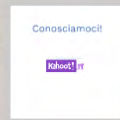


UF 28 - Coding e pensiero computazionale



1

LESSON PROFILE

2

COMPUTATIONAL
THINKING

3

CODING

Learn to code
code to learn

4

NORMATIVA

5

CONCETTI E
TECNICHE

6

RISORSE

7

ASSESSMENT

8

CODING -
EDUCATION

Doc. Silvia Mazzeo
(Iniziativa formativa ID.33586)
Scuola Polo IPSEOA PAOLA - CSRH07000Q

Conosciamoci!

<https://create.kahoot.it/share/uf28-odding-k-percorsi-computazionale/9a7b5f6-9ab1-4869-a75a-9adba1f2af01>



<https://create.kahoot.it/share/uf-28-coding-e-pensiero-computazionale/9a016bf6-3eb1-4968-a759-95dba4f2af01>

The Kahoot! logo features the word "Kahoot!" in a large, white, bold, sans-serif font. The exclamation mark is stylized with a small dot. To the right of the word, the ".IT" domain is written in a smaller, white, stylized font. The entire logo is set against a solid purple rectangular background.

Kahoot!.IT

Dai principi teorici ad una ipotesi di lavoro

Promuovere l'innovazione didattica allo scopo di favorire lo sviluppo del pensiero computazionale e il raggiungimento delle competenze chiave digitali e di cittadinanza attiva e delle competenze trasversali, cognitive, meta cognitive, operative e relazionali, di ogni studente

**PENSIERO
COMPUTAZIONALE**

APPLICAZIONE

CODING

PENSIERO COMPUTAZIONALE

- 1** Cosa s'intende
- 2** Riferimenti e documenti
- 3** Perché introdurlo nella scuola

CODING

1

COSA S'INTENDE

2

QUALI PERCORSI

3

QUALI STRUMENTI

APPLICAZIONE LABORATORIALE

1

Interdisciplinarietà

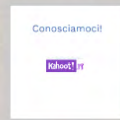
2

Dal curriculum all'ora di lezione

3

Unità d'Apprendimento

UF 28 - Coding e pensiero computazionale



1

LESSON PROFILE

2

COMPUTATIONAL
THINKING

3

CODING

Learn to code
code to learn

4

NORMATIVA

5

CONCETTI E
TECNICHE

6

RISORSE

7

ASSESSMENT

8

CODING -
EDUCATION

Doc. Silvia Mazzeo
(Iniziativa formativa ID.33586)
Scuola Polo IPSEOA PAOLA - CSRH07000Q

Computational thinking

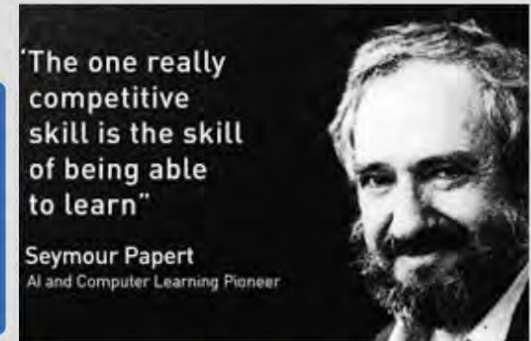
Una questione semantica e di sostanza

Alcune definizioni

**Scomposizione
Schematizzazione
Progettazione**

Seymour Papert

Ha lavorato con Piaget. Inventore del linguaggio LOGO



Teoria del costruzionismo, come evoluzione del costruttivismo. ('60-'70)

- Il costruttivismo afferma che l'individuo che apprende costruisce modelli mentali per comprendere il mondo intorno a lui
- Il costruzionismo sostiene che l'apprendimento avviene in modo più efficiente se questi modelli mentali vengono riflessi nella produzione di "oggetti tangibili"

"Costruzionismo indica due aspetti della teoria della didattica delle scienze. Dalle teorie costruttiviste in psicologia prendiamo la visione dell'apprendimento come una ricostruzione piuttosto che come trasmissione di conoscenze. Successivamente estendiamo il concetto dei materiali manipolativi nell'idea che l'apprendimento è più efficiente quando è parte di un'attività come la costruzione di un prodotto significativo"

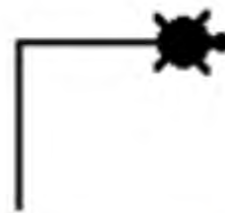
Seymour Papert. Constructionism: A New Opportunity for Elementary Science Education. Proposal to NSF, 1997



`forward 50`



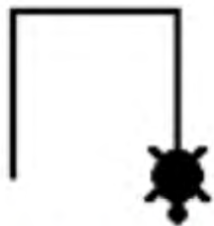
`right 90`



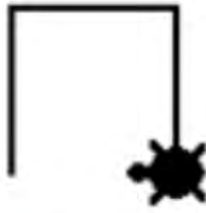
`forward 50`



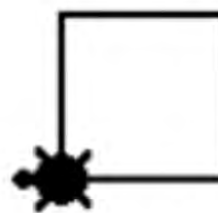
`right 90`



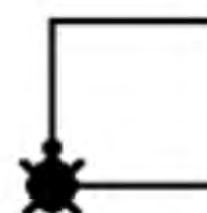
`forward 50`



`right 90`



`forward 50`



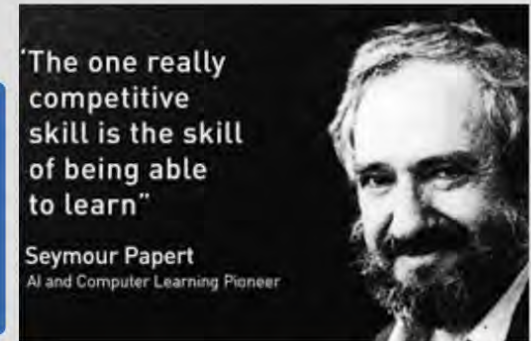
`right 90`

© 2000 Logo Foundation

[https://it.wikipedia.org/wiki/Logo_\(informatica\)](https://it.wikipedia.org/wiki/Logo_(informatica))

Seymour Papert

Ha lavorato con Piaget. Inventore del linguaggio LOGO



Teoria del costruzionismo, come evoluzione del costruttivismo. ('60-'70)

- Il costruttivismo afferma che l'individuo che apprende costruisce modelli mentali per comprendere il mondo intorno a lui
- Il costruzionismo sostiene che l'apprendimento avviene in modo più efficiente se questi modelli mentali vengono riflessi nella produzione di "oggetti tangibili"

"Costruzionismo indica due aspetti della teoria della didattica delle scienze. Dalle teorie costruttiviste in psicologia prendiamo la visione dell'apprendimento come una ricostruzione piuttosto che come trasmissione di conoscenze. Successivamente estendiamo il concetto dei materiali manipolativi nell'idea che l'apprendimento è più efficiente quando è parte di un'attività come la costruzione di un prodotto significativo"

Seymour Papert. Constructionism: A New Opportunity for Elementary Science Education. Proposal to NSF, 1997

Jeanette Wing

Direttrice degli avansiani del Data Science Institute presso la Columbia University, dove è anche professore di informatica.



«Il pensiero computazionale rappresenta un **atteggiamento** ed un **complesso di abilità** che sono universalmente applicabili e che chiunque, non soltanto gli informatici, dovrebbe essere desideroso di apprendere e di utilizzare».

Computational Thinking, Communications of the ACM, 49(3), 33-35, 2006

«il pensiero computazionale è il **processo di pensiero** messo in gioco nel formulare un problema e nell'esprimere le sue soluzioni in modo tale che un agente computazionale – uomo o macchina – possa efficacemente portare a termine il processo risolutivo».

Computational Thinking's Influence on Research and Education for All. International Journal of Educational Technology, 25(2), 2017

Simon Peyton Jones

Scienziato informatico britannico che ricerca l'implementazione e le applicazioni dei linguaggi di programmazione funzionale, in particolare la programmazione funzionale pigra-
National Centre for Computing Education



Il pensiero computazionale è la capacità di:

- **sviluppare astrazioni computazionali** di problemi del mondo reale;
- progettare, sviluppare, perfezionare, ragionare di artefatti computazionali (programmi).

Il computing curriculum inglese dice che anche i bambini della primaria dovrebbero essere in grado di "utilizzare il **ragionamento logico per prevedere il comportamento di semplici programmi**".

Cioè, essere in grado di:

- **spiegare** a qualcun altro che cosa un programma dovrebbe fare;
- capire **perché** un programma non si comporta come previsto.

Prevedere è importante. Programmare non è solo scrivere, ma anche essere in grado di eseguire mentalmente ciò che si è scritto. Questo è pensiero computazionale.

(Computing At School)

Mitch Resnick

Professore di Learning Research, direttore del Centro Okawa e direttore del gruppo Lifelong Kindergarten presso il **M**assachusetts **I**nstitute of **T**echnology Media Lab



Il pensiero computazionale è un modo di esprimersi e comprendere il mondo utilizzando computer e idee computazionali.

Possiamo scomporre il pensiero computazionale in tre categorie:

concetti, pratiche e prospettive.

- I **concetti** sono quelli che emergono dall'uso e dalla comprensione dei computer e dell'informatica, ad es. iterazione e variabili.
- Le **pratiche** sono strategie utili per progettare e risolvere problemi (ad esempio, sperimentare e iterare, "debugging", "remixare" e riutilizzare).
- Nel creare cose con i computer gli studenti cominciano a vedere se stessi e il mondo in una **nuova prospettiva**. Si rendono conto che possono essere "creatori" e da questa nuova prospettiva ripensare a come le cose nel mondo sono fatte.

(MIT Media Lab)

Leo Pahkin

Counsellor of Education at Finland's National Agency for Education



Pensiero algoritmico.

Il pensiero algoritmico non deve necessariamente includere l'utilizzo di un computer. Non è qualcosa che è stato inventato con il computer, ma è sempre esistito nella storia umana. Ad esempio, la somma di due numeri è anche una sorta di algoritmo.

Utilizzando le fasi del pensiero algoritmico, gli studenti possono risolvere problemi usando strumenti semplici come i fogli di calcolo.

Il termine copre l'idea che è necessario utilizzare una **serie di procedure logiche** strutturate per trovare una soluzione a un problema.

Gli algoritmi, di solito, aiutano a trovare la soluzione in modo più veloce o più semplice.

(Finnish National Board of Education)

Judith Gal-Ezer

Scienziata informatica israeliana e educatrice informatica nota per il suo sviluppo del curriculum di informatica delle scuole superiori in Israele. È una professoressa emerita presso l'Open University of Israel.



Il pensiero algoritmico è lo spirito e l'arte dell'informatica, (Donald Knuth - David Harel).

Significa guardare ai problemi attraverso **molte livelli di astrazione**. Cerchiamo di dividere un problema in moduli più piccoli, ognuno dei quali è di per sé un problema algoritmico. Una volta trovato un algoritmo, vogliamo sapere e possibilmente dimostrare che l'algoritmo è corretto, si blocca ad un certo punto, fornisce la risposta corretta ed è il più efficace possibile.

In generale, dopo questi passaggi, traduciamo l'algoritmo in un linguaggio di programmazione, infine, il programma viene eseguito su un computer.

Ci sono altre forme di pensiero che gli studenti incontrano a scuola, ad esempio **il pensiero matematico**; i matematici sono felici di dimostrare che un problema ha una soluzione, sono ancora più soddisfatti di scoprire che la soluzione è unica. All'altro estremo, c'è la modalità di **pensiero ingegneristico** in cui si pone l'accento sul collocare i dati nelle formule giuste per ottenere un risultato. Gli informatici vogliono andare oltre, procedere attraverso tutti i livelli di astrazione e infine calcolare la soluzione.

Alessandro Bogliolo

Professore associato ING-INF/05 presso DiSPeA Università degli Studi di Urbino Carlo Bo ; Ambasciatore per l'Italia della Eu Code Week



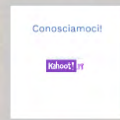
Il pensiero computazionale è la capacità di individuare un procedimento costruttivo, fatto di passi semplici e non ambigui, che ci porta alla soluzione di un problema complesso.

Nova - Il Sole 24 ore - https://nova.ilsole24ore.com/progetti/il-senso-profondo-del-pensiero-computazionale/?refresh_ce=1

Pensiero computazionale - approccio ai problemi e alla loro soluzione

-  SCOMPOSIZIONE del problema in piccole parti e quesiti
conoscenza profonda del problema
-  SCHEMATIZZAZIONE, analisi dei dati per organizzarli
gerarchicamente individuazione della strategia risolutiva
-  PROGETTAZIONE di soluzioni procedendo per step sequenziali
la formulazione dettagliata delle procedure che portano alla
risoluzione

UF 28 - Coding e pensiero computazionale



1

LESSON PROFILE

2

COMPUTATIONAL
THINKING

3

CODING

Learn to code
code to learn

4

NORMATIVA

5

CONCETTI E
TECNICHE

6

RISORSE

7

ASSESSMENT

8

CODING -
EDUCATION

Doc. Silvia Mazzeo
(Iniziativa formativa ID.33586)
Scuola Polo IPSEOA PAOLA - CSRH07000Q

CODING

Strumenti e metodi di programmazione visuale a blocchi o testuali per favorire lo sviluppo del pensiero computazionale.

Alcune definizioni

**Scomposizione
Schematizzazione
Progettazione**

... è l'operazione di programmazione informatica;
ovvero l'insieme delle attività e tecniche di scrittura di istruzioni da far eseguire ad un computer o ad un robot, utilizzando un determinato
linguaggio di programmazione (codice).

M. Giordano - C. Moscetti

Coding e pensiero computazionale
nella Scuola Primaria,
ELI -La Spiga Ed., 2016

Il pensiero computazionale ha...
una forte componente legata al
coding

Il suo obiettivo è quello di
descrivere procedimenti effettivi
per la risoluzione di problemi da
parte del computer... può essere
fatto soprattutto con la
programmazione

Stefano Penge, Wikibook

Dietro il coding

[https://it.wikibooks.org/wiki/
Dietro_il_coding/Che_cos%27è_il_coding](https://it.wikibooks.org/wiki/Dietro_il_coding/Che_cos%27è_il_coding)

coding letteralmente è “l'attività
di scrivere codice sorgente”

Programmare può significare anche
analizzare, progettare, verificare,
integrare un codice sorgente,
mentre coding fa riferimento solo
alla scrittura del codice.

Enrico Nardelli

Professore ordinario di Informatica, sono dal 2002 al Dipartimento di Matematica di "Tor Vergata"
Coordinatore (con Giorgio Ventre (Università di Napoli "Federico II") il progetto **Programma il Futuro** attuato dal CINI, Consorzio Interuniversitario Nazionale di Informatica, per conto del Ministero dell'Istruzione, Università e Ricerca
Membro del Consiglio Direttivo del CINI, e membro del Comitato Direttivo di ACM Europe, la branca europea dell'Association for Computing Machinery
<https://www.ilfattoquotidiano.it/blog/enardelli/>



Per i non addetti ai lavori chiarisco che con la parola “coding” si intende, in informatica, la stesura di un programma, cioè di una di quelle sequenze di istruzioni che, eseguite da un calcolatore, danno vita alla maggior parte delle meraviglie digitali che usiamo quotidianamente.

Capisco che questo termine generi meno confusione del corretto ed usatissimo equivalente italiano “programmazione”, ma spendo solo questo inciso per invitare i nostri cervelli a non essere pigri: usare la nostra lingua significa sostenere la nostra identità sociale e culturale...

<https://www.ilfattoquotidiano.it/2014/08/24/informatica-dal-coding-al-pensiero-computazionale/1097593/>

Alessandro Bogliolo

Professore associato ING-INF/05 presso DiSPeA Università degli Studi di Urbino Carlo Bo ; Ambasciatore per l'Italia della Eu Code Week



Il coding è uno dei termini inglesi utilizzati nel linguaggio internazionale per descrivere la programmazione.

Andando oltre la traduzione letterale, il termine assume un significato più ampio della semplice "codifica" e fa riferimento a un'attività che comprende non solo la **scrittura di codice**, ma anche l'individuazione di una **soluzione algoritmica al problema da affrontare**.

La settimana europea della programmazione - <http://www.indire.it/content/index.php?action=read&id=1859&graduatorie=0>

Il linguaggio delle cose

<https://youtu.be/AEXF33EgH0w>

CODING

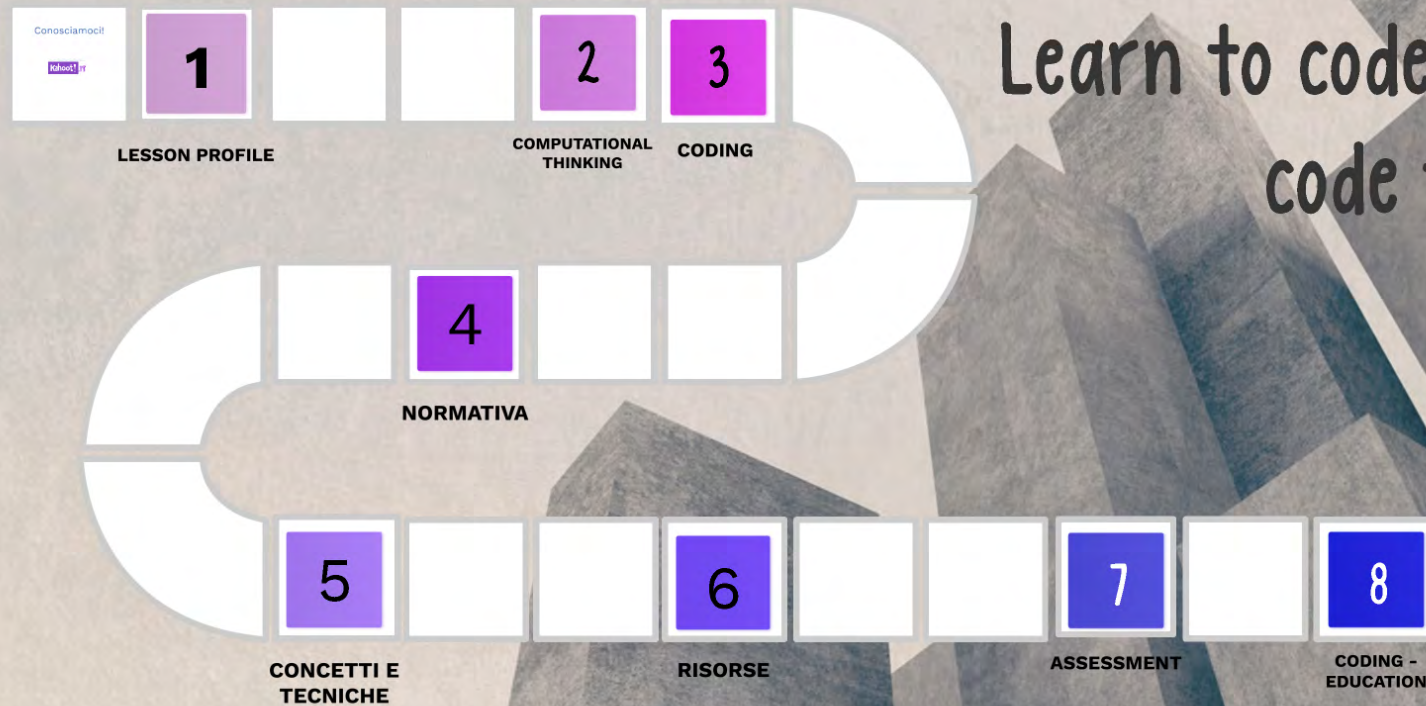
approccio ai problemi e alla loro soluzione

- SCOMPOSIZIONE del problema in piccole parti e quesiti conoscenza profonda del problema
- SCHEMATIZZAZIONE, analisi dei dati per organizzarli gerarchicamente individuazione della strategia risolutiva
- PROGETTAZIONE di soluzioni procedendo per step sequenziali la formulazione dettagliata delle procedure che portano alla risoluzione

Il punto di forza del coding è la rappresentazione del pensiero computazionale.

Il coding consente di creare e visualizzare schemi di pensiero che possano essere analizzati, perfezionati e poi riutilizzati applicandoli a diversi argomenti. Il coding così inteso è uno **strumento utile all'analisi metacognitiva** perchè offre ad alunni e studenti una **rappresentazione visiva** dei processi sottesi ai loro ragionamenti.

UF 28 - Coding e pensiero computazionale



Learn to code
code to learn

Doc. Silvia Mazzeo
(Iniziativa formativa ID.33586)
Scuola Polo IPSEOA PAOLA - CSRH07000Q

Riferimenti Europei e Nazionali

“Oggi, il 90% di tutti i posti di lavoro richiedono almeno un livello base di competenze nelle TIC e queste ora includono il **coding**”

Marianne Thyssen, la commissaria europea responsabile per l'Occupazione, gli affari sociali, competenze e la mobilità del lavoro (Jacobsen, 2015).



Competenze chiave

4 - competenza digitale

La competenza digitale consiste nel saper utilizzare con **dimestichezza e spirito critico** le tecnologie della società dell'informazione (TSI) per il lavoro, il tempo libero e la comunicazione.

Essa è **supportata** da abilità di base nelle TIC: l'uso del computer per reperire, valutare, conservare, produrre, presentare e scambiare informazioni nonché per comunicare e partecipare a reti collaborative tramite Internet.

<https://eur-lex.europa.eu/legal-content/IT/TXT/PDF/?uri=CELEX:32006H0962&from=IT>

European Commission 2015

Relazione congiunta del Consiglio e della Commissione sull'attuazione del quadro strategico per la cooperazione europea nel settore dell'istruzione e della formazione (ET 2020) — Nuove priorità per la cooperazione europea nel settore dell'istruzione e della formazione. Gazzetta Ufficiale dell'Unione Europea.

«l'acquisizione di competenze digitali, incluso il coding, è considerata essenziale per sostenere lo sviluppo e la competitività».

European Commission 2016



A new skills agenda for Europe. Working together to strengthen human capital, employability and competitiveness. Gazzetta Ufficiale dell'Unione Europea - European Commission. (2016).

Strengthening the foundation: basic skills

Europe faces a basic skills challenge. People need a minimum level of basic skills, including numeracy, literacy and basic digital skills, to access good jobs and participate fully in society. These are also the building blocks for further learning and career development. Around a quarter of the European adult population struggles with reading and writing, and has poor numeracy and digital skills. More than 65 million people in the EU have not achieved a qualification corresponding to upper secondary level. This rate varies significantly across EU countries, reaching 50% or more in some.

Si sottolinea la necessità di sviluppare competenze digitali per il mercato del lavoro

2.1 . innalzare il livello di padronanza delle competenze di base (alfabetiche, matematiche e digitali) e sostenere lo sviluppo della capacità di imparare a imparare quale presupposto costantemente migliore per apprendere e partecipare alla società in una prospettiva di apprendimento permanente;
...

2.3. promuovere l'acquisizione di competenze in scienza, tecnologia, ingegneria e matematica (STEM), tenendo conto dei collegamenti con le arti, la creatività e l'innovazione, e motivare di più i giovani, soprattutto ragazze e giovani donne, a intraprendere carriere STEM;

2.4. innalzare e migliorare il livello delle competenze digitali in tutte le fasi dell'istruzione e della formazione per tutti i segmenti della popolazione;

2.5. incoraggiare la competenza imprenditoriale, la creatività e lo spirito di iniziativa in particolare tra i giovani, ad esempio favorendo le occasioni in cui i giovani possano fare almeno un'esperienza imprenditoriale pratica durante l'istruzione scolastica;

[https://eur-lex.europa.eu/legal-content/IT/TXT/PDF/?uri=CELEX:32018H0604\(01\)&from=EN](https://eur-lex.europa.eu/legal-content/IT/TXT/PDF/?uri=CELEX:32018H0604(01)&from=EN)

2. sostenere lo sviluppo delle competenze chiave prestando particolare attenzione a quanto segue:

Competenz
a digitale

2.1 . innalzare il livello di padronanza delle competenze di base (alfabetiche, matematiche e digitali) e sostenere lo sviluppo della capacità di imparare a imparare quale presupposto costantemente migliore per apprendere e partecipare alla società in una prospettiva di apprendimento permanente;

...

2.3. promuovere l'acquisizione di competenze in scienza, tecnologia, ingegneria e matematica (STEM), tenendo conto dei collegamenti con le arti, la creatività e l'innovazione, e motivare di più i giovani, soprattutto ragazze e giovani donne, a intraprendere carriere STEM;

2.4. innalzare e migliorare il livello delle competenze digitali in tutte le fasi dell'istruzione e della formazione per tutti i

2.4. innalzare e migliorare il livello delle competenze digitali in tutte le fasi dell'istruzione e della formazione per tutti i segmenti della popolazione;

2.5. incoraggiare la competenza imprenditoriale, la creatività e lo spirito di iniziativa in particolare tra i giovani, ad esempio favorendo le occasioni in cui i giovani possano fare almeno un'esperienza imprenditoriale pratica durante l'istruzione scolastica;

[https://eur-lex.europa.eu/legal-content/IT/
TXT/PDF/?
uri=CELEX:32018H0604\(01\)&from=EN](https://eur-lex.europa.eu/legal-content/IT/TXT/PDF/?uri=CELEX:32018H0604(01)&from=EN)

2.1 . innalzare il livello di padronanza delle competenze di base (alfabetiche, matematiche e digitali) e sostenere lo sviluppo della capacità di imparare a imparare quale presupposto costantemente migliore per apprendere e partecipare alla società in una prospettiva di apprendimento permanente;
...

2.3. promuovere l'acquisizione di competenze in scienza, tecnologia, ingegneria e matematica (STEM), tenendo conto dei collegamenti con le arti, la creatività e l'innovazione, e motivare di più i giovani, soprattutto ragazze e giovani donne, a intraprendere carriere STEM;

2.4. innalzare e migliorare il livello delle competenze digitali in tutte le fasi dell'istruzione e della formazione per tutti i segmenti della popolazione;

2.5. incoraggiare la competenza imprenditoriale, la creatività e lo spirito di iniziativa in particolare tra i giovani, ad esempio favorendo le occasioni in cui i giovani possano fare almeno un'esperienza imprenditoriale pratica durante l'istruzione scolastica;

[https://eur-lex.europa.eu/legal-content/IT/TXT/PDF/?uri=CELEX:32018H0604\(01\)&from=EN](https://eur-lex.europa.eu/legal-content/IT/TXT/PDF/?uri=CELEX:32018H0604(01)&from=EN)

2. sostenere lo sviluppo delle competenze chiave prestando particolare attenzione a quanto segue:

Competenz
a digitale

Competenza digitale

La competenza digitale presuppone **l'interesse** per le tecnologie digitali e il loro utilizzo con dimestichezza e **spirito critico e responsabile** per apprendere, lavorare e partecipare alla società. Essa comprende l'alfabetizzazione informatica e digitale, la comunicazione e la collaborazione, l'alfabetizzazione mediatica, la creazione di contenuti digitali (inclusa la **programmazione**), la sicurezza (compreso l'essere a proprio agio nel mondo digitale e possedere competenze relative alla cbersicurezza), le questioni legate alla proprietà intellettuale, la risoluzione di problemi e il pensiero critico.

Per un nuovo
umanesimo

The diagram consists of three overlapping rectangular boxes. A large purple box on the left contains the text 'Indicazioni nazionali per il curriculum della scuola dell'infanzia e del primo ciclo d'istruzione' and the year '2012'. To its right, a smaller orange box contains the text 'Per un nuovo umanesimo'. Below the orange box, another orange box contains the text 'Profilo dello studente'. The background on the right is a blurred image of a person's face.

Indicazioni
nazionali
per il curriculum
della scuola
dell'infanzia
e del primo ciclo
d'istruzione

2012

Profilo dello
studente

Per un nuovo umanesimo

promuovere i saperi propri di un nuovo umanesimo: la capacità di cogliere gli aspetti essenziali dei problemi; la capacità di comprendere le implicazioni, per la condizione umana, degli inediti sviluppi delle scienze e delle tecnologie; la capacità di valutare i limiti e le possibilità delle conoscenze; la capacità di vivere e di agire in un mondo in continuo cambiamento

Profilo dello studente

Le sue conoscenze matematiche e scientifico-tecnologiche gli consentono di **analizzare dati e fatti della realtà** e di verificare l'attendibilità delle analisi quantitative e statistiche proposte da altri. Il possesso di un pensiero razionale gli consente di **affrontare problemi e situazioni** sulla base di elementi certi e di avere consapevolezza dei limiti delle affermazioni che riguardano questioni complesse che non si prestano a spiegazioni univoche.

Si orienta nello spazio e nel tempo dando espressione a **curiosità e ricerca di senso; osserva ed interpreta** ambienti, fatti, fenomeni e produzioni artistiche.

Ha buone **competenze digitali**, usa con consapevolezza le tecnologie della comunicazione per ricercare e analizzare dati ed informazioni, per distinguere informazioni attendibili da quelle che necessitano di approfondimento, di controllo e di verifica e per interagire con soggetti diversi nel mondo.



COMPETENZE E CONTENUTI

LE COMPETENZE DEGLI STUDENTI

OBIETTIVI

- Definire una matrice comune di competenze digitali che ogni studente deve sviluppare
- Sostenere i docenti nel ruolo di facilitatori di percorsi didattici innovativi, definendo con loro strategie didattiche per potenziare le competenze chiave
- Coinvolgere gli studenti attraverso format didattici innovativi e 'a obiettivo'
- Innovare i curricula scolastici

AZIONI

- Azione #14 - Un framework comune per le competenze digitali degli studenti
- Azione #15 - Scenari innovativi per lo sviluppo di competenze digitali applicate
- Azione #16 - Una research unit per le Competenze del 21mo secolo
- Azione #17 - Portare il pensiero computazionale a tutta la scuola primaria
- Azione #18 - Aggiornare il curriculum di "Tecnologia" alla scuola secondaria di



OBIETTIVI

- Definire una matrice comune di competenze digitali che ogni studente deve sviluppare
- Sostenere i docenti nel ruolo di facilitatori di percorsi didattici innovativi, definendo con loro strategie didattiche per potenziare le competenze chiave
- Coinvolgere gli studenti attraverso format didattici innovativi e 'a obiettivo'
- Innovare i curricula scolastici

AZIONI

- Azione #14 - Un framework comune per le competenze digitali degli studenti
- Azione #15 - Scenari innovativi per lo sviluppo di competenze digitali applicate
- Azione #16 - Una research unit per le Competenze del 21mo secolo
- Azione #17 - Portare il pensiero computazionale a tutta la scuola primaria
- Azione #18 - Aggiornare il curriculum di "Tecnologia" alla scuola secondaria di primo grado

Lingua e
matematica

Agire
consapevole

Indicazioni Nazionali e Nuovi scenari 2018

5.4 - il pensiero computazionale

Obiettivo 4. SDGs -

Fornire un'educazione di qualità, equa ed inclusiva, e opportunità di apprendimento per tutti

Nei contesti attuali, in cui la tecnologia dell'informazione è così pervasiva, la padronanza del coding e del pensiero computazionale possono aiutare le persone a governare le macchine e a comprenderne meglio il funzionamento, senza esserne invece dominati e asserviti in modo acritico.

Problem
solving

Pensiero logico
Interdisciplinarietà

Lingua e matematica, apparentate, sono alla base del pensiero computazionale... Attività legate al pensiero computazionale sono previste nei Traguardi delle Indicazioni in particolare nell'ambito della Tecnologia, tuttavia se ne possono prevedere in ogni ambito del sapere.

Per **pensiero computazionale** si **intende un processo mentale** che consente di risolvere problemi di varia natura seguendo metodi e strumenti specifici pianificando una **strategia**. È un **processo logico creativo** che, più o meno consapevolmente, viene messo in atto nella vita quotidiana per affrontare e risolvere problemi.

L'educazione ad **agire consapevolmente** tale strategia consente di apprendere ad affrontare le situazioni in modo analitico, scomponendole nei vari aspetti che le caratterizzano e pianificando per ognuno le soluzioni più idonee.

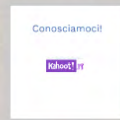
Tali strategie sono indispensabili nella **programmazione** dei computer, dei robot, ecc. che hanno bisogno di istruzioni precise e strutturate per svolgere i compiti richiesti. Tuttavia, nella didattica, si possono proficuamente mettere a punto attività legate al **pensiero computazionale anche senza le macchine.**

Ogni situazione che presupponga una procedura da costruire, un problema da risolvere attraverso una sequenza di operazioni, una rete di connessioni da stabilire (es. un ipertesto), si collocano in tale ambito, a patto che **le procedure e gli algoritmi** siano accompagnati da riflessione, **ricostruzione metacognitiva**, esplicitazione e giustificazione delle scelte operate.

Sostanzialmente, si tratta di una **educazione al pensiero logico e analitico** diretto alla soluzione di problemi. Impiegandolo in contesti di gioco educativo (es. la robotica), dispiega al meglio le proprie potenzialità, perché l'alunno ne constata immediatamente le molteplici e concrete applicazioni.

Ciò contribuisce alla costruzione delle competenze matematiche, scientifiche e tecnologiche, ma anche allo spirito di iniziativa, nonché all'affinamento delle competenze linguistiche.

UF 28 - Coding e pensiero computazionale



1

LESSON PROFILE

2

COMPUTATIONAL
THINKING

3

CODING

Learn to code
code to learn

4

NORMATIVA

5

CONCETTI E
TECNICHE

6

RISORSE

7

ASSESSMENT

8

CODING -
EDUCATION

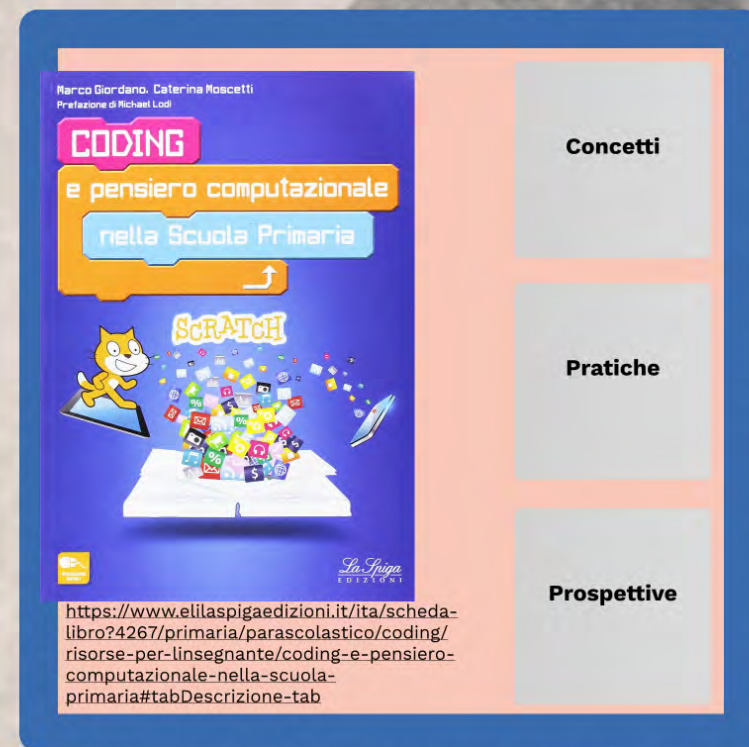
Doc. Silvia Mazzeo
(Iniziativa formativa ID.33586)
Scuola Polo IPSEOA PAOLA - CSRH07000Q

Guida per docenti



Computational thinking - A guide for teachers
© Copyright 2015 Computing At School
(<http://community.computingschool.org.uk/resources/2324>)

© Copyright 2016 Consiglio Nazionale delle Ricerche
http://pensierocomputazionale.itd.cnr.it/pluginfile.php/957/mod_page/content/7/Guida%20al%20Pensiero%20Computazionale.pdf



<https://www.elilaspigaedizioni.it/ita/scheda-libro?4267/primaria/parascolastico/coding/risorse-per-linsegnante/coding-e-pensiero-computazionale-nella-scuola-primaria#tabDescrizione-tab>

Computational thinking - A guide for teachers
© Copyright 2015 Computing At School
([http://community.computingatschool.org.uk/
resources/2324](http://community.computingatschool.org.uk/resources/2324))

© Copyright 2016 Consiglio Nazionale delle Ricerche
[http://pensierocomputazionale.itd.cnr.it/pluginfile.php/957/
mod_page/content/7/Guida%20al%20Pensiero
%20Computazionale.pdf](http://pensierocomputazionale.itd.cnr.it/pluginfile.php/957/mod_page/content/7/Guida%20al%20Pensiero%20Computazionale.pdf)

Concetti del pensiero computazionale



Ragionamento logico

Dare senso alle cose..... attraverso le varie azioni



Astrazione

Processo di rimozione di dettagli superflui



Valutazione

Bontà del processo, funzionale allo scopo



Pensiero algoritmico

Arrivare a una soluzione attraverso una chiara definizione dei passaggi



Scomposizione

Pensare ad artefatti in termini delle loro componenti



Generalizzazione

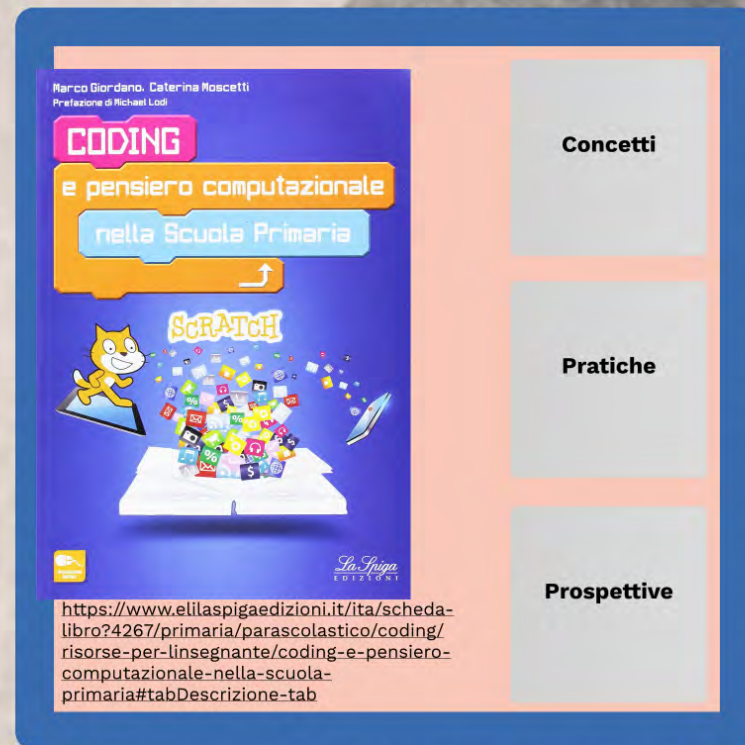
Schemi ricorrenti, somiglianze, connessioni... capitalizzare le precedenti esperienze

Guida per docenti



Computational thinking - A guide for teachers
© Copyright 2015 Computing At School
(<http://community.computingatschool.org.uk/resources/2324>)

© Copyright 2016 Consiglio Nazionale delle Ricerche
http://pensierocomputazionale.itd.cnr.it/pluginfile.php/957/mod_page/content/7/Guida%20al%20Pensiero%20Computazionale.pdf



Marco Giordano, Caterina Moschetti

Prefazione di Michael Lodi

CODING

e pensiero computazionale

nella Scuola Primaria

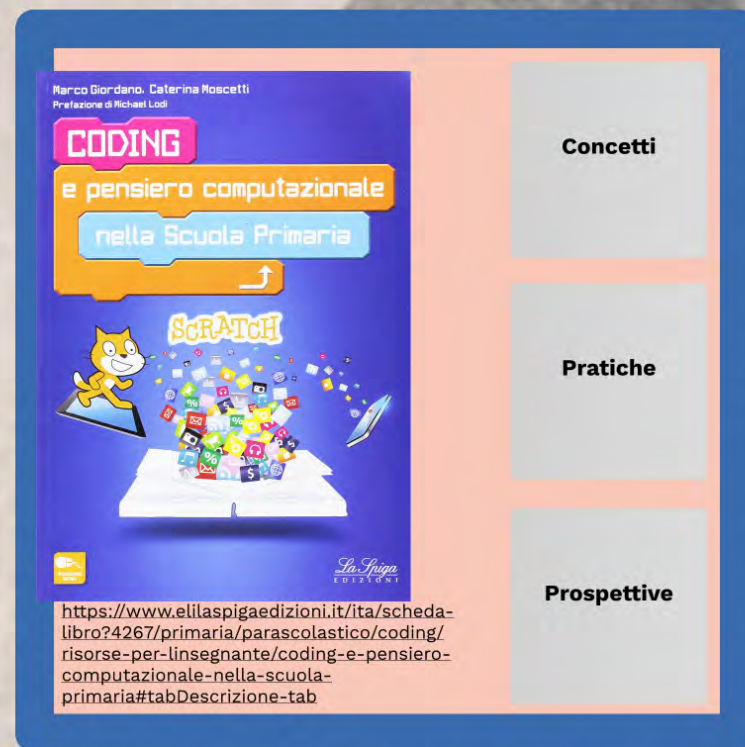


Guida per docenti



Computational thinking - A guide for teachers
© Copyright 2015 Computing At School
(<http://community.computingatschool.org.uk/resources/2324>)

© Copyright 2016 Consiglio Nazionale delle Ricerche
http://pensierocomputazionale.itd.cnr.it/pluginfile.php/957/mod_page/content/7/Guida%20al%20Pensiero%20Computazionale.pdf



Concetti comuni ai linguaggi di programmazione



Sequenze

passi ordinati per risolvere un problema



Condizioni

agire sulla base del verificarsi di una condizione



Ripetizioni

eseguire più volte le stesse istruzioni senza doverle ripetere



Eventi

eseguire istruzioni nel momento accade qualcosa



Parallelismo

più istruzioni vengono eseguite nello stesso tempo



Operatori

espressioni matematiche e logiche



Dati

raccogliere, analizzare e rappresentare informazioni

Pratiche e modi di procedere



Essere incrementali ed iterativi

sviluppare un progetto per raffinamenti successivi



Testing e debugging

provare il progetto, ricercare gli errori, risolverli



Riuso e remixing

costruire qualcosa di nuovo sulla base di programmi già scritti da altri



Simulazione

modellare un processo ed eseguire esperimenti per testarne l'efficacia



Scomposizione

scomporre un problema in parti per ottenere una soluzione completa



Astrazione

ridurre la complessità (tralasciare dettagli irrilevanti) per far emergere l'idea principale



Generalizzazione

applicare modelli usati a problemi simili



Automazione

svolgere compiti ripetitivi con pochi comandi



Efficienza, calcolabilità, complessità

individuare la miglior strategia risolutiva

Sviluppo di prospettive



Esprimere se stessi

la computazione come strumenti creativo e di espressione



Interrogarsi

utilizzare la computazione per farsi domande sul funzionamento del mondo che ci circonda



Ambiguità e problemi aperti

individuare le varie possibili soluzioni con idee nuove, personali e creative



Connettersi

riconoscere il potere della collaborazione con gli altri nella creazione di qualcosa



Gestire la complessità

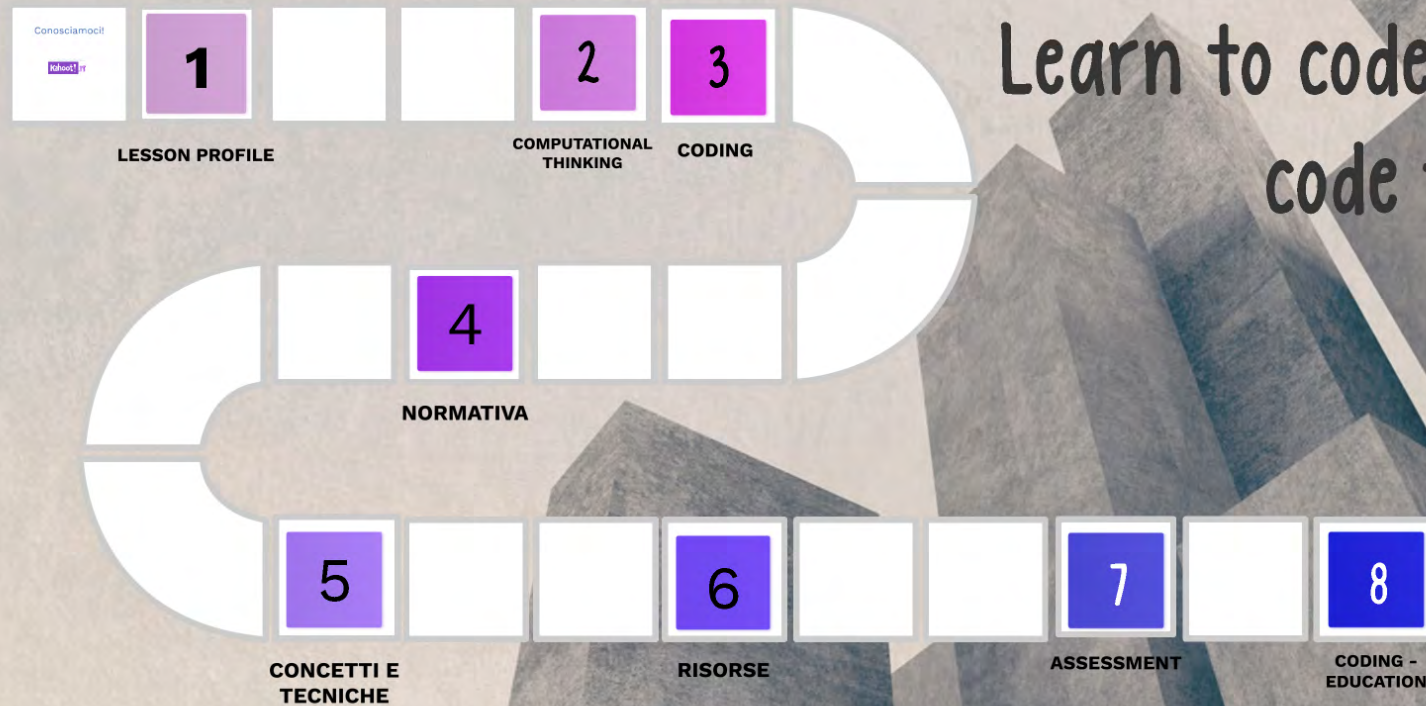
non temere il problema, affrontandolo per piccoli passi

Approcci principali

Problem solving

Costruzione creativa

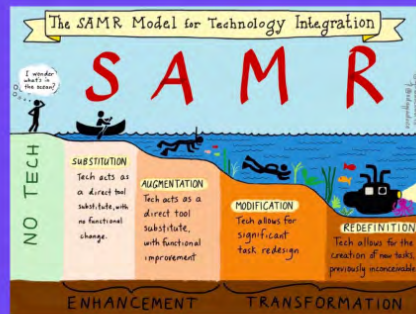
UF 28 - Coding e pensiero computazionale



Learn to code
code to learn

Doc. Silvia Mazzeo
(Iniziativa formativa ID.33586)
Scuola Polo IPSEOA PAOLA - CSRH07000Q

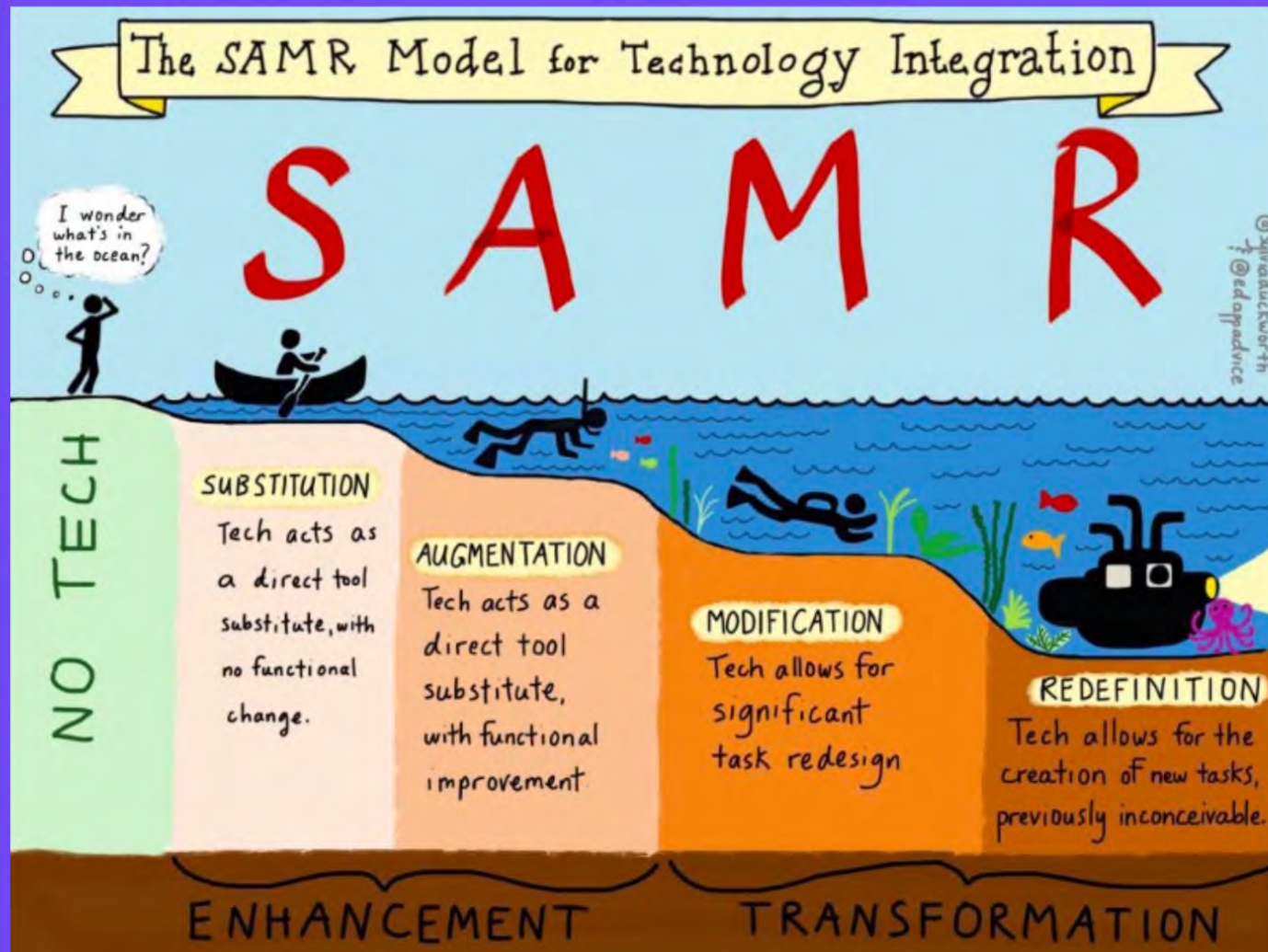
Gli strumenti per passare
da passivo
consumatore digitale
a realizzatore consapevole



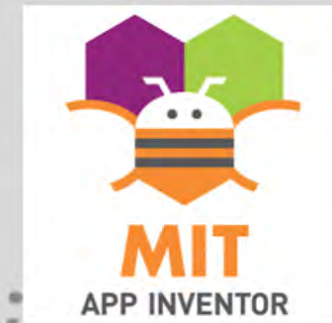
**Robot e
dispositivi
programmabili**

**App per
dispositivi
mobili**

**Programmi
e siti**



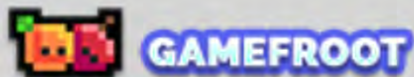
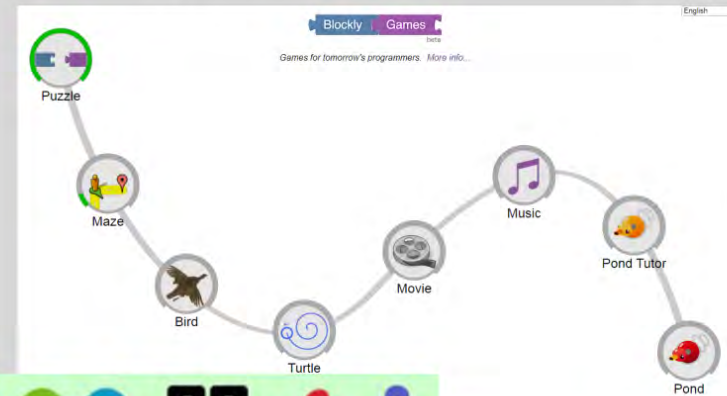
Programmi e siti



Made w/ Code

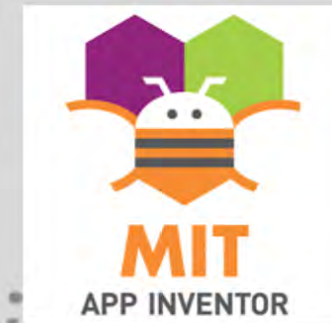


script;

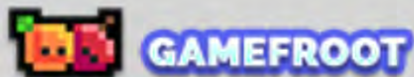
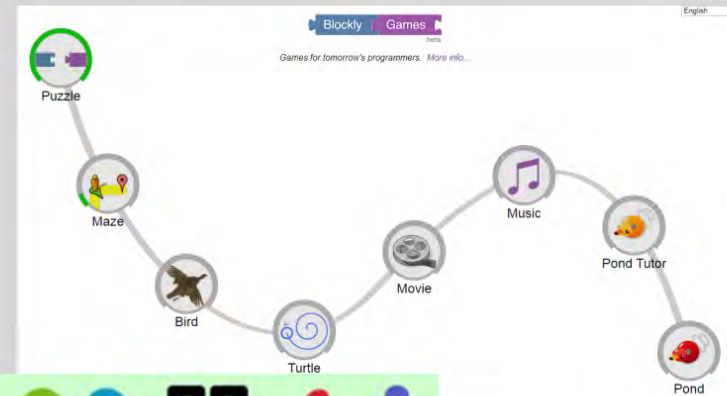




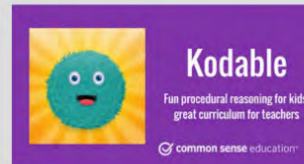
Programmi e siti



Made w/ Code



App per smartphone e tablet



Robotica educativa STEAM



OZOBOT Blockly



OPEN Roberta®



PIPER



littleBits™



Dot

ing
ode



Oh So Savvy Mom



P I P E R



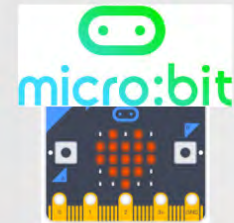


WeDo 2.0

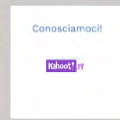


education

Robotica educativa STEAM



UF 28 - Coding e pensiero computazionale



1

LESSON PROFILE

2

COMPUTATIONAL
THINKING

3

CODING

Learn to code
code to learn

4

NORMATIVA

5

CONCETTI E
TECNICHE

6

RISORSE

7

ASSESSMENT

8

CODING -
EDUCATION

Doc. Silvia Mazzeo
(Iniziativa formativa ID.33586)
Scuola Polo IPSEOA PAOLA - CSRH07000Q

L'ambiente di apprendimento

- tecnologia leggera dentro le classi
- spazio fluido e modificabile
- isole per gruppi di lavoro
- utilizzo degli strumenti personali (BYOD)
- angoli e spazi tematici attrezzati
- momenti di condivisione

Collaborazione
e confronto

Approccio
interdisciplinare

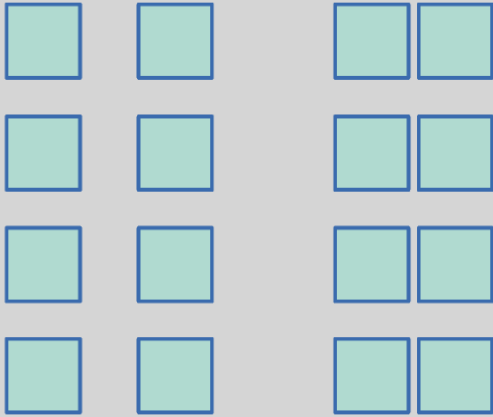
Ambiente di
programmazione
...
e di apprendimento

Collaborazione e confronto

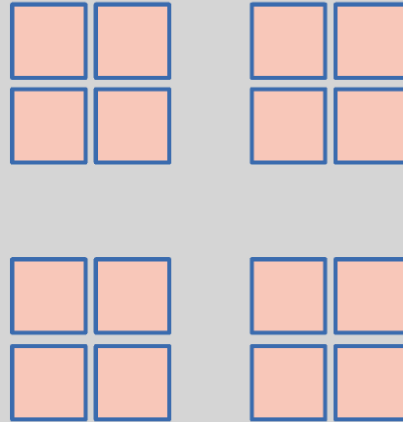
- piccoli gruppi
- tutoring del docente
- problem solving group
- disordine creativo vs staticità ordinata
- cooperative learning
- clima relazionale positivo



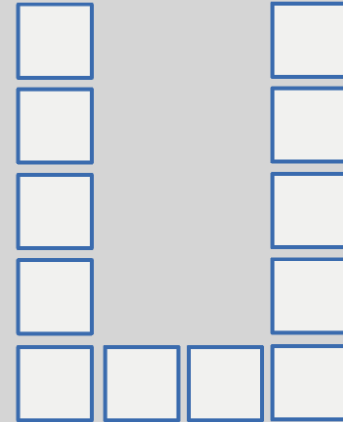
SETTING



Rows



Groups



Letter 'U'

Collaborazione e confronto

- piccoli gruppi
- tutoring del docente
- problem solving group
- disordine creativo vs staticità ordinata
- cooperative learning
- clima relazionale positivo



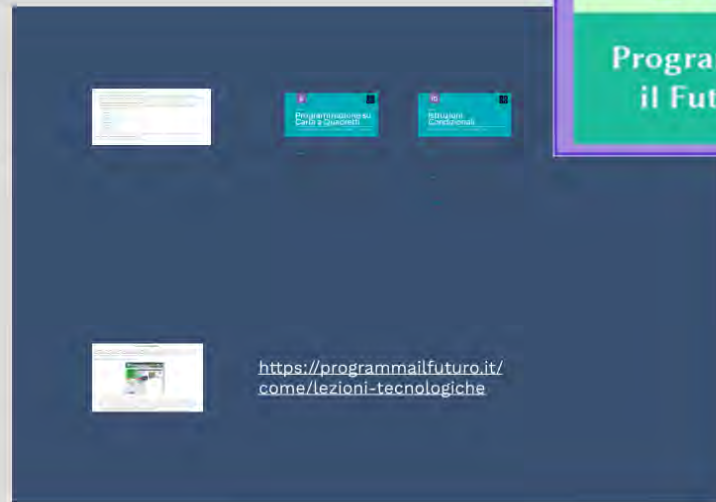
Approccio interdisciplinare

si impara a programmare e si programma per imparare!

(Mitchel Resnick)

fase analogica

fase digitale



Il percorso delle lezioni tradizionali avviene secondo due modalità:

1. una modalità **base** (denominata l'*Ora del Codice*) consistente nello svolgere solo un'ora di avviamento al *pensiero computazionale*. Si suggerisce che questa attività avvenga nelle settimane 6-21 ottobre 2018 oppure 3-9 dicembre 2018, in concomitanza con analoghe attività in corso in tutto il mondo. Accedi alla versione tradizionale dell'Ora del Codice chiamata *Pensiero Computazionale*
2. una modalità **avanzata** consistente nel far seguire a quest'ora di avviamento un percorso più approfondito, che sviluppa i temi del *pensiero computazionale* con ulteriori lezioni tradizionali, da svolgere quando si vuole. Dopo *Pensiero Computazionale* prosegui in sequenza con le lezioni sotto indicate:
 1. *Programmazione su carta a quadretti*
 2. *Algoritmi*
 3. *Funzioni*
 4. *Istruzioni condizionali*
 5. *Composizione di canzoni*
 6. *Astrazione*
 7. *Programmazione a staffetta*
 8. *Internet*
 9. *Conclusione*

Per commenti e richieste comuni a tutte le lezioni tradizionali consulta la relativa [bacheca generale di discussione](#). Per commenti relativi a una specifica lezione usa la bacheca della relativa lezione.

Nell'a.s. 2014-15 queste lezioni si chiamavano Lezioni "Senza Rete".

LEZIONE:

Programmazione su Carta a Quadretti

Tempo della lezione: 45-60 Minuti. Tempo di preparazione: 10 Minuti. (revisione 2016)

Obiettivo Principale:

Aiutare gli studenti a capire cos'è la programmazione

SOMMARIO:

Lo studente impara a capire cos'è davvero la programmazione dando istruzioni ad un altro studente affinché faccia un disegno. L'obiettivo è far sì che uno studente scriva un programma seguendo il quale un altro studente possa colorare le caselle di un foglio di carta a quadretti in modo da riprodurre un disegno esistente. Se c'è tempo, nella lezione si può affrontare la creazione da parte degli studenti di disegni originali.

OBIETTIVI:

- Gli studenti:
- Comprendono la difficoltà di tradurre problemi reali in programmi.
 - Imparano che idee che sembrano chiare, possono comunque venir interpretate in modo scorretto da un calcolatore.
 - Capiscono la necessità di strutture di programmazione formali come ripetizioni e funzioni.

MATERIALI:

- Kit per Disegni/Algoritmi di esempio (da pag. 8 in avanti).
- Scheda dei Simboli per la Programmazione (a pag. 5).
- Carta a quadretti grandi.
- Pennarelli, penne, o matite (due o tre colori).

PREPARAZIONE:

Stampa un kit per Disegni/Algoritmi per ogni gruppo.

Stampa una Scheda dei Simboli per la Programmazione per ogni gruppo.

Fornisci ad ogni gruppo un po' di fogli per i disegni.

VOCABOLARIO:

Algoritmo – Una serie di passi che descrivono come portare a termine un compito.

Programmazione – Espressione di un algoritmo mediante una sequenza di istruzioni in un formato eseguibile da un calcolatore.

Debug – Attività di correzione degli errori (bug).

Funzione – Una parte di codice alla quale è associato un nome e che può essere chiamata più volte.

Parametro – Informazione aggiuntiva che può essere passata ad una funzione per personalizzarne il comportamento.

LEZIONE:

Istruzioni Condizionali

Tempo della lezione: 45-60 Minuti. - Tempo di preparazione: 2 Minuti.

Obiettivo Principale: Questa lezione introduce il concetto di istruzione condizionale e le specifiche istruzioni "se" e "se-altrimenti".

SOMMARIO:

Questo è un gioco per l'intera classe. Dopo aver introdotto in generale il concetto di istruzione condizionale ed aver spiegato le specifiche istruzioni "se" e "se-altrimenti", gli studenti si esercitano tutti insieme usando un programma già scritto, un mazzo di carte, e un po' di fortuna.

OBIETTIVI:

- Gli studenti:
- Acquisiscono esperienza nel determinare il risultato (vero o falso) di condizioni logiche.
 - Valutano delle condizioni logiche per determinare quale ramo di un programma seguire.

MATERIALI:

- 1 mazzo di carte da gioco francesi.

PREPARAZIONE:

Separa le carte vestite (figure) dalle altre. I Jolly vanno tenuti assieme alle carte non vestite (numerale), e possono essere utilizzati per concedere un vantaggio

speciale a chi li detiene, come ad esempio la vittoria istantanea o la cancellazione di tutti i punti dell'avversario.

Proietta sullo schermo il programma appropriato al livello della tua classe (nell'ultima pagina ci sono degli esempi).

VOCABOLARIO:

Istruzione Condizionale – Un'istruzione che richiede la verifica di una condizione

Istruzione "se" – Un'istruzione che determina se eseguire o meno altre istruzioni

Altrimenti – Un altro modo per dire "in caso contrario".

Funzione – Una parte di codice che può essere chiamata più volte.

Incrementare – Aggiungere una certa quantità (solitamente 1).

Decrementare – Sottrarre una certa quantità (solitamente 1).

Istruzioni annidate – Istruzioni all'interno di altre istruzioni.

Lezioni tecnologiche

Le **lezioni tecnologiche** sono fruibili tramite web e sono disponibili in 5 Corsi, suddivisi in una serie di esercizi progressivi. Non è richiesta alcuna abilità tecnica, al di là di una elementare capacità di navigare su Internet, né è necessaria una particolare preparazione scientifica. (Nota: per chi non ha a disposizione un calcolatore o non ha la connessione Internet sono disponibili le [lezioni tradizionali](#).)

Vedi più oltre le [indicazioni per scegliere il Corso più adatto](#).

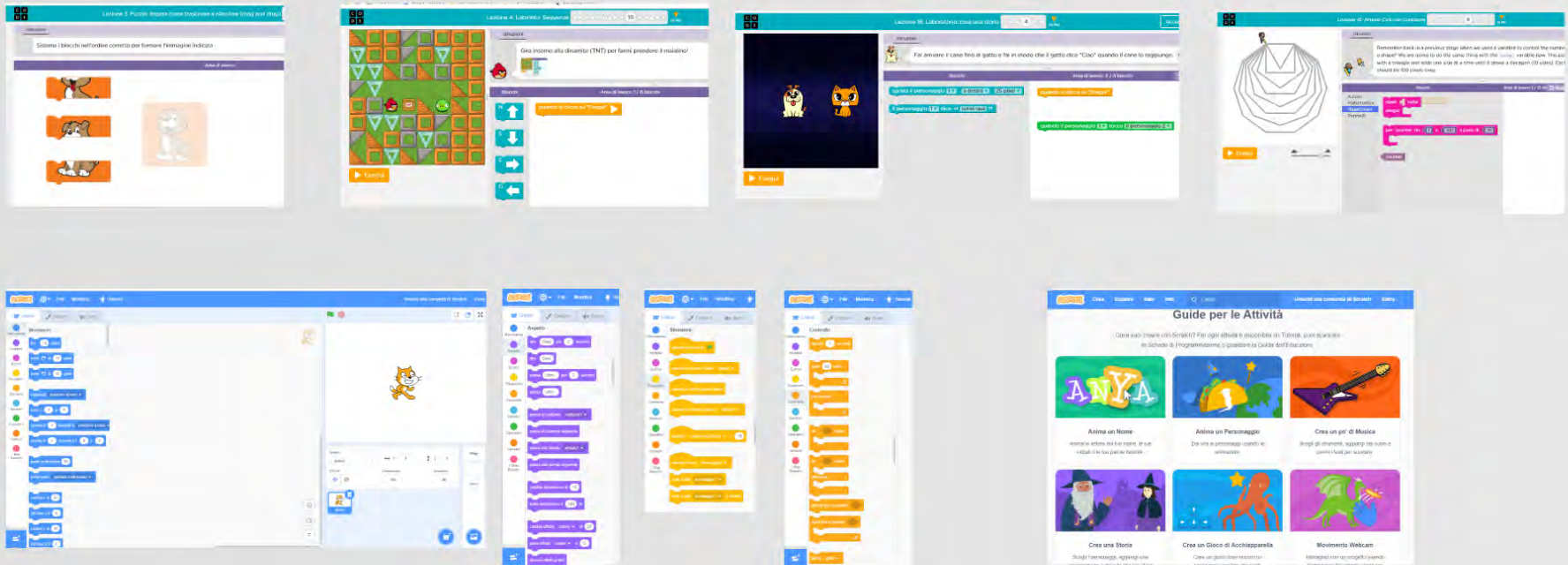


Il percorso delle lezioni tecnologiche avviene secondo due modalità:

1. Una modalità **base** (denominata *l'Ora del Codice*) consistente nello svolgere solo un'ora di avviamento al *pensiero computazionale*. Si suggerisce che questa attività avvenga nelle settimane 6-21 ottobre 2018 oppure 3-9 dicembre 2018, in concomitanza con analoghe attività in corso in tutto il mondo. Puoi svolgere una qualunque delle 11 versioni tecnologiche disponibili sulla pagina dell'*Ora del Codice* (c'è anche un'esercitazione introduttiva per chi non ha mai interagito con un computer).
2. Una modalità **avanzata** consistente nel far seguire a quest'ora di avviamento uno o più percorsi maggiormente articolati che

Ambiente di programmazione

codice di programmazione visuale a blocchi



Istruzioni

Sistema i blocchi nell'ordine corretto per formare l'immagine indicata

Area di lavoro:



C

O

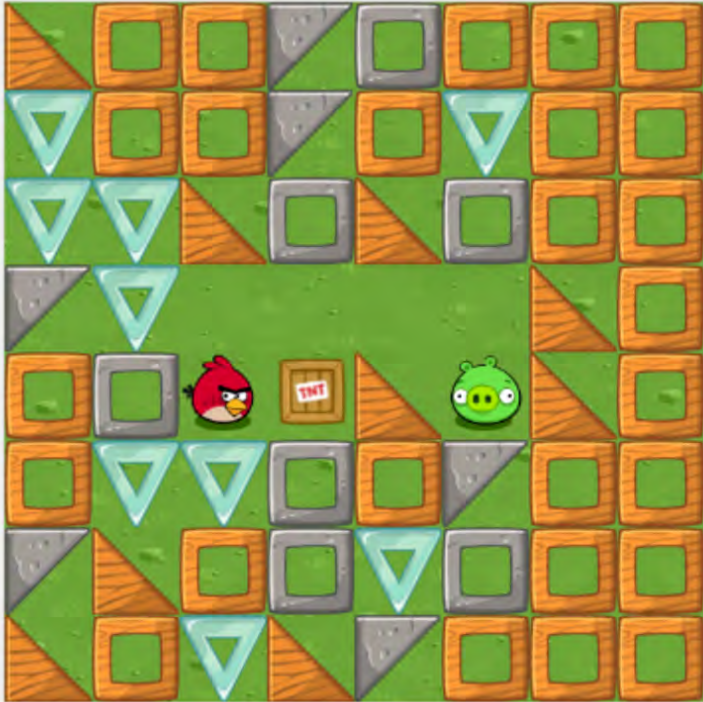
D

E

Lezione 4: Labirinto: Sequenze

10

DI PIÙ



The game interface shows a 10x10 grid maze. A red bird is at row 6, column 3. A green pig is at row 6, column 7. A TNT block is at row 6, column 4. The maze is composed of green grass, brown wooden blocks, and grey stone blocks. Some cells contain blue triangular obstacles.

Esegui

Istruzioni

Gira intorno alla dinamite (TNT) per farti prendere il maialino!



Blocchi

Area di lavoro: 1 / 6 blocchi

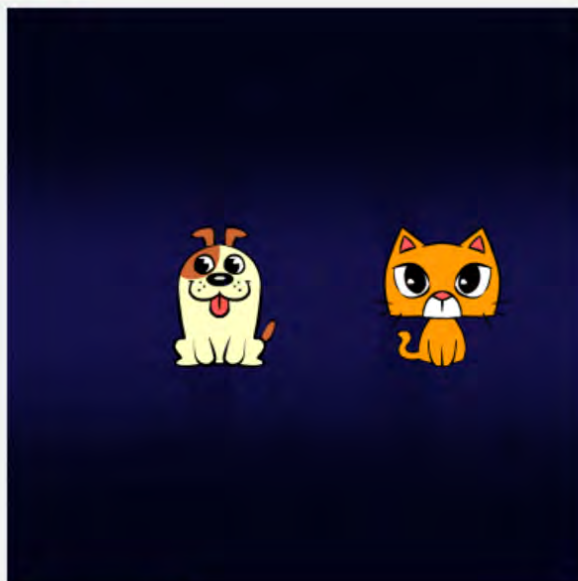
N

S

E

O

quando si clicca su "Esegui"



▶ Esegui

Istruzioni



Fai arrivare il cane fino al gatto e fai in modo che il gatto dica "Ciao" quando il cane lo raggiunge.

Blocchi

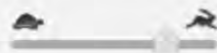
Area di lavoro: 2 / 4 blocchi

sposta il personaggio 1 a destra 25 pixel

il personaggio 1 dice "scrivi qua"

quando si clicca su "Esegui"

quando il personaggio 1 tocca il personaggio 2

 Esegui

Istruzioni



Remember back in a previous stage when we used a variable to control the number of sides of a shape? We are going to do the same thing with the `counter` variable now. This palette starts with a triangle and adds one side at a time until it draws a decagon (10 sides). Each side should be 100 pixels long.

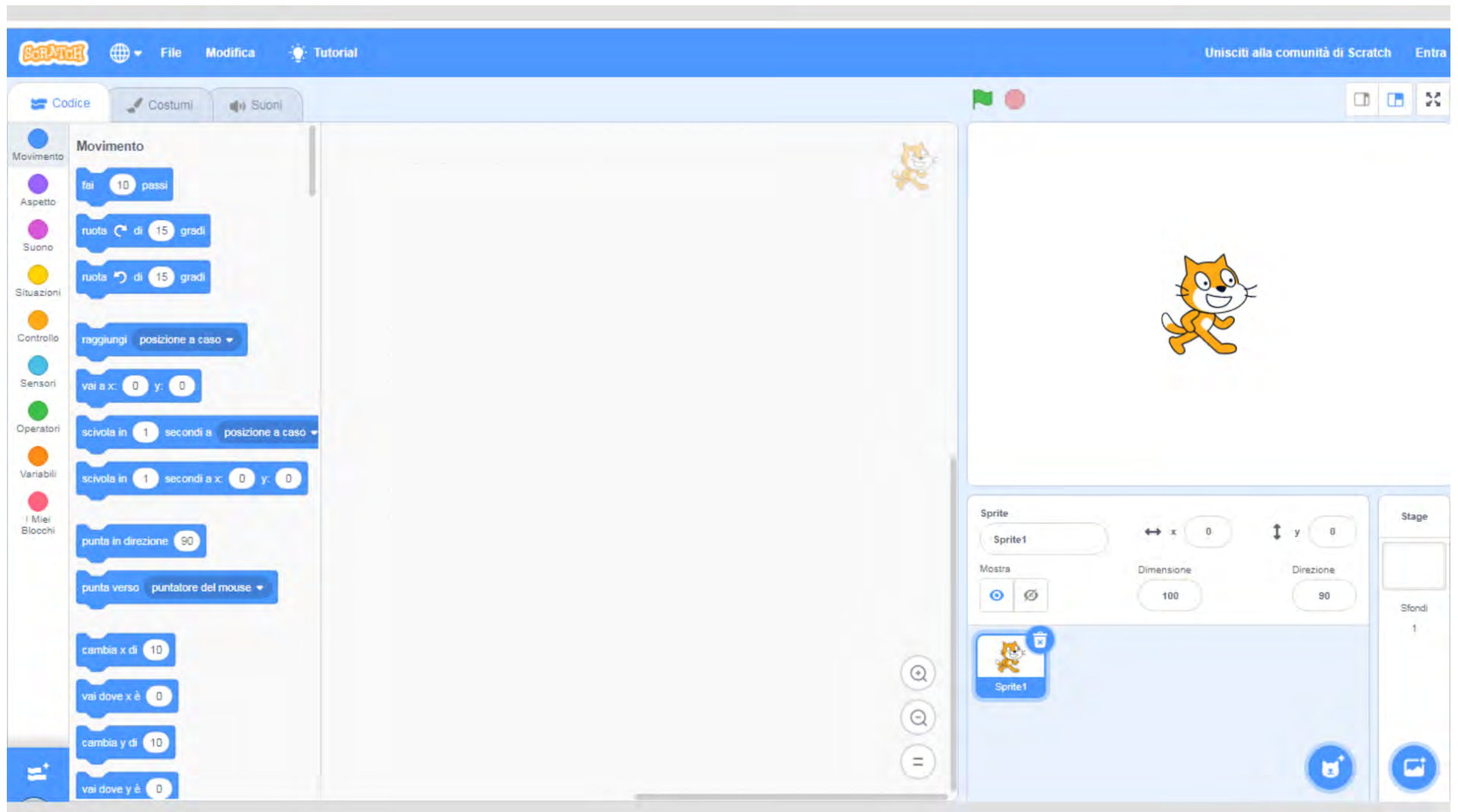
Blocchi

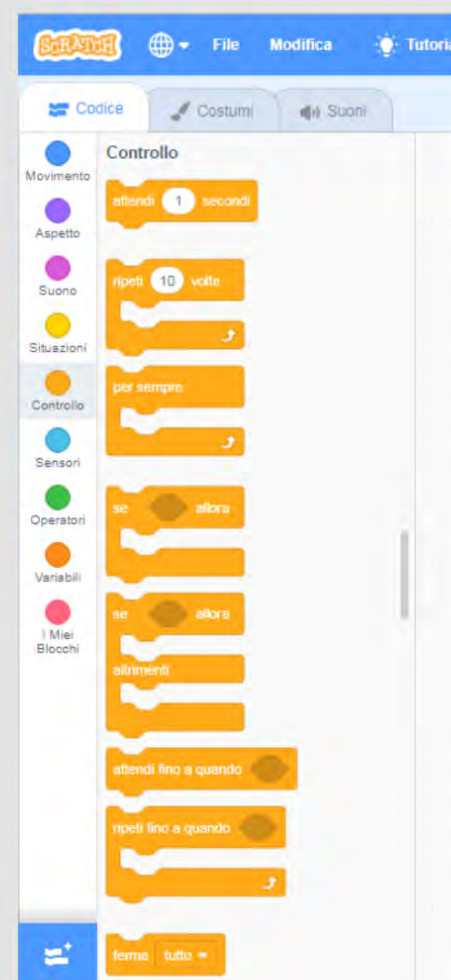
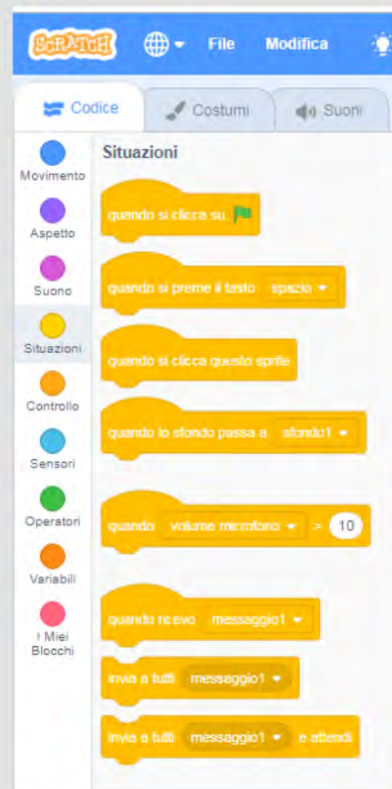
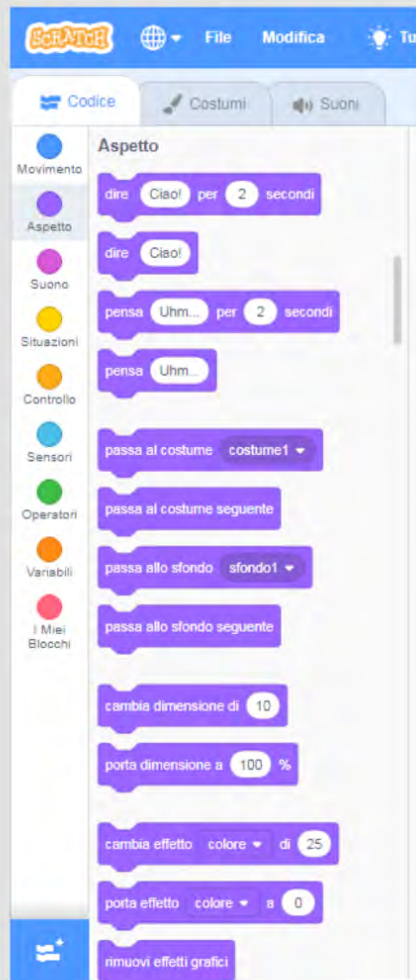
Area di lavoro: 1 / 13 blocchi

Azioni
Matematica
Ripetizioni
Pennelliripeti
esegui volte

per counter da 1 a 100 a passi di 10

counter





Guide per le Attività

Cosa vuoi creare con Scratch? Per ogni attività è disponibile un Tutorial, puoi scaricare le Schede di Programmazione o guardare la Guida dell'Educatore.



Anima un Nome

Anima le lettere del tuo nome, le tue iniziali o le tue parole favorite.



Anima un Personaggio

Dai vita ai personaggi usando le animazioni.



Crea un po' di Musica

Scegli gli strumenti, aggiungi dei suoni e premi i tasti per suonare.



Crea una Storia

Scegli i personaggi, aggiungi una conversazione e dai vita alla tua storia.



Crea un Gioco di Acchiapparella

Crea un gioco dove rincorri un personaggio per fare dei punti.

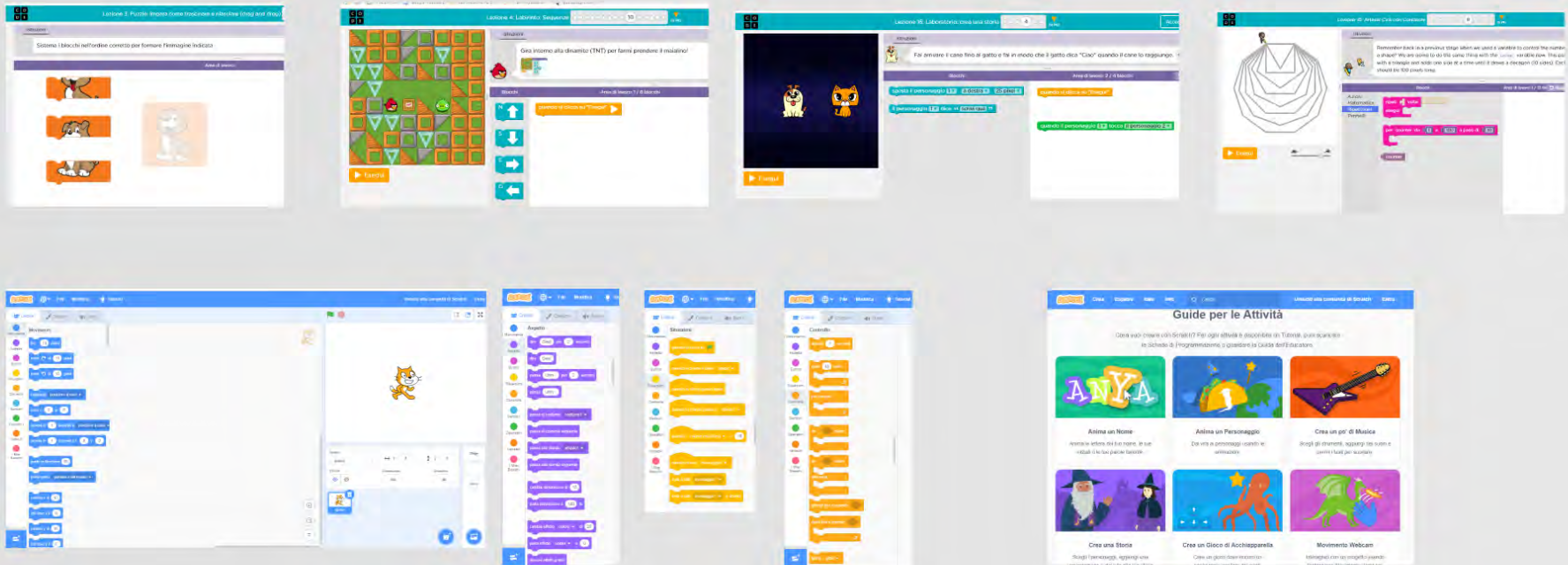


Movimento Webcam

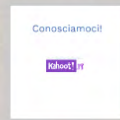
Interagisci con un progetto usando l'estensione Movimento Webcam.

Ambiente di programmazione

codice di programmazione visuale a blocchi



UF 28 - Coding e pensiero computazionale



1

LESSON PROFILE

2

COMPUTATIONAL
THINKING

3

CODING

Learn to code
code to learn

4

NORMATIVA

5

CONCETTI E
TECNICHE

6

RISORSE

7

ASSESSMENT

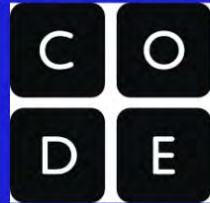
8

CODING -
EDUCATION

Doc. Silvia Mazzeo
(Iniziativa formativa ID.33586)
Scuola Polo IPSEOA PAOLA - CSRH07000Q

Coding for education

<https://code.org>



<https://scratch.mit.edu>



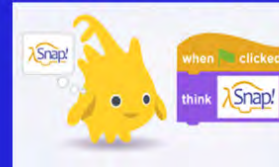
<https://programmailfuturo.it>



https://csfirst.withgoogle.com/s/en/home?_ga=2.3461296.1676496936.1570199268-988793491.1570199268



<https://snap.berkeley.edu>





Come partecipare

I percorsi didattici del progetto **Programma il futuro** sono disponibili al [sito di fruizione](#) delle lezioni. Il sito che stai leggendo ha invece lo scopo di guidarti ed assisterti nel loro utilizzo.

Nota: lo sappiamo, il link precedente non funziona! :-) Termina per favore di leggere questa pagina ed assicurati di aver letto le [informazioni sulla partecipazione](#). Dopo, il tuo accesso al [sito di fruizione](#)⁷ delle lezioni sarà certamente più proficuo!

Sono disponibili cinque tipi di percorso:

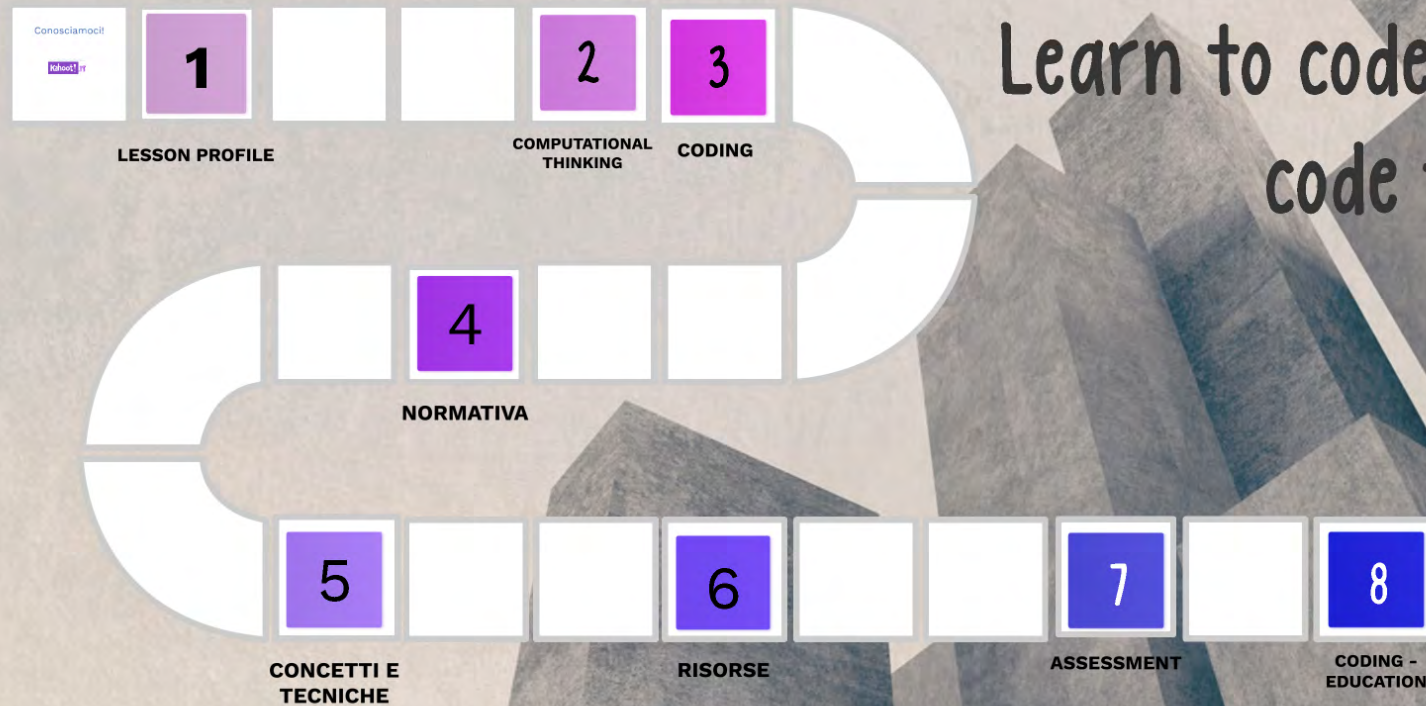
(1) Il corso [Principi dell'Informatica](#) (novità 2016), orientato agli studenti della scuola secondaria superiore in vista di un corso di studi universitari.



(2) Le [lezioni tecnologiche](#) sono fruibili tramite web e sono suddivise in una serie di esercizi progressivi. Non è richiesta alcuna abilità tecnica al di là di una elementare capacità di navigare su Internet. Né è necessaria alcuna particolare preparazione scientifica.

(3) Le [lezioni tradizionali](#) possono essere svolte senza la disponibilità di un calcolatore.

UF 28 - Coding e pensiero computazionale



Learn to code
code to learn

Doc. Silvia Mazzeo
(Iniziativa formativa ID.33586)
Scuola Polo IPSEOA PAOLA - CSRH07000Q