29 ottobre 2011. Anche nella seconda parte alcune delle attività hanno la stessa origine. **Grazie dunque a Martina e Katarína che ne hanno concesso l'uso.**

Informazioni utili:

- **Sito web di Scratch:** <http://scratch.mit.edu>
- **dove è il download:** l'ambiente di programmazione Scratch è disponibile in molte lingue e può essere scaricato gratuitamente <http://scratch.mit.edu/download>
- **Sito web per gli insegnanti:** <http://scratched.media.mit.edu>

Cosa è Scratch?

Scratch è un ambiente di sviluppo programmi con un proprio linguaggio di programmazione usando il quale ciascuno può creare facilmente storie interattive, animazioni, giochi, musica e arte. Importante componente del progetto è la comunità Scratch sul web dove si è invitati a condividere le creazioni realizzate in modo che altri possano vederle, guardare come sono fatte e magari modificarle mettendo in pratica l'approccio *guardaCapisciModifica* con cui gli autori hanno concepito il progetto.

Scratch consente a chi lo usa di creare animazioni, giochi interattivi, storie, simulazioni oppure di modificare in modo semplice creazioni altrui - cambiando sfondi e oggetti, scegliendo immagini da librerie di immagini messe a disposizione o fare figure e scene proprie con immagini soddisfacenti lo stile e le preferenze personali.

Creando e condividendo attività Scratch i ragazzi possono imparare concetti matematici e computazionali importanti, e insieme imparano a pensare in modo creativo, a ragionare in modo sistematico e a lavorare in modo collaborativo.



Obiettivo

del laboratorio è esplorare attività e approcci che consentano agli studenti di esprimere le loro idee tramite la programmazione e offrano modo di sviluppare la loro creatività durante questo processo. Quindi *la programmazione come un linguaggio in più oltre ai Cento Linguaggio delle scuole di reggio Emilia.*

Ci si concentrerà su più progetti multimediali e giochi che forniscono agli studenti l'opportunità di usare la loro immaginazione e insieme danno loro una certa libertà di esprimere se stessi in modo insolito. Cercheremo di introdurre attività selezionate sul campo di tipo diverso e con risultati aperti.



1. Introduzione a Scratch

Ci sono due versioni di Scratch: la 1.4 e la 2.0. Le useremo entrambe: quella più recente è più ricca, quella precedente ci permette di vedere senza problemi attività realizzate con quella versione.

Vediamo per prima cosa un esempio. Apriamo l'attività FarfallaBlu realizzata in una classe di scuola secondaria di primo grado e guardiamo la schermata iniziale nella figura qui sotto.

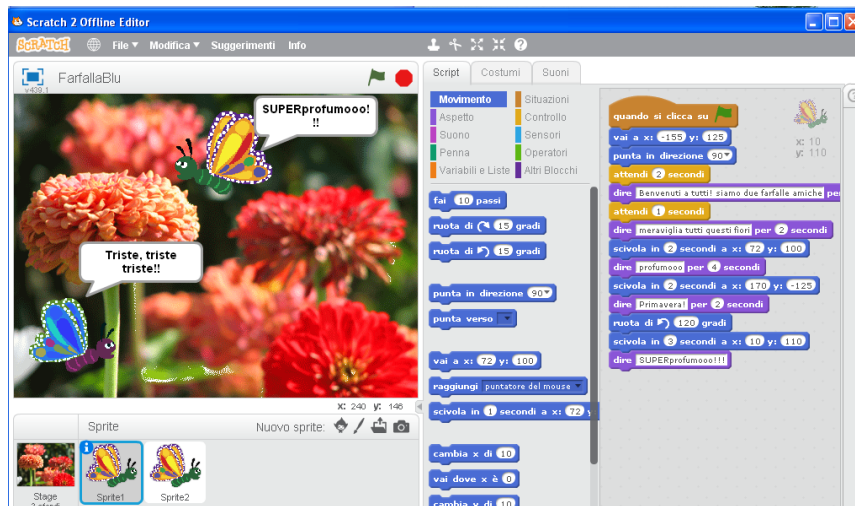


Figura 1. Una schermata per l'attività FarfallaBlu

Nella finestra di Scratch 2.0 compaiono: in orizzontale in alto una *barra-comandi*, di cui diremo via via, e tre colonne verticali a loro volta divise in due o più sezioni (o sottofinestre):

- **Colonna sinistra in alto – Stage o Scena:** come abbiamo visto in FarfallaBlu e come dice il suo nome è lo spazio in cui le creazioni prendono vita, cioè dove sono “recitate” le storie. Può essere ingrandita ad occupare l'intero schermo.
- **Colonna sinistra in basso - Lista degli Sprite o attori:** Quest'area contiene il fondale della scena e l'elenco degli attori che compaiono nelle storie (selezionate gli attori con un clic).
- **Colonna destra - Area degli script cioè della parti** (in senso teatrale): ogni attore recita cioè si comporta come specificato dalla sua parte o Script. Uno Script è un programma associato a un attore ed è composto di comandi che l'autore degli script sceglie tra quelli della colonna centrale di cui si dice tra poco
- **Colonna centrale in alto** - ci sono le linguette (o tab) **Script, Costumi, Suoni:** pigiando questi tab script, costumi, o suoni associati agli attori si definiscono rispettivamente: le loro parti (script in inglese è la parte di un attore), i costumi cioè le immagini con cui appariranno

durante la “recitazione” e i suoni (frasi registrate o canzoni o altra musica) che ciascun attore emette in diversi momenti. Provare a pigiare su ciascun tab.

- **Colonna centrale, in alto (sotto le linguette) - Paletta dei comandi (o block palette):** le istruzioni con cui possiamo realizzare le attività sono molte ed il loro elenco occuperebbe una colonna molto lunga, scomoda da consultare. Per questo le istruzioni sono divise in dieci insiemi di comandi corrispondenti ai dieci bottoni che si vedono in alto in centro ciascuno con un proprio nome e colore. Torneremo sui criteri della suddivisione dell’insieme dei comandi in sei sottoinsiemi. Quando si seleziona un sottoinsieme i comandi relativi sono disponibili nella medesima colonna (a sinistra) ma in basso.
- **Colonna centrale in basso - comandi (o istruzioni):** contiene i comandi del sottoinsieme selezionato in Paletta. Per comporre gli script i comandi vanno trascinati nella colonna destra. Le tessere-comando si incastrano insieme come pezzi di un puzzle.



2. Mani in pasta --- Attività pratica:

Progetto 1

Aprire Scratch e poi il file FarfallaBLu.sb :

1. cambiare lo sfondo (vedere poco oltre) cercando prima un altro sfondo che ci piaccia e poi il comando adatto nell’insieme “Aspetto”
2. le frasi delle farfalle sono adatte al nuovo sfondo?

Nella figura 2 vediamo invece la schermata iniziale di Scratch1.4 senza apertura di nessun file con attività già pronta

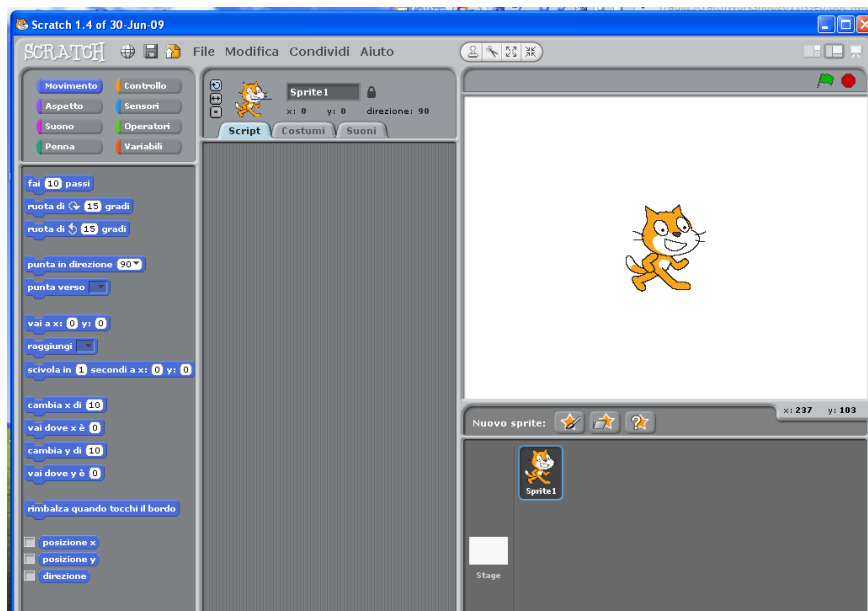
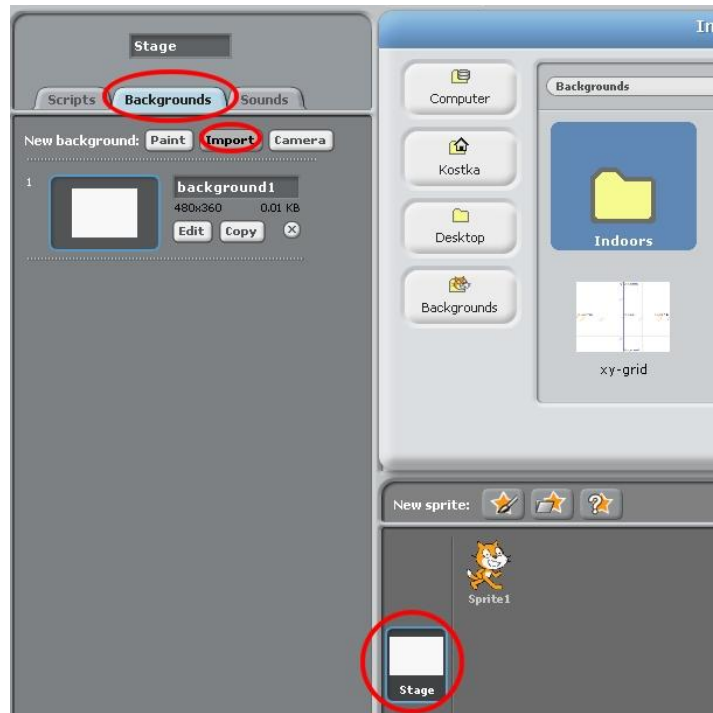


Figura 2. Schermata iniziale di Scratch1.4

Per cambiare il fondale della scena: Clicca su "stage" nella lista degli Sprite. Scegliere "sfondo" tra i tab. Fare clic su Importa e selezionare l'immagine che si desidera utilizzare come sfondo per la scena.



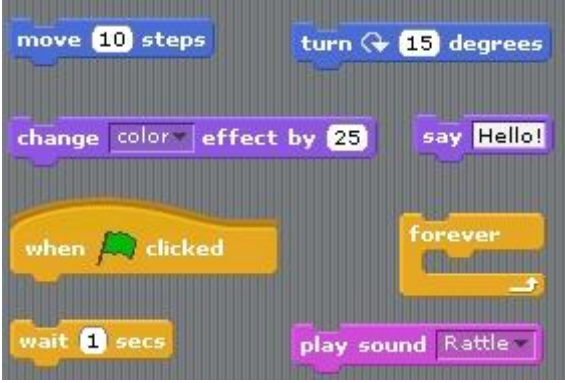

Come aggiungere un nuovo sprite alla storia: Fare clic sul pulsante "Nuovo sprite da file" nella lista Sprite. Scegliere l'immagine per il nuovo sprite.





3. Cominciamo con l'usare otto soli comandi

Attività pratica: Crea una breve storia con un fondale, uno sprite (cioè un attore) ed uno script per quell'attore utilizzando soltanto 8 comandi Scratch.

Progetto 2	<p>Ricordiamo che i comandi appartengono a insiemi differenti indicati dal colore della tessera su cui il comando è scritto. Crea una attività usando soltanto gli otto in figura:</p> 
Progetto 3	<p>Migliora il tuo progetto utilizzando altri quattro comandi:</p> 
Fai vedere e illustra	<p>Riposiamoci guardando le carte Scratch:</p> <p>Mostra ai tuoi compagni/colleghi di classe l'attività realizzata nel progetto 3:</p> <ul style="list-style-type: none">• Che cosa fa?• Quali sono le prime impressioni nell'usare Scratch come un gioco?



4. La comunità Scratch e altre attività

Scratch ha una comunità molto viva di utenti entusiasti. Sul sito ufficiale di Scratch chiunque può registrare e caricare un proprio progetto da condividere con gli altri. I progetti spaziano da strumenti interattivi di musica , giochi , applicazioni didattiche , labirinti , simulazioni , storie animate a tutto il possibile . Alcuni esempi :

- **Pianoforte:** <http://scratch.mit.edu/Attivitàs/Animecat33/1209527>
- **Twig's Adventure- Maisie's Maze:** <http://scratch.mit.edu/Attivitàs/7scratch7/1207684>
- **Fantasy RPG:** <http://scratch.mit.edu/Attivitàs/wedsneday/1155407>



Nel
web

Vai al sito web dei progetti <http://scratch.mit.edu/channel/featured> e cerca qualcosa che ti piaccia. Esplora pochi progetti, decidi a quale tipo di attività possono appartenere (ad esempio storia animata, gioco, labirinto, quiz, ...) e parlane coi colleghi.

Nelle pagine del sito di Scratch leggiamo: "A wide variety of educators have been supporting Scratch creators, in both formal and informal learning environments: a teacher who wants to share stories about Scratch and cross-curricular integration; a researcher who wants feedback on materials developed for exploring Scratch as participatory literacy; a parent who wants advice on how to introduce Scratch at a local all-girls high school; a museum program director who wants to connect with other museums who have introduced Scratch. In response to this growing community of educators working with Scratch, we developed ScratchEd."

Lanciata nel luglio 2009, ScratchEd è una nuova comunità online in cui gli educatori che usano Scratch **condividono** storie, **scambiano** risorse, **fanno domande** e trovano altri con cui condividere interesse per qualche attività

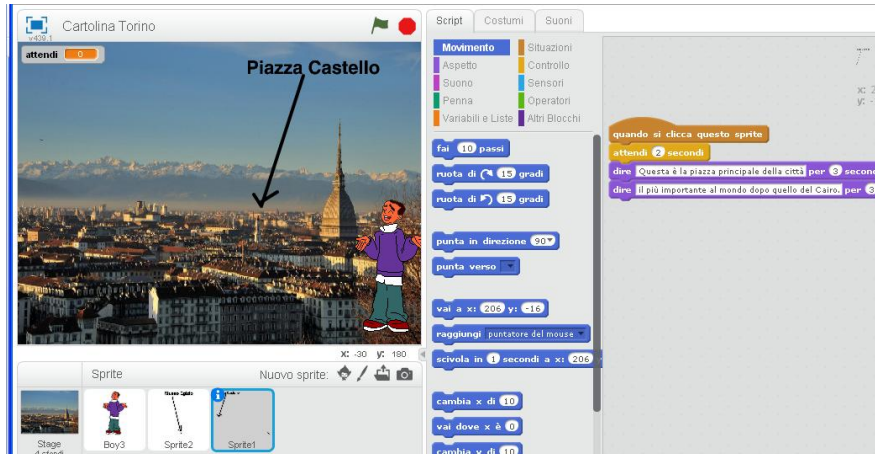
Nel
web

Vai all'indirizzo <http://scratched.media.mit.edu/> e cerca risorse che possano aiutarti su Scratch con la tua classe.



5. Cartoline e interazione

Ora vogliamo creare una “cartolina interattiva” cioè una attività Scratch dove almeno uno degli attori reagisce ad una azione proveniente dall'esterno (cioè eseguita da chi ha in mano la tastiera o il mouse per esempio) come accade nell'esempio della cartolina cui si riferisce la figura che segue su una visita a Torino che è nei materiali.



6. Crea la tua cartolina

Attività pratica: A gruppi o singolarmente create la vostra cartolina.

Attività 3

Creare la tua cartolina:

- Scegliere una destinazione,
- trovare sul web una bella immagine che possa comparire come sfondo della cartolina
- aggiungere degli attori (cioè degli sprite), farli muovere, farli emettere un suono quando accade qualcosa: come si accorge uno sprite che accade qualcosa? Cosa può accadere?
- aggiungere qualche frase tipo “Benvenuti a XXX”

Cosa abbiamo imparato

Discutiamo insieme:

- Guardate insieme a qualcun altro le vostre cartoline e condividete osservazioni su Scratch venute fuori durante la creazione delle cartoline.
- Scrivete una lista di suggerimenti per chi dovesse o volesse trovarsi anche lui a creare una cartolina
- Quali comandi Scratch hai trovato più utili o anche obbligatori da usare in un progetto interattivo?



7. Invento e costruisco una equazione con una variabile poi la risolvo

Anche con conoscenza di informatica limitate si possono costruire attività interdisciplinari come l'indovinello *IndovinoNumeroPensato.sb* e l'indovinello simile dove si deve capire *PerchéRimaneStessoNumero.sb*. Interesseranno i docenti di matematica.

Pensiamo al gioco "Pensa un numero":

- Pensa un numero
- Aggiungi 1
- Moltiplica per 3
- Sottrai il numero con cui hai cominciato
- Aggiungi 5
- Dimmi quale numero ti rimane
- E ora io indovino il numero con cui hai cominciato!

Chiamiamo x il numero iniziale e a il numero che rimane, il gioco si riassume nella equazione:

$$(x + 1) * 3 - x - 2 = a$$
$$x = (a - 1) / 2$$

The screenshot shows a Scratch project titled "PensaUnNumero". The script area contains the following code blocks:



- quando si clicca su
- pensa Pensa ad un numero... tra 1 e 9 per 4 sec
- pensa adesso sommagli 1 per 4 secondi
- pensa adesso moltiplica per 3 il valore ottenuto...
- pensa e sottraigli il valore pensato da te all'inizio p
- pensa e per finire, sottraigli ancora 2 per 4 secon
- chiedi Adesso dimmi il risultato che hai ottenuto e
- porta risultato a risposta - 1 / 2
- dire unione di Il numero che avevi pensato è: e
- ferma lo script

The stage area shows a female character with a speech bubble saying "Il numero che avevi pensato è: 1". The "risultato" field displays "1".



8. Libera la tua creatività – idee di attività

Attività pratica: crea piccole attività basandoti sulle idee che seguono. Poi sviluppane una in un progettino più ampio degli altri tuo piacimento .

Attività 4	<p>Balliamo insieme</p> <p>Use several figures to create dance theatre - each figure will dance in a different way. Find suitable background and dance music. You can add some extra effects - e.g. small bird will suddenly fly over the stage. Use color effects and changing of costumes. You may add music and sounds.</p>	
Attività 5	<p>Acquario</p> <p>Create interactive fish tank. Put inside various fish. They move around like fish do and when clicked they do something - e.g. change the direction, or do flip-flop.</p> <p>You can add can with fish food - invent some way to feed the fish.</p>	
Attività 6	<p>Dizionario di disegni interattivi</p> <p>Crea un dizionario di disegni interattivi su un tema particolare, per esempio una fattoria o una biblioteca - find suitable background (farm) and pictures for sprites (animals). When the sprite-animal is clicked it will pop a bubble with English and Slovak name of the animal.</p>	
Attività 7	<p>Il castello spaventoso</p> <p>Create una scena con mostri e fantasmi che si muovono e fanno rumori misteriosi.</p>	
Attività 8	<p>L'orchestra degli extraterrestri</p> <p>Creare una attività in cui ci siano vari extraterrestri. Quando col mouse si clicca su un orchestrale quello suona un strumento diverso da quello degli altri.</p> <p>Se non si hanno casse far saltare il personaggio su cui si clicca.</p>	



9. "Racconta una storia"

La creazione di una storia in Scratch offre l'opportunità di usare molti concetti fondamentali del pensiero computazionale. Qui di seguito sono raccolti alcuni dei blocchi molto utili nelle storie.

WAIT

Insert a pause



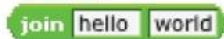
VISIBILITY

Make a sprite appear or disappear



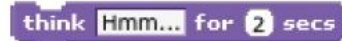
STRINGS

Test, access, and change words and sentences



SAY/THINK

Have a speech or thought bubble appear over a sprite



COSTUMES

Change the appearance of your sprite



ASK

Get input to use in a project



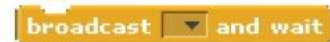
SOUNDS

Play recorded audio



COORDINATE

Synchronize actions between and within sprites



Attività 5. Una presentazione

Creare una presentazione con un insieme di immagini di sfondo su cui viene raccontata una storia oppure su cui si legge una storia. Si veda l'esempio che segue.



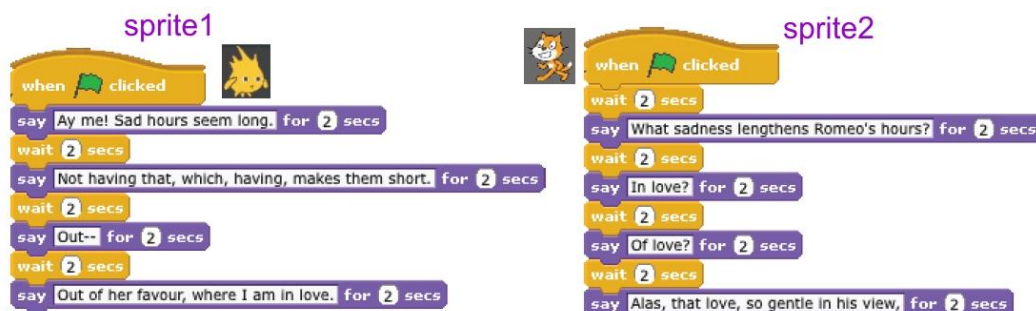


10.1 Crea tu una storia

Attività pratica: Creare una storia con degli attori, vari sfondi, testo e suoni.

Attività

Conversazione. Far chiacchierare due attori usando i blocchi dire e aspetta per coordinare la conversazione. Ci si può ispirare con le storie che si trovano su internet oppure ci si inventa una propria storia.



Non dimenticare di scegliere degli sfondi appropriati e trovare delle figure per i personaggi adatti alle parti.



Miglioramento

Per migliorare le storie usare il blocco "invia messaggio" per coordinare la conversazione.

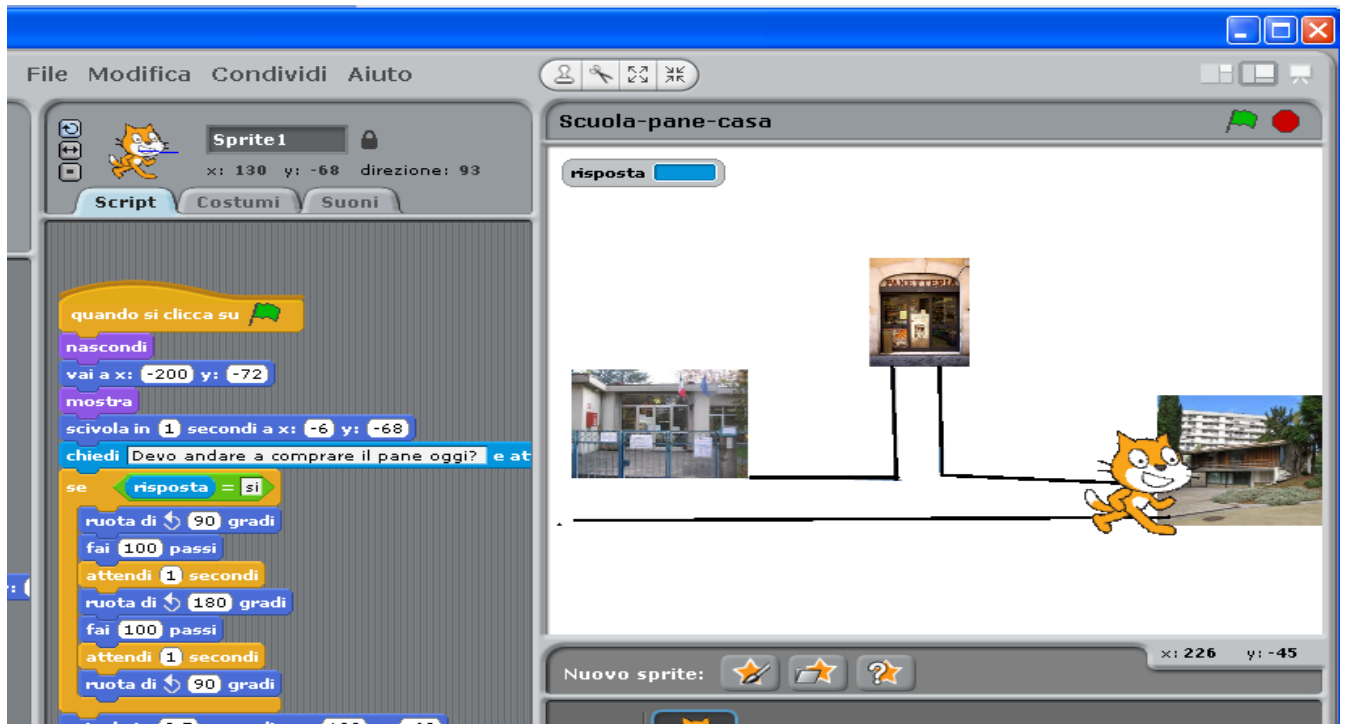


10.2 Racconta una storia tua

Vediamo insieme l'attività ScuolaPaneCasa, poi leggiamo lo script e in particolare ci soffermiamo sul comando di selezione

"se condizione sequenza-comandi altrimenti sequenza-comandi"

Il comando di selezione è stato introdotto con questa storia per sottolineare quanto esprima situazioni quotidiane dove già facciamo delle scelte.



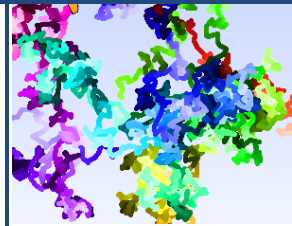
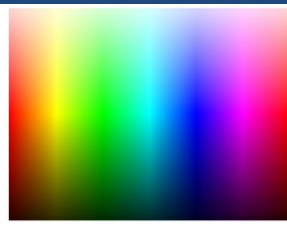
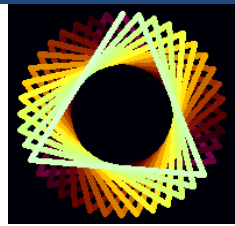
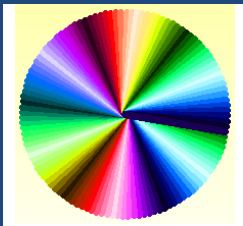

Attività pratica: raccontare in forma di storia Scratch qualcosa anche molto semplice della propria vita meglio se con attività differenti che vengono eseguite a seconda che succeda una cosa o un'altra (ad esempio scelgo di vedere un film romantico oppure un western a seconda che sia in compagnia di una certa amica o di un certo amico).

11.Three steps to foster creativity through given task:

- use **multimedia** to enable pupil's express their ideas in the original way (to raise pupils' motivation, attract them to programming),
- pose the problem where **result is given** but they can **explore and generate their own strategies** to get the result (problem should offer more than one solution to be solved),
- ask pupils to **come up with their own problems to solve** (teacher should be very flexible to adjust the problem so that pupils could solve them based upon their knowledge and skills).



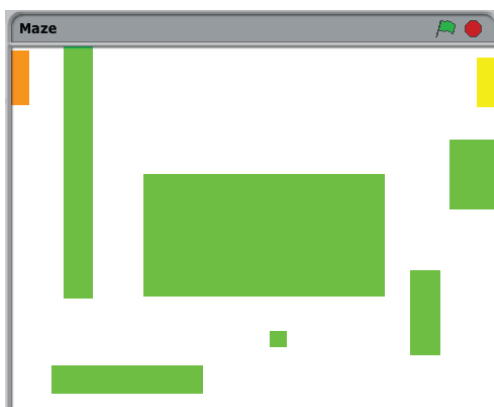
Giocare coi suoni, i colori, le forme, gli effetti

Attività 3	<p>Play with colors and shapes</p> <p>Create some nice colorful image. Play with lengths, colors and angles.</p> <div style="display: flex; justify-content: space-around;"></div>
Attività 4	<p>Xylophone</p> <p>Create a xylophone and make it sounded.</p> <div style="text-align: right;"></div>



I giochi offrono molte opportunità di esplorazione di concetti dell'informatica che finora non abbiamo visto.

Attività 6	<p>Il labirinto</p> <ul style="list-style-type: none">• GOAL: Get from the start of the maze to the end• RULES: Don't touch the green walls• OUTCOME: Win when the yellow marker is reached
------------	--



There will be 7 scripts for the sprite in shape of orange box. Green and yellow shapes are background. The scripts for orange box should do:

- direct the orange box in 4 directions and move it,
- bounce the orange box away from green walls,
- announce "You win!" in case the orange box have reached the yellow box,
- restart the position of orange box in case green flag is pressed.

Parte 2 – Qualche concetto di informatica in più

Nella prima parte di questo lavoro abbiamo introdotto Scratch come strumento per realizzare attività quali storie, indovinelli e giochi semplici. In questa seconda parte introduciamo concetti che ci permettono di creare in Scratch attività più complesse arricchendo gli aspetti di programmazione.



1. Quiz

Introduciamo qui quanto ci serve per realizzare attività più complesse a cominciare dal poter manipolare dei dati per esempio poter calcolare il punteggio di un giocatore impegnato in un gioco.

Attività-1 senza calcolatore. Organizzo un quiz di 5 domande con “premio” finale se chi risponde ha totalizzato un certo punteggio. Decido che una risposta corretta aumenta il punteggio di tre mentre una risposta sbagliata causa una diminuzione di due. *Ognuno provi a specificare come procedere per calcolare correttamente il punteggio finale avendo a disposizione un foglio di carta su cui scrivere.*

Passo 1. Se si fa una prova interrogando i presenti che tengono ciascuno nota del punteggio in genere tutti calcolano correttamente il risultato. Qui di seguito il procedimento descritto da uno dei presenti:

1. Faccio la prima domanda
2. Sento la risposta
3. Se la risposta è corretta scrivo sulla carta 3 altrimenti scrivo -2
4. Sento la seconda domanda e la risposta
5. Se la risposta è corretta aggiungo 3 al numero scritto sulla carta altrimenti gli tolgo 2
6. E così via ripeto le operazioni 4 e 5 per le domande successive fino alla quinta domanda e relativa risposta
7. Il numero scritto sulla carta è il punteggio finale.

Domanda: come faccio ad essere sicura che sono arrivata alla quinta domanda?

Risposta: conto anche le domande cioè scrivo anche il numero della domanda sulla carta.

Passo 2-a. Qui di seguito riscrivo il procedimento scritto sopra.

1. Faccio la prima domanda
2. Sento la risposta
3. Se la risposta è corretta scrivo sulla carta 3 altrimenti scrivo -2
4. Scrivo sulla carta 1 per tenere conto di quante domande ho posto
5. Sento la seconda domanda e la risposta

6. Se la risposta è corretta aggiungo 3 al numero scritto sulla carta altrimenti gli tolgo 2
7. Aggiungo 1 al numero che conta le domande (per evitare confusione col punteggio gli scrivo accanto un nome per esempio numDomande e do un nome anche al numero del punteggio (lo chiamo Punteggio)
8. E così via ripeto le operazioni 5 e 6 per le domande successive fino alla quinta domanda e relativa risposta. Arrivare alla quinta domanda vuol dire arrivare ad avere numDomande=5
9. Punteggio è il punteggio finale.

Passo 2-b. Altro modo per scrivere il procedimento sopra.

1. Faccio la prima domanda
2. Sento la risposta
3. Se la risposta è corretta scrivo sulla carta Punteggio = 3 altrimenti scrivo Punteggio = -2
4. Scrivo sulla carta 1 per tenere conto di quante domande ho posto
5. Sento la seconda domanda e la risposta
6. Se la risposta è corretta aggiungo 3 al numero scritto sulla carta altrimenti gli tolgo 2
7. Aggiungo 1 a numDomande
8. Se numDomande<5 torno al punto 5 altrimenti proseguo col punto successivo.
9. Il numero in Punteggio è il punteggio finale.

Domanda: E se di domande ne voglio porre sei o sette o un numero n ?

Risposta: semplice: devo arrivare ad avere numDomande=6 oppure 7 oppure n.

Passo 3. Allora per essere più precisi decidiamo quanto segue: dico per prima cosa quante domande farò e poi seguiranno le domande. Qui di seguito riscrivo il procedimento tenendo conto che abbiamo deciso di avere sulla carta tre numeri opportune etichette tre etichette per non confonderli tra loro : NumDomande, Punteggio, QualeDomanda.

1. Sento il primo numero e lo scrivo (per non dimenticarlo) accanto al nome NumDomande
2. Faccio la prima domanda
3. Sento la risposta
4. Se la risposta è corretta scrivo sulla carta Punteggio = 3 altrimenti scrivo Punteggio = -2
5. Scrivo sulla carta 1 e gli do nome QualeDomanda
6. Se QualeDomanda<NumDomande vuol dire che non ho ancora finito le domande quindi devo eseguire i punti dal 7 al 9
7. Sento la domanda successiva e la risposta
8. Se la risposta è corretta aggiungo 3 a Punteggio altrimenti gli tolgo 2
9. Aggiungo 1 a numDomande
10. Torno al punto 6 altrimenti proseguo col punto successivo.
11. Il numero in Punteggio è il punteggio finale.

Il quiz può essere giocato con una attività Scratch dove ci sono due attori: quello che fa le domande e quello che risponde. Chi fa le domande deve anche dire se la risposta data dall'altro attore è corretta. Si veda uno scheletro possibile per questa attività in SchelQuiz dove i due attori sono aiutati ciascuno da un gruppo di amici rispettivamente nel porre le domande e nel dare le risposte. Si suggerisce di iniziare provando le istruzioni dello scheletro suggerito, poi cominciare a completarlo, dopodiché modificarlo, alla fine trovandosi con degli script completi (e quindi una storia di contorno alle domande) che possono essere del tutto differenti dalla bozza iniziale.



Creiamo un quiz con punteggio finale

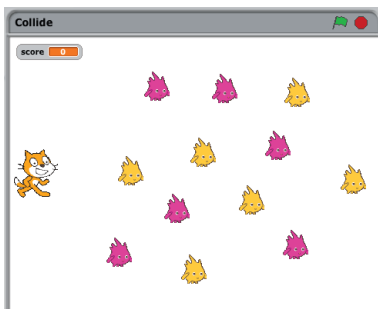
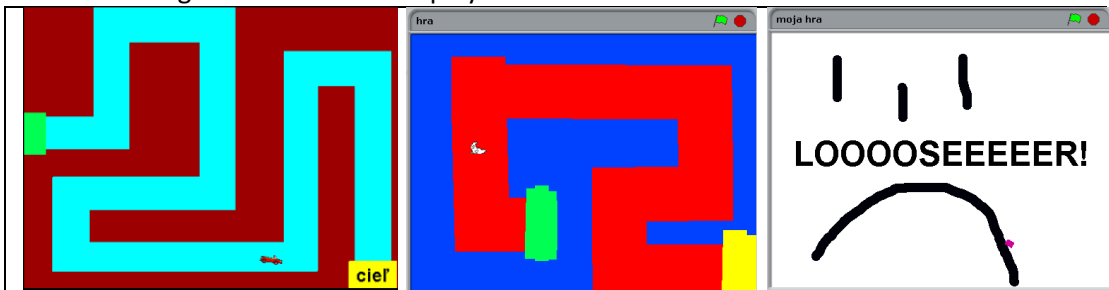
Impariamo ad usare le variabili progettando e realizzando una storia dove vengono poste domande al termine delle quali si da un punteggio.

Progetto 1	<p>Usiamo le variabili</p> <p>Progettare un quiz con domande su qualche argomento trattato a scuola</p> <p>Per ogni risposta esatta il punteggio aumenta di 1.</p>
Progetto 2	<p>Come il Progetto 1 ma calcolando diversamente il punteggio, ad esempio togliendo 1 per ogni risposta sbagliata e aggiungendo 2 per ogni risposta esatta.</p>



2. Progetta il tuo gioco

Migliora giochi della prima parte	<p>Modification</p> <ul style="list-style-type: none"> Make the player moving constantly in direction of mouse. In case the player have reached "the end" box, change the level to the more difficult one. You can add "losing screen" in case the player have reached "the forbidden" area.
Attività 7	<p>Collide</p> <ul style="list-style-type: none"> GOAL: Help the cat navigate a gobo minefield RULES: Collect yellow gobos to earn points, avoid pink gobos to avoid losing points OUTCOME: Maximize your score



Nel gioco della figura a sinistra lo sfondo non ha script. Ogni gobo ed il gatto sono sprite diversi.

Gli script del gatto:

- Risettano il gatto alla sua posizione iniziale,
- Conducono il gatto a seguire il mouse.

Gli script dei gobo gialli:

- quando il gatto si scontra con uno di loro questo sparisce e il punteggio aumenta di 10.

Script per i gobo rosa: quando il gatto si scontra con uno di loro questo sparisce e il punteggio diminuisce di 10.



Idee per attività varie di gioco e non

Hands on activity: Choose one of following prompts or make up your own Attività.

Attività 8

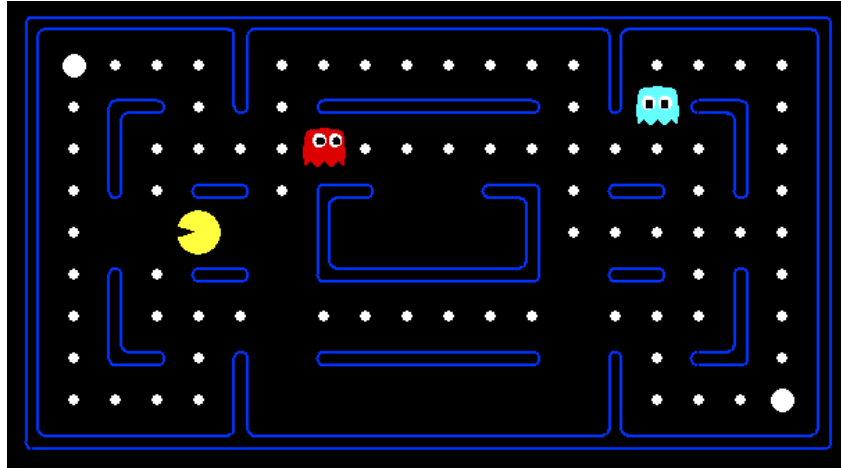
Favole di Esopo:

- Find some Esop's fable on the internet.
- Create a short animated story according the fable.
- Find appropriate characters and backgrounds, make them talk.
- Create a multi scene story with various backgrounds

Attività 9

Pacman

Create a pacman game - a character collects dots in a maze and has to avoid monsters.



3. Numeri numeri

Consideriamo ora un secondo problema da risolvere senza calcolatore cioè in modo unplugged.

Attività 2 senza calcolatore. Uno dei presenti dice una sequenza di numeri (almeno due) e gli altri devono alla fine dire quale è il maggiore di tali numeri. Normalmente tutti trovano lo stesso risultato, vediamo di specificare come.

PROBLEMA: *Trovare il massimo di n numeri detti ad alta voce senza usare fogli (su cui prendere appunti o simile). Abbiamo almeno due numeri.*

Passo 1. Diciamo ad alta voce una sequenza qualunque di almeno due numeri

Passo 2. Normalmente tutti i partecipanti trovano il massimo dei numeri detti. Chiediamo di pensare al procedimento attraverso il quale ciascuno è arrivato al risultato

Specifica del procedimento. Ciascuno descriva in linguaggio naturale il modo in cui ha proceduto, facendo attenzione a che il procedimento possa essere compreso, per esempio, da un bimbo che sa confrontare due numeri (magari limitandoci a numeri tra 0 e 30 o simile) ma non molto di più

Esempio di descrizione del procedimento in linguaggio naturale (cioè linguaggio usato quotidianamente in contrapposizione a linguaggio formale di cui diciamo più oltre):

1. Ascolto il primo numero
2. Tengo in mente il primo numero
3. Ascolto il numero successivo
4. Confronto il numero successivo con il numero che tengo in mente, se è maggiore tengo in mente il numero successivo, altrimenti continuo a tenere in mente il numero che avevo in mente prima del confronto
5. Chiedo se ci sono altri numeri, se no il numero che ho in mente è il maggiore dei numeri sentiti, FINE attività altrimenti:
6. Ascolto il numero successivo
7. Confronto il numero successivo con il numero che tengo in mente e così via fino a quando non ci sono più numeri.

Riscrivo il punto 7 più precisamente:

- 7'. Confronto il numero successivo con il numero che tengo in mente e vado a ripetere l'esecuzione delle operazioni 3, 4 e 5 fino a quando non ci sono più numeri

Ma allora posso scrivere il procedimento come segue:

1. Ascolto il primo numero
2. Tengo in mente il primo numero
3. Ascolto il numero successivo
4. Confronto il numero successivo con il numero che tengo in mente: se è maggiore tengo in mente il numero successivo altrimenti continuo a tenere in mente il numero che avevo in mente prima del confronto
5. Chiedo se ci sono altri numeri, se ci sono torno al punto 3 altrimenti : il numero che ho in mente è il maggiore dei numeri sentiti

Let's see and discuss some descriptions: are they precise? i.e. someone having no idea of how to perform the given task can get to the solution?

osservazioni: a) Scrivere le varie azioni in disordine: non ha senso è sbagliato

b) Per scegliere tra due diverse sequenze di azioni: come facciamo? Bisogna esprimere: b1) la causa per cui si sceglie l'una o l'altra sequenza poi b2) esprimere le due sequenze differenti.

Stessa procedura espressa in un linguaggio un poco più strutturato – ma struttura definita con regole che decido col “gruppo con cui lavoro”; lo chiamiamo pseudo-linguaggio:

1. Ascolto il primo numero e da ora lo chiamo PrimoNumero
2. NumeroInMente \leftarrow PrimoNumero (stabilisco voglia dire: a NumeroInMente do il valore di quello che chiamo PrimoNumero)
3. Ascolto il numero successivo che chiamo NumSucc
4. Se NumSucc > NumeroInMente allora NumeroInMente \leftarrow NumSucc
5. Se ci sono altri numeri allora vado al punto 3 altrimenti : NumeroInMente è il maggiore dei numeri sentiti.
6. Fine procedimento

Consideriamo ora un terzo problema da risolvere in modo unplugged.

Attività 3 senza calcolatore. Ogni giorno mi viene detta la temperatura media di quel giorno. A fine mese devo comunicare a qualcuno quale è stata la temperatura massima, quale la minima e in quali date si sono verificate. Non posso scrivere su carta ma ho telefono non smart usabile.

Da inserire seguendo i lucidi:

1. usa il cellulare per trovare max. min e media delle temperature di un mese e le date in cui si sono verificate temperature minima e massima
2. memoria cellulare \leftrightarrow memorie calcolatore (persistenti/non)
3. figura con architettura di von Neumann minima e esempi di device di I/O
4. Le variabili di un programma Scratch sono simili alla rubrica del cellulare: <nome, valore> , ma non persistenti



4. Algoritmi e programmi

Diagrammi di flusso o a blocchi o flowchart.

Esistono varie notazioni per la rappresentazione con diagrammi di flusso di algoritmi. Tutte le notazioni sottendono a un meta-modello molto semplice, caratterizzato da una lettura sequenziale:

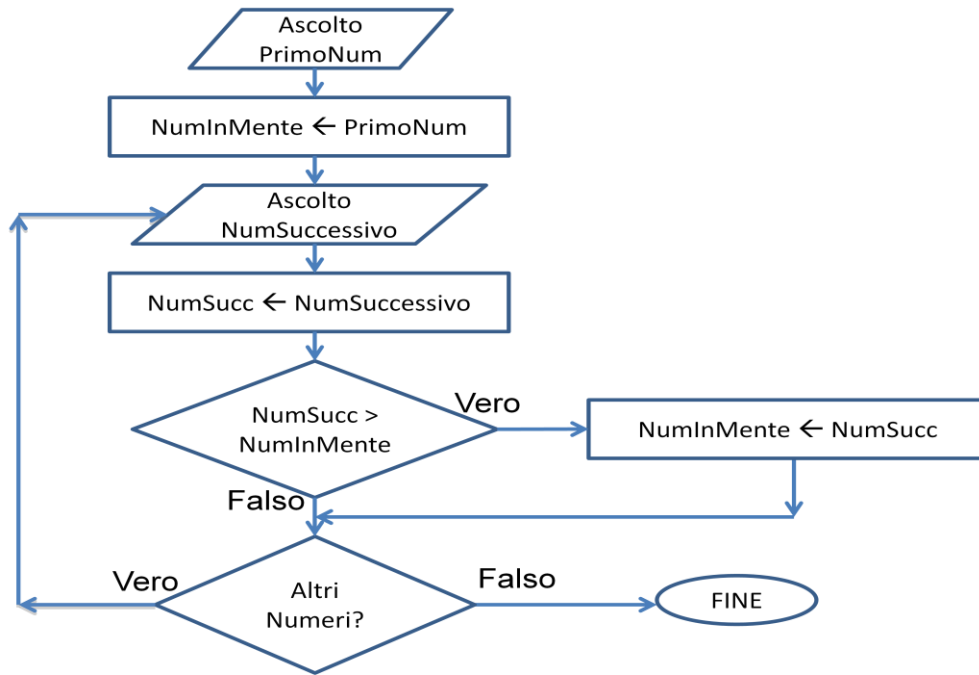
1. si parte dal blocco iniziale, graficamente rappresentato con una ellissi al cui interno è scritto INIZIO o START,
2. si segue la freccia in uscita,
3. si giunge al blocco successivo e si effettua l'operazione descritta nel blocco
4. si procede iterando i passi 2 e 3 fino a giungere al blocco finale, graficamente rappresentato con una ellissi al cui interno è scritto FINE o END.

Tra le operazioni si distinguono:

- azione, comportano una attività o un'elaborazione, graficamente rappresentata con un rettangolo,

- test, che indicano due o più direzioni di continuazione di un a seconda del verificarsi o meno di una condizione. E' rappresentato con un rombo a due uscite (vero/falso) all'interno del quale è scritta la condizione
- ingresso/uscita, che comportano l'immissione di informazioni dall'esterno oppure l'invio di informazioni verso l'esterno, rappresentati con un parallelogramma.

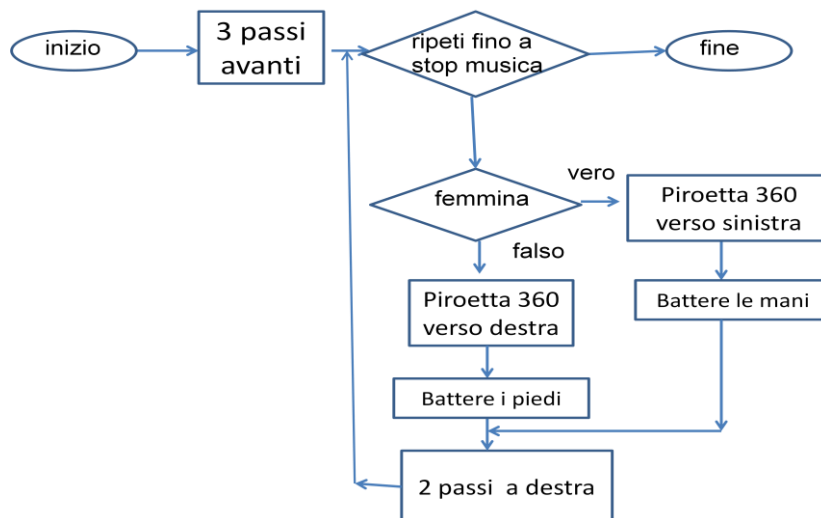
Il procedimento per trovare il massimo in una sequenza di (almeno due) numeri vista in precedenza potrebbe essere specificato con un diagramma di flusso come il seguente:



Discussione. Nella vita quotidiana abbiamo tanti esempi di processi più o meno complessi che ci troviamo a dover comunicare in modo preciso. Indichiamone qualcuno.

One of my favorite and easiest example is the table: “What to do in case the alarm sounds” present in every school. The order of the actions is important.

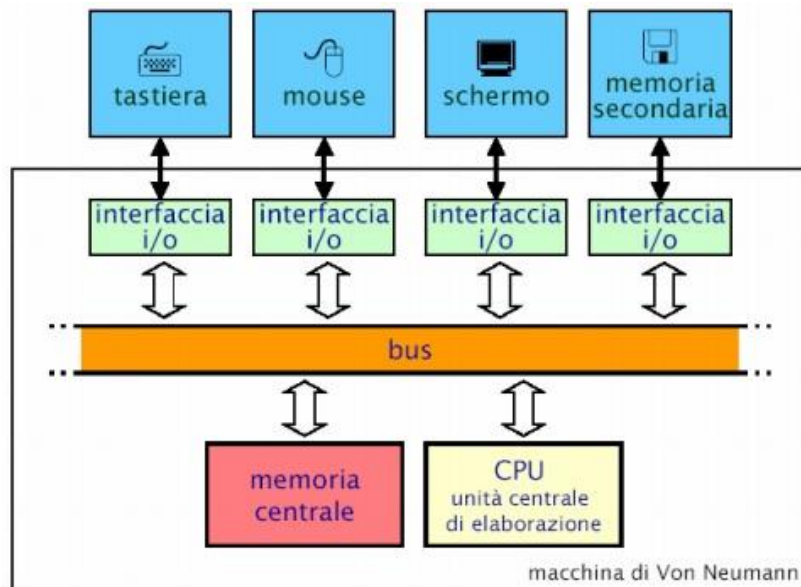
Other examples are dances: they can be very easy sequences of instructions. Specify a dance in a flow diagram



ATTIVITA'. Trascriviamo in Scratch. Esprimiamo in Scratch la soluzione al problema del massimo in una sequenza di numeri. Si veda il file Scratch MaxQuanti-Numeri.sb nella cartella MAterialiScratch.

5. Architettura di un pc

L'architettura di un calcolatore moderno è molto complessa, ma è basata su un modello molto semplificato detto **macchina di Von Neumann**:



La macchina di Von Neumann è composta da 4 componenti fondamentali:

1. **unità centrale di elaborazione (CPU – Central Processing Unit):** esegue istruzioni per l'elaborazione dei dati e svolge anche funzioni di controllo;
2. **memoria centrale (RAM - Random Access Memory):** memorizza e fornisce l'accesso a dati e programmi in esecuzione;
3. **interfacce di ingresso (input) e uscita (output):** componenti di collegamento con le periferiche del calcolatore;
4. **bus:** un canale che collega tutti i componenti fra loro.

Le periferiche sono le apparecchiature che consentono al calcolatore di scambiare informazioni con il mondo esterno.

Si dividono in:

- **periferiche di input:** usate per inserire dati di vario genere.
Esempi: tastiera, mouse, lettore cd, scanner, ecc
- **periferiche di output:** hanno lo scopo di presentare i dati elaborati all'utente.
Esempi: schermo, stampante, ecc.

In un calcolatore sono presenti due principali e differenti tipi di memoria, memoria centrale e memoria secondaria. La memoria centrale viene considerata parte integrante del modello di Von Neumann, mentre la **memoria secondaria** (ad esempio l'Hard Disk ed il lettore CD-ROM) viene considerata una periferica, anche se in genere è presente all'interno del calcolatore.

5.1 Un esempio concreto

I componenti della macchina di Von Neumann possono essere facilmente riconosciuti nei computer in vendita sui volantini. Ad esempio:



Leggiamo:

- **Processore** Intel Core i7-5500U: è la marca e la versione della CPU;
- **RAM** 8GB DDR3L: è la memoria centrale espressa in quantità di memoria (GigaByte - GB) e versione DDR3L ;
- **HD** 500GB: è la memoria secondaria (l'hard disk) utilizzata per la memorizzazione dei dati (in questo caso fino a 500 GigaByte);
- **schermo** 15.6": è una periferica di output e la sua dimensione è espressa in pollici (in diagonale);
- **webcam** HP TrueVision HD: è una periferica di input che riesce a registrare video anche in qualità alta definizione (HD)

Le altre specifiche del volantino si riferiscono ai dettagli su batteria, compatibilità con periferiche esterne, schede dedicate (ad esempio la scheda video) e periferiche di input/output come il masterizzatore DVD.

5.2 PC Desktop, Notebook, Tablet e Smartphone

PC Desktop, Notebook, Tablet, e Smartphone sono tutti basati sull'architettura di Von Neumann ma nascono da esigenze di mercato diverse. Si passa dal calcolatore in postazione fissa, il PC Desktop, al calcolatore miniaturizzato utilizzabile in mobilità, lo Smartphone.



PC Desktop

Il PC Desktop è una tipologia di personal computer e di computer fisso che si contraddistingue per avere dimensioni fisiche sufficientemente contenute da poter essere comodamente utilizzata su una scrivania, ma non in mobilità.



Vantaggi:

- Alte prestazioni
- Alta espandibilità (si possono aumentare le prestazioni e si può connettere qualsiasi tipo di periferica con l'interfaccia/adattatore giusto)
- Prezzi Bassi

Svantaggi:

- Dimensioni e peso
- Dipendenza dalla presa elettrica

Notebook

Un computer portatile (anche chiamato notebook) è un personal computer dotato di schermo, tastiera e alimentazione a batteria, tutto integrato nello stesso telaio e caratterizzato da dimensioni e peso ridotti in modo da permetterne un facile trasporto ed un uso in mobilità.



Vantaggi:

- Prestazioni sempre più simili a quelle dei PC Desktop
- Dimensione dello schermo (in genere 15") adatta a tutti i programmi di uso comune
- memoria secondaria (Hard Disk) con buona capacità
- tastiera comoda
- tante porte di connessione per periferiche esterne

Svantaggi:

- Dimensioni e peso ne limitano l'utilizzo in mobilità
- Autonomia della batteria spesso limitata (dipende dal modello)
- Rumorosità delle ventole (dipende dal modello)

Tablet

Il tablet PC (tavoletta) è un computer portatile che grazie alla presenza di uno o più digitalizzatori permette all'utente di interfacciarsi con il sistema direttamente sullo schermo mediante le dita (o pennini). Il tablet PC è di fatto un normale Personal Computer portatile con capacità di input superiori.



Vantaggi:

- Dimensioni e peso ridotti

- Facilità d'uso grazie a touchscreen e interfacce intuitive
- Nessuna rumorosità grazie all'assenza delle ventole
- Utilizzano processori (CPU) a basso consumo, di conseguenza presentano un'autonomia superiore a quella dei notebook. L'Apple iPad supera le otto ore.

Svantaggi:

- L'immissione del testo sulla tastiera del touchscreen può risultare piuttosto scomoda.
- La capienza della memoria secondaria è spesso troppo bassa
- Poche connessioni fisiche con periferiche esterne (spesso si utilizzano adattatori)
- Rispetto alle prestazioni e alla dotazione, i tablet pc sono spesso cari

Smartphone

Lo Smartphone è un telefono cellulare con capacità di calcolo, di memoria e di connessione dati molto più avanzate rispetto ai normali telefoni cellulari. E' basato su un sistema operativo per dispositivi mobili.



Vantaggi:

- Hanno gli stessi vantaggi dei tablet ma con dimensioni notevolmente ridotte per una mobilità massima
- calcolatore e telefono in un unico e piccolo dispositivo

Svantaggi:

- L'autonomia spesso è ridotta dall'utilizzo della rete telefonica
- Lo schermo troppo piccolo può rendere più scomoda l'interazione.

Parte 3 – Strumenti che favoriscono la creatività

In questo percorso nel primo capitolo si è voluto avviare il lettore a usare Scratch come ambiente per creare attività semplici mentre nel secondo si sono introdotti argomenti che consentono di progettare e realizzare attività più complesse attraverso l'uso di più componenti di programmazione.

Crediamo infatti che primo obiettivo della presenza di strumenti di sviluppo software nella scuola primaria sia la promozione della creatività. Nella scuola secondaria di primo grado all'obiettivo creatività si associa quello dell'acquisizione dei concetti fondamentali della programmazione per poter creare attività multidisciplinari. In ogni caso per entrambi i livelli di istruzione e quindi per il primo ciclo in genere stimolare la creatività è obiettivo primario.

1 Funzioni caratteristiche

Per questo torniamo a leggere sia Shneiderman 2002 sia Romeike 2007: il primo perché mette in evidenza le proprietà che devono soddisfare (quali funzionalità dobbiamo trovare nei) gli ambienti di sviluppo programmi per promuovere la creatività, il secondo perché ha ripreso le osservazioni del primo calandole nella realtà degli ambienti di sviluppo più noti e più usati ai nostri giorni. Per la verità entrambi parlano di IT e non di ambienti di sviluppo ma se per il primo possiamo dire strumenti digitali per il secondo, dagli esempi e dalle affermazioni fatte, si evince che pensa soprattutto agli strumenti di sviluppo sw.

Shneiderman scrive che per favorire la creatività gli strumenti digitali devono offrire funzionalità attraverso le quali l'utente:

- sia incoraggiato a esplorare e sperimentare e naturalmente possa farlo senza problemi (pain-free)
- possa avere feedback immediato e utile per quanto sta facendo
- possa sbagliare senza grandi penalizzazioni e essere gratificato quando fa qualcosa di buono
- sia facilitato nel fare e disfare
- abbia modo di visualizzare quello che sta facendo anche in corso d'opera
- sia incoraggiato a cercare di estendere quello che sa e cercare ispirazione per fare cose nuove
- possa comporre qualcosa di nuovo per quanto più possibile passo-passo
- sia facilitato nel diffondere quello che ha realizzato per avere valorizzato il suo lavoro.

Romeike riprende Shneiderman e porta esempi di ambienti di sviluppo sw che offrono agli utenti le funzionalità indicate da Shneiderman come proprietà caratteristiche di uno strumento digitale che favorisce la creatività. Per ognuna delle proprietà caratteristiche viene citato anche Scratch, oggetto di questo nostro documento. Col permesso dell'autore riporto qui per intero il paragrafo "5.1 Support for creative practice" in "Three drivers for creativity in Computer Science Education" (in attesa di tradurlo).

"Computer based experimentation and exploration can inspire new ideas and should involve design, simulation, surprise and creation (Mitchell et al., 2003). Programmers utilize the possibility of experimentation allowed by the programming environments by testing out ideas and possible paths to solutions. Using selective trial and error is an essential part of software development (Glass, 2006).

Those “what if” scenarios are explored quickly and can easily be changed. This is only possible due to the support of the environment. Scratch, as other programming environments for beginners, makes exploration even more accessible; operations can easily be executed on the objects and the results can be observed in real-time. Thus also learning to program involves the exploration of the programming environment and language (“What are my possibilities?”) and the experimentation with the objects and constructs they offer (“What happens if I apply them like this?”). The immediate and useful feedback for one’s actions given by the compiler or interpreter supports the experimentation and is quite unique to software development – in other fields the tools to work with either do not provide qualitative feedback automatically or do not point out problems. It can be observed with many students that the action-related feedback promotes a sense of control and with it self-regulated learning. As Shneiderman requires no big penalties from creativity supporting software, the biggest penalty of a SDE is the error-feedback itself. As students in other fields have the teachers or class-mates as only source of feedback, when learning to program this is done by the machine in a not intimidating way. The functioning program will be rewarding for success, e.g. in Scratch this will be a nice animation or game. Supporting for experimentation and consideration of feedback is the certitude that nothing can “break” easily. Scratch offers an easy way to undo and redo the last changes. While the aforementioned points of creative work in other fields only are possible in simulations, in programming the experimentation is done at the product itself. Environments for learning how to program also meet others of Shneiderman’s criteria, e.g. visualizing data and processes, which helps to organize ideas and facts and to discover relationships. The trend to visual programming reflects that creativity-relevant aspect. In Scratch programming constructs (such as loops, operations, broadcasts) are represented as colorful building-blocks, separating the kinds of constructs in different colors. In design oriented approaches data structures and processes are modeled and viewed as diagrams with their relationships and may be automatically transformed into code and vice versa (e.g. with Fujaba). The trend to such a visual, more abstract, level of programming suggests that it will become even more emphasized in the future and thus support creativity better. It is important for a creativity supporting environment to make knowledge easy accessible and provide inspiration. Programming environments allow searching in a help knowledge database and provide a documentation of comments and examples. Scratch additionally offers inspiring example-programs made by other children and provides cards that explain constructs to be used and initiates experimentation with those constructs. Composing a work step-by-step allows to slowly approach a problem by leaving plenty of possibilities open as not all important decisions are made at the beginning as with the topdown design. While top-down design is more emphasized in professional software development, the process of learning to program may be different (Kaasbøll, 1998) and involves a bottom-up step by step approach. It allows to review the work done and to evaluate it in the light of what shall be done. This is important to creative practice, since often the outcome is not set clearly from the beginning. In Scratch this matter is implemented by making it possible to run and review a program at any point of the process. Disseminating the results to gain recognition is important for peer-recognition and for motivation of creative work. As the classroom only offers limited possibilities to present one’s achievements, the internet allows to present them to a wide audience. Scratch has a built-in function to upload a work to an internet server where the programs can be shared and evaluated by other learners and also serve for inspiration to others. Hence Scratch, as other environments for learning to program do, fulfills Shneiderman’s tasks for creativity supporting software“

Finiamo questa lunga citazione di Romeike e del suo lavoro con una ulteriore ma ultima citazione anch’ essa molto importante, dal paragrafo 5.2 “I concetti di computer science sono la base per una pratica creativa con l’ uso della IT” nel lavoro citato: “Se opportuni sistemi sw sono necessari per

favorire la creatività in computer science, conoscere i concetti basilari della computer science è indispensabile per concepire usi creativi della IT in altri ambienti. ... tra i concetti basilari è incluso il pensiero algoritmico ... Conoscere concetti base della computer science permette agli studenti di essere creativi in altre discipline e ambienti”.

2 L’ambiente di sviluppo Scratch è tra questi strumenti

Come abbiamo già scritto Scratch compare per ciascuna funzionalità negli esempi che Romeike fa su quali sw offrano le funzionalità che Shneidermann considera indispensabili per favorire la creatività.

Questo significa che la conoscenza di Scratch, o se vogliamo anche soltanto l’averlo usato qualche volta, fornisce elementi di confronto nell’analisi di come altri sw si comportino riguardo ciascuna caratteristica. Per questo pensiamo che l’insegnante debba puntare ed insistere su questi aspetti più che su quelli di avvio alla programmazione in sé quando propone Scratch ai suoi allievi ma soprattutto quando lo propone ai colleghi durante una iniziativa di aggiornamento. In questo senso Scratch è modello di ambiente di sviluppo che favorisce la creatività degli utenti.

Qui di seguito riprendiamo ciascuna caratteristica puntualizzando dove è stata oggetto di osservazione in queste note magari aggiungendo un esempio o una nota dove necessario ribadire l’attenzione.

L’utente va incoraggiato a esplorare e sperimentare e naturalmente deve poterlo fare senza problemi (pain-free). La funzione di “aiuto” o help disponibile pigiando col tasto destro del mouse è un esempio e così la eseguibilità di un unico comando. Lo stesso ambiente favorisce la sperimentazione essendo molto ricco di strumenti interni (registrazione di suoni, file dove sono già disponibili immagini, suoni per creare le prime esperienze)

L’utente deve poter avere feedback immediato e utile per quanto sta facendo La funzione di esecuzione di uno script nella forma in cui si trova ad un certo momento va in questo senso. La possibilità di estrarre parte di uno script ed eseguirlo

L’utente deve poter sbagliare senza grandi penalizzazioni e essere gratificato quando fa qualcosa di buono La gratificazione dell’utente Scratch è la storia che si svolge sulla scena durante l’ esecuzione del programma.

L’utente deve essere facilitato nel fare e disfare. Scratch non permette errori sintattici rendendo disponibili o indisponibili insiemi di istruzioni diverse a seconda che si siano create variabili e quali oppure no ad esempio.

L’utente deve aver modo di visualizzare quello che sta facendo anche in corso d’opera Contrariamente ad altri ambienti non bisogna arrivare ad avere un certo pattern di codifica per eseguire un programma, ad ogni nuova istruzione si può decidere di eseguire da un punto ad un altro, basta estrarre se il caso la porzione di istruzioni da provare. Si può decidere con una spunta di vedere o non il contenuto di una variabile o renderlo non visibile, e così via.

L’utente deve essere incoraggiato a cercare di estendere quello che sa e cercare ispirazione per fare cose nuove La metodologia tipica degli ambienti Scratch è: cerca, prova, modifica, crea. L’impostazione globale dell’ ambiente Scratch incoraggia ad estendere un comportamento guardingo: ad esempio un utente alle prime esperienze sceglie per i suoi script immagini dai file scaricati al momento del download ma dopo un po’ vedendo che può scegliere tra alternative possibili gli viene spontaneo esplorare quelle.

L'utente deve poter comporre qualcosa di nuovo per quanto più possibile passo-passo.

L'utente deve essere facilitato nel diffondere quello che ha realizzato per avere valorizzato il suo lavoro. Le comunità utenti e docenti sono caratteristiche di Scratch e molto frequentate.

3.CONCLUSIONI

Quanto si è osservato finora ci porta a concludere che Scratch ha un valore non tanto come linguaggio (su cui in realtà potremmo avanzare anche qualche critica) ma come ambiente che favorisce gli utenti nell'avvicinarsi a capire come progettare oggetti sw e ad effettivamente crearli. Lo stesso ambiente diventa poco pratico quando i programmi diventano lunghi, quando si vorrebbe avere disponibili strutture dati oltre le variabili e le liste, ecc. Insomma diventa insoddisfacente quando l'utente non è più da avviare al progetto e realizzazione di programmi: per quanto riguarda la programmazione l'avvio è l'obiettivo dichiarato degli autori di Scratch (e nostro nel proporre l'uso). Arrivati a questo punto gli utenti che vogliono continuare ad approfondire algoritmi, programmazione e conoscenze degli ambienti digitali in genere passeranno ad altri strumenti.

A termine di questo lavoro citiamo ancora una volta Ralf Romeike dal paragrafo 5.2 dal lavoro Romeike 2007a "I concetti di computer science sono la base per una pratica creativa con l'uso della IT" del lavoro già ampiamente citato: *"Se opportuni sistemi sw sono necessari per favorire la creatività in computer science, conoscere i concetti basilari della computer science è indispensabile per concepire usi creativi della IT in altri ambienti. ... tra i concetti basilari è incluso il pensiero algoritmico ... Conoscere concetti base della computer science permette agli studenti di essere creativi in altre discipline e ambienti"*.

Bibliografia

- Fasko, D. (2000) *Education and creativity*. Creativity Research Journal, 13(3-4), p. 317-327. Routledge, Philadelphia.
- Greene, S. L. (2002) *Characteristics of applications that support creativity*. Communications of the ACM, Vol. 45, No. 10 (Oct. 2002), Pages 100-104.
- Loveless, A., M. (2002) *Literature review in creativity, new technologies and learning*. Futurelab series, Report 4, School of education, University of Brighton. ISBN 0-9544695-4-2. On-line: <http://archive.futurelab.org.uk/resources/documents/lit_reviews/Creativity_Review.pdf>
- Qualifications and Curriculum Authority (QCA, 1999): *How can teachers promote creativity?* On-line: <<http://webarchive.nationalarchives.gov.uk/20100823130703/http://curriculum.qcda.gov.uk/key-stages-1-and-2/learning-across-the-curriculum/creativity/howcanteacherspromotecreativity/index.aspx>>
- Romeike, R. (2007a) - A Creative Introduction to Programming (About flying Elephants, Dogs, Cats and Ideas!) <http://www.cs.uni-potsdam.de/~romeike/UEWettbewerb/index-english.htm>
- Romeike, R. (2007b): Three drivers for creativity in computer science education. In: Benzie, D.; Iding, M. (eds.): Proceedings of IFIP-Conference on "Informatics, Mathematics and ICT: A golden triangle", June 27-29, 2007, Boston, USA.
- Schneiderman, B. (2002) *Creativity support tools*. Communications of the ACM, Vol. 45, Issue 10 (Oct. 2002), p. 116-120.