

Coding in STEM Education



<Informazioni di Stampa>

<Pubblicato da>

Science on Stage Deutschland e. V.
Am Borsigturm 15
13507 Berlino
Germania

<Coordinatore principale>

↳ **Dr Jörg Gutschank**, Leibniz Gymnasium
Dortmund International School, Dortmund, Germania
Presidente di Science on Stage Deutschland e. V.

<Coordinatori>

↳ **Sebastian Funk**, Villa Wewersbusch, Velbert-Langenberg,
Germania, Commissione Science on Stage Deutschland e. V.
↳ **Jean-Luc Richter**, Lycée Jean-Baptiste Schwilgué,
Sélestat, Francia, Vice Presidente di Science on Stage France
↳ **Bernard Schriek** (in pensione), Marien-Gymnasium, Werl,
Germania

<Coordinamento generale e redazione>

↳ **Daniela Neumann**, Project manager
Science on Stage Deutschland e. V.
↳ **Stefanie Schlunk**, Executive manager
Science on Stage Deutschland e. V.
↳ **Johanna Schulze**, Deputy executive manager
Science on Stage Deutschland e. V.

<Traduzione italiana a cura di>

↳ **Matteo Cattadori**, Liceo STEAM Rovereto (Trento)
↳ **Marco Nicolini**, Liceo Scientifico Statale "A. Tassoni",
Modena; Science on Stage Italia
↳ **Giovanni Pezzi**, Science on Stage Italia

<Edizione italiana a cura di>

<Design>

WEBERSUPIRAN.berlin

<Illustrazioni>

Rupert Tacke, Tricom Communication and Publishing GmbH

<Crediti>

Gli autori hanno verificato al meglio delle loro conoscenze tutti gli aspetti del diritto d'autore delle immagini e dei testi utilizzati in questa pubblicazione.

<Supportato da>

SAP SE

<Stampa della versione italiana a cura di>



www.iprase.tn.it

<Si prega di ordinare da>

www.science-on-stage.eu
info@science-on-stage.eu

<ISBN PDF>

978-3-942524-64-3

Questo lavoro è concesso in licenza Creative Commons Attribution-ShareAlike 4.0 International License:
<https://creativecommons.org/licenses/by-sa/4.0/>.



Prima edizione pubblicata nel 2019

© Science on Stage Deutschland e. V.



<Contenuti>

Saluto della Commissione Europea <04>

Saluto SAP SE <05>

Premessa <06>

Autori <07>



Scienze fondamentali con le cifre 1 e 0

Science Friction <30>

Rolling Sounds <36>

La Fisica animata <42>

Science Magic Box <48>

Ambiente 4.0



<08> Programmare H₂O

<14> Il funzionamento dell'acqua

<20> SPDV Salvavita per Piante Durante le Vacanze

<26> Il guanto magico

Il microcontrollo del mondo



<54> CoALA-Code a Little Animal

<58> Dati liquidi

<62> Capitan telecomando

<68> Come programmare

<72> Imparate la programmazione con Snap!

<73> Partecipare ai Meet and Code

<74> Altro materiale e Eventi nell'ambito del progetto
Coding in STEM Education

<75> Science on Stage Europe

<Saluto>

Sono molto lieto di dare il mio sostegno alla realizzazione di questa brochure sull'educazione di Scienze, Tecnologia, Ingegneria e Matematica – le importanti materie definite come STEM – e, più in generale, al progetto *Coding in STEM Education*.

Le discipline STEM sono fondamentali per costruire un'Europa competitiva e resiliente in futuro. La costruzione della conoscenza e dell'eccellenza è una parte fondamentale della nostra ambizione di costruire un vero spazio europeo dell'istruzione entro il 2025. Ci troviamo, tuttavia, di fronte ad un deficit di competenze. Dobbiamo fare di più per promuovere i temi STEM in Europa. In particolare, l'insegnamento di queste discipline deve essere una scelta di carriera più attraente e servono più modelli di comportamento che siano di esempio, soprattutto femminili.

Non si tratta solo di crescita economica e sviluppo. L'istruzione STEM deve essere inclusiva, consentendo agli studenti con diverse abilità e background di impegnarsi e sfruttare al meglio i loro talenti. Per costruire il futuro, i giovani hanno bisogno delle giuste competenze e attitudini – e quelle di area STEM devono essere al centro.

Grazie alla scienza possiamo imparare sia dal nostro passato che dal nostro presente; essa ci pone nelle condizioni di plasmare un futuro migliore e più consapevole. Affinché le generazioni future possano scegliere e vivere in una società più consapevole.

Voglio ringraziare Science on Stage Deutschland e.V., la rete europea per gli insegnanti di scienze, che è il cervello di questa operazione, e SAP SE, per aver sostenuto la realizzazione di questa brochure.

Gli insegnanti di sette nazioni europee hanno contribuito a sviluppare esempi pratici e consigli su come insegnare competenze di programmazione. Questo dimostra cosa è possibile realizzare quando condividiamo uno scopo ma anche quanto possiamo imparare dagli altri.

Progetti come *Coding in STEM Education* svolgono un ruolo chiave nella promozione delle materie STEM nelle scuole, aiutando i giovani europei ad acquisire le competenze vitali di cui hanno bisogno per avere successo nella vita. Mi congratulo con tutti coloro che sono coinvolti e spero che il loro esempio ne ispiri molti altri.

Tibor Navracsics

Commissario europeo per l'istruzione, la cultura, la gioventù e lo sport



<Saluto>

Secondo una previsione del World Economic Forum, circa il 65% dei bambini che iniziano oggi la scuola primaria lavorerà in professioni che attualmente non esistono ancora. E' chiaro che questi bambini troveranno più facilmente la loro strada se possiederanno determinate competenze tecniche: la digitalizzazione della nostra economia non può essere fermata. Pertanto, oltre alle capacità e conoscenze di lettura, scrittura e matematica, quelle in ambito nuove tecnologie sono diventate un argomento chiave per l'istruzione.

SAP partecipa da anni a iniziative europee nell'ambito dell'istruzione e della formazione di bambini e giovani, nei settori della robotica (*First Lego League*) e della tecnologia di programmazione (*Meet and Code*). Il nostro obiettivo è quello di introdurre le nuove tecnologie ai giovani in modi divertenti con lo scopo di favorire un inizio positivo del loro processo di apprendimento.

Oggi molti insegnanti vogliono trasmettere ai loro studenti conoscenze tecniche e digitali di base. Ciò non significa necessariamente che questi insegnanti debbano tutti essere degli

esperti, però necessitano ugualmente di materiali di lavoro pratici e collaudati per svolgere proficuamente le attività didattiche per l'apprendimento in diverse discipline anche tecniche e per diversi livelli.

Con il presente progetto, facilitiamo il processo di acquisizione di competenze digitali nella scuola e forniamo materiale didattico di qualità per i docenti delle materie STEM. In passato abbiamo già svolto numerosi progetti simili per sostenere le scuole attraverso il network di Science on Stage Germania, anche questa pubblicazione *Coding in STEM Education* si colloca all'interno di questa strategia già collaudata.

Sono lieto di questa collaborazione con Science on Stage Germany e focalizzata su un progetto così orientato a degli obiettivi pratici; sono convinto che anche questa edizione ripeterà il successo delle pubblicazioni precedenti. Vorrei anche ringraziare gli insegnanti che hanno dato il loro sostegno a questo lavoro.

Michael Kleinemeier

Membro del Comitato Esecutivo, SAP SE



<Premessa>

Questo opuscolo è speciale da molti punti di vista. Permettetemi di spiegare cosa voglio dire.

La programmazione è un'abilità di base nel mondo moderno di oggi ed è particolarmente importante nelle discipline di scienze, tecnologia, ingegneria e matematica (STEM). La programmazione informatica sta diventando una competenza sempre più ricercata e necessaria in tutti i settori della nostra vita e non può più essere lasciata ai soli specialisti di IT (Information Technology); ecco perché deve essere insegnata non solo nelle classi di informatica, ma in maniera trasversale anche in ogni altra materia. Purtroppo però, i programmi di studio europei delle materie STEM non rispondono adeguatamente a questa esigenza.

Nel progetto *Coding in STEM Education*, la rete europea per gli insegnanti di scienze di Science on Stage, ha sviluppato attività didattiche specifiche aventi lo scopo di cercare di colmare questo gap di competenze. L'obiettivo più generale della rete di insegnanti è quello di fornire a tutti i docenti STEM europei, una piattaforma specifica per lo scambio di idee e materiali. Attualmente il network di Science on Stage raggiunge 100.000 educatori distribuiti in oltre 30 paesi membri della UE.

Questo opuscolo è il frutto del lavoro di alcuni dei migliori insegnanti europei: 23 per l'esattezza e provenienti da sette nazioni, che si sono incontrati anche di persona per scambiare le loro idee e confrontarsi su metodi e ruolo della programmazione nell'educazione scientifica. Tutti hanno prodotto uno sforzo notevole, nell'arco dei 18 mesi di durata del processo.

Ognuna delle sette nazioni coinvolte era rappresentata da un team formato da un insegnante di scienze e uno di informatica che si è riunito, almeno una volta con quello di un altro paese, per confrontarsi sui temi della didattica della programmazione.

Hanno discusso questi temi e collaudato le attività con le loro classi, per assicurarsi che il materiale che tenete ora nelle vostre mani sia utile ed efficace per lo svolgimento di lezioni reali.

Questa pubblicazione si focalizza sulla programmazione di piccoli dispositivi elettronici come le schede Arduino, Calliope-mini, oppure il computer Raspberry Pi. Sono dispositivi economici ideati e realizzati specificatamente per essere utilizzati per l'apprendimento, il che li rende ideali per l'utilizzo a scuola.

I partecipanti hanno sviluppato 11 unità didattiche nei settori "Scienze fondamentali con le cifre 1 e 0", "Il microcontrollo del mondo" e "Ambiente 4.0". Sono tutti ottimi esempi di ciò che si potrebbe fare per introdurre la programmazione nei curricula di materie come biologia, chimica e fisica.

Il progetto *Coding in STEM Education* è stato reso possibile solo grazie agli sforzi dei nostri entusiasti partecipanti, che hanno svolto tutto questo lavoro nel tempo libero e in aggiunta al loro regolare lavoro di insegnamento. Grazie a tutti per questa stimolante pubblicazione! Un grazie speciali anche agli altri coordinatori quali Jean-Luc Richter, Bernd Schriek e Sebastian Funk, che hanno fatto un ottimo lavoro condensando in un'unica opera omogenea i pensieri e le idee eterogenee di insegnanti di diversa provenienza culturale e scientifica. Desidero inoltre ringraziare SAP SE e Gabriele Hartmann e Jens Mönig in particolare. Tale pubblicazione non sarebbe stata possibile senza il costante sostegno di questi partner professionali.

Coding in STEM Education è un libretto veramente speciale che è stato sviluppato e testato dagli insegnanti per i loro colleghi in Europa e non solo. Lasciate che vi ispiri nella realizzazione dei vostri progetti in classe!

Dr Jörg Gutschank

Presidente di Science on Stage Deutschland e. V.



<Autori>

<Cognome>	<Nome>	<Paese>	<Sezione>
Abad Nebot	Immaculada	Spagna	Il microcontrollo del mondo
Botelho	Lúcio	Portogallo	Ambiente 4.0
Compte Jové	Pere	Spagna	Il microcontrollo del mondo
Fernandes	Liliana	Portogallo	Ambiente 4.0
Funk	Sebastian	Germania	Coordinatore
Georgoulakis	Georgios	Grecia	Scienze fondamentali con le cifre 1 e 0
Giurgea	Mihaela Irina	Romania	Scienze fondamentali con le cifre 1 e 0
Gutschank	Jörg	Germania	Coordinatore Scienze fondamentali con le cifre 1 e 0
Hančl	Mirek	Germania	Il microcontrollo del mondo
Ivarra	Luc	Belgio	Scienze fondamentali con le cifre 1 e 0
Karagiorgou	Eleftheria	Grecia	Il microcontrollo del mondo
Lőkös	Annamária	Romania	Ambiente 4.0
Meier	Andreas	Germania	Ambiente 4.0
Mestvirishvili	Ilia	Georgia	Scienze fondamentali con le cifre 1 e 0
Nicolini	Marco	Belgio/Italia	Scienze fondamentali con le cifre 1 e 0
Padin	Beatriz	Spagna	Ambiente 4.0
Poncela	Elena	Spagna	Ambiente 4.0
Rațiu	Camelia Ioana	Romania	Scienze fondamentali con le cifre 1 e 0
Reis	Jorge	Portogallo	Ambiente 4.0
Richter	Jean-Luc	Francia	Coordinatore Ambiente 4.0
Schriek	Bernard	Germania	Coordinatore Il microcontrollo del mondo
Shapakidze	David	Georgia	Scienze fondamentali con le cifre 1 e 0
Toma	Corina Lavinia	Romania	Scienze fondamentali con le cifre 1 e 0
Tsiliki	Sevasti	Grecia	Il microcontrollo del mondo
Tsoutsoudakis	Astrinos	Grecia	Scienze fondamentali con le cifre 1 e 0
van der Byl	Sonja	Germania	Ambiente 4.0
Winckler	Julia	Germania	Il microcontrollo del mondo

Un ringraziamento speciale a Gabriele Hartmann e Jens Mönig di SAP SE per il loro supporto!
Ringraziamo Holger Bach e Paul Nugent per i loro preziosi consigli durante il processo di editing!



Programmare H₂O

<Autore> Beatriz Padin

<Autore> Elena Poncela



<Info>

<Parole chiave> acqua, sensori, efficienza, acquisizione dati, ambiente, evaporazione, condensazione, soluzioni, miscele, design, calore e temperatura, conduttori e isolanti, energia solare, radiazione infrarossa (IR), riflessione

<Discipline> fisica, scienze ambientali, chimica, informatica, matematica

<Età degli studenti> 13–15

<Hardware> Arduino^[1], Calliope mini^[2], Raspberry Pi^[3]

<Linguaggi> Arduino^[4], Python^[5], programmazione a blocchi

<Livello di difficoltà di programmazione> medio

<Sommario>

Gli studenti progettano, costruiscono e testano un depuratore solare per purificare l'acqua. Programmano sensori per misurare l'efficienza dei loro alambicchi solari.

<Introduzione concettuale>

Vengono trattati i seguenti concetti di fisica:

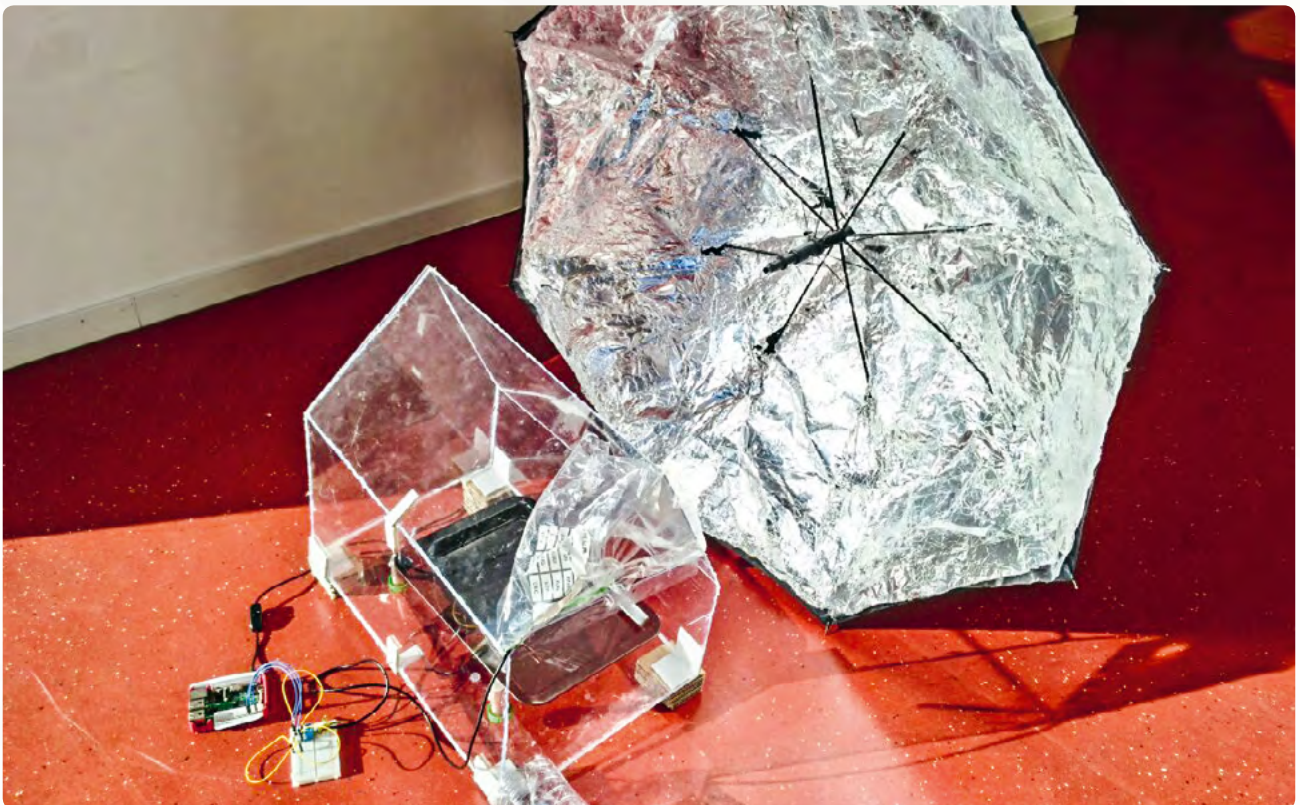
- ↳ Cambiamenti di stato (in particolare, evaporazione e condensazione) e loro caratteristiche principali
- ↳ Fattori che influenzano il processo di evaporazione (temperatura, superficie, ecc.)

- ↳ Effetti del calore: cambiamenti di stato
- ↳ Differenza tra fonti energetiche rinnovabili (energia solare) e non rinnovabili
- ↳ La radiazione IR e il suo ruolo nel trasporto di calore dal sole
- ↳ La radiazione e la sua riflessione su determinate superfici
- ↳ Metodi di separazione delle miscele
- ↳ Soluzioni: cosa sono, concentrazione (g/L e percentuale di massa, ecc.)

A seconda delle conoscenze pregresse degli studenti, essi potranno scoprire alcuni concetti da soli, mentre altri dovranno essere spiegati dall'insegnante e convalidati sperimentalmente dagli studenti.

Lo scopo dell'attività è quello di progettare e costruire il più efficiente impianto solare per purificare l'acqua. Può essere utilizzata solo energia solare. In primo luogo, l'efficienza del depuratore solare sarà determinata calcolando la percentuale in volume dell'acqua purificata che si ottiene. In seguito, gli studenti programmeranno diversi sensori per analizzare l'efficacia dei loro progetti.

Il compito principale di questa attività è quello di programmare i sensori, quindi gli insegnanti avranno bisogno di competenze di programmazione e alcune conoscenze di base per farlo.



© Depuratore solare

Dovranno sapere come costruire il circuito necessario per collegare i sensori alla scheda del microcontrollore. A seconda del loro livello di abilità, possono usare la programmazione a blocchi (Calliope mini^[2], Snap4Arduino^[6], ecc.) o la programmazione testuale (Arduino^[4], Python^[5], ecc.) per programmare i sensori.

<Cosa fanno gli studenti/insegnanti>

L'unità è composta da tre parti: progettazione e costruzione di un depuratore solare, programmazione dei sensori e verifica dell'alambicco solare.

<Parte uno: Progettare e costruire il depuratore solare>

Il progetto viene presentato agli studenti, che sperimentano il loro primo esempio di depuratore solare come classe, misurando la quantità di acqua pulita raccolta e calcolando l'efficienza di questo progetto. Nel fare ciò, esamineranno concetti fisici come i cambiamenti di stato, le soluzioni e l'energia solare.

Successivamente, gli studenti saranno incoraggiati a migliorare questo progetto iniziale. Per farlo, dovranno lavorare in gruppi (2–3 studenti per gruppo). Questa fase dell'attività può essere suddivisa in diversi compiti:

1. Gli studenti cercheranno informazioni sui depuratori solari, sul loro funzionamento, sui diversi modelli già in uso, ecc. Durante il processo di ricerca, saranno invitati a riflettere sulle seguenti domande:
 - a. Processo di evaporazione: quali sono i principali fattori che influenzano questo processo? Pensa alla superficie dove andrai a versare l'acqua sporca. E' meglio avere una superficie larga o stretta, una profondità maggiore o minore? Il colore del contenitore è importante?
 - b. Processo di condensazione: cosa è necessario per produrre la condensazione? Avete bisogno di progettare una superficie grande o piccola per la condensazione dell'acqua? Come farete in modo che l'acqua pulita si sposti al punto in cui desiderate raccoglierla?
 - c. Processo di radiazione: come massimizzare la radiazione IR che colpisce il vostro impianto solare? Come si può raggiungere la massima temperatura possibile nel vostro impianto solare? Prendete in considerazione l'utilizzo di una superficie ricoperta da un foglio di alluminio per riflettere la luce del sole nell'alambicco solare.
2. I gruppi creeranno i loro progetti e li spiegheranno all'insegnante.
3. Una volta che l'insegnante ha approvato il progetto, agli studenti sarà chiesto di trovare i materiali appropriati (a scuola, a casa, online, ecc.) e costruire il loro alambicco.
4. Gli studenti determineranno l'efficienza dei loro depuratori solari senza sensori. Utilizzando un cilindro graduato, essi

misurano non solo il volume di acqua sporca ma anche il volume di acqua pulita raccolta e applicano la seguente equazione matematica:

$$\text{Efficienza} = \frac{\text{volume di acqua raccolta}}{\text{volume di acqua impura}}$$

Per la seconda e la terza parte di questa attività viene fornita una guida passo dopo passo che include compiti e domande per gli studenti.

L'obiettivo è quello di offrire agli studenti scelte diverse per quanto riguarda i sensori, i linguaggi di programmazione e l'hardware, ma questo dipende anche dalle singole scuole/classi (materiali disponibili, conoscenza del linguaggio di programmazione, ecc.).

<Parte seconda: Utilizzo dei sensori>

Solo i migliori depuratori solari progettati saranno testati con l'ausilio di sensori. In questa parte, è necessario:

1. Selezionare la scheda del microcontrollore, il linguaggio di programmazione e i sensori con cui si intende lavorare. Rispondi alle seguenti domande per aiutarti a prendere la decisione migliore:
 - a. Utilizzate la programmazione a blocchi o la programmazione testuale? Se si sceglie la programmazione a blocchi, considerare l'utilizzo di Calliope mini^[2] o, in alternativa, programmare Raspberry Pi^[3] con Scratch^[8]. Nel caso in cui si optasse per la programmazione testuale, si potrebbe usare un Arduino^[4], o programmare in Python^[5] per Raspberry Pi^[9].
 - b. Hai intenzione di usare sensori digitali o analogici? Se si decide per quest'ultimo, Arduino potrebbe essere la scelta migliore.
2. Scegli i sensori che desideri utilizzare. Per semplificare le cose, è meglio limitarsi a selezionare non più di due sensori. Alcuni esempi di grandezze da misurare sono la temperatura, l'umidità, la pioggia e la radiazione IR. Prima di prendere una decisione, considera le specifiche di ciascun sensore. Il segnale di uscita è analogico o è un segnale digitale con solo due possibili valori (vero/falso)? Come si rapporta il segnale di uscita al valore del parametro che si sta misurando? Sono direttamente proporzionali? La potenza aumenta al diminuire del parametro o decresce?
3. Costruisci il circuito per collegare il sensore alla scheda del microcontrollore. Utilizza le risorse aggiuntive fornite^[7] o cerca esempi in Internet.

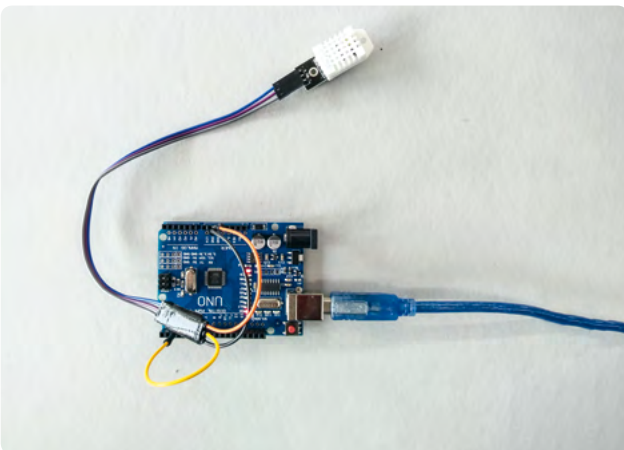


© Sensore di fiamma

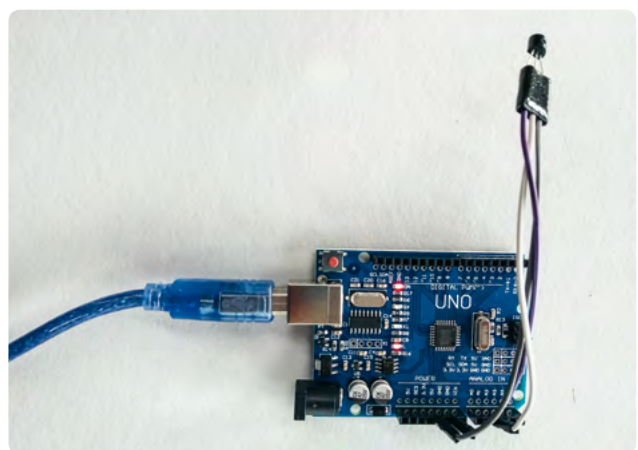
4. Programmazione dei sensori. Devi seguire questi passi:
- Scrivi quello che vuoi che il tuo programma faccia, considerando le caratteristiche dei sensori che hai selezionato. Vuoi che il programma visualizzi solo il valore del parametro misurato? Vuoi mostrare anche i valori massimi e minimi? Il sensore mostra il valore reale del parametro o è necessario effettuare dei calcoli e fare una conversione?
 - Scrivi il tuo programma. Non dimenticare di scrivere commenti sul tuo codice. È possibile utilizzare le risorse aggiuntive fornite dal proprio insegnante come guida.^[7]
5. Prova il tuo codice. Il programma funziona come previsto?
Esempi:
- ↳ È possibile misurare la quantità di radiazione infrarossa che raggiunge l'alambicco solare con un sensore di fiamma e un Arduino UNO^[4]. Se si utilizza una super-

ficie ricoperta da un foglio di alluminio, utilizza questo sensore per verificare se l'irraggiamento viene riflesso nell'alambicco solare.

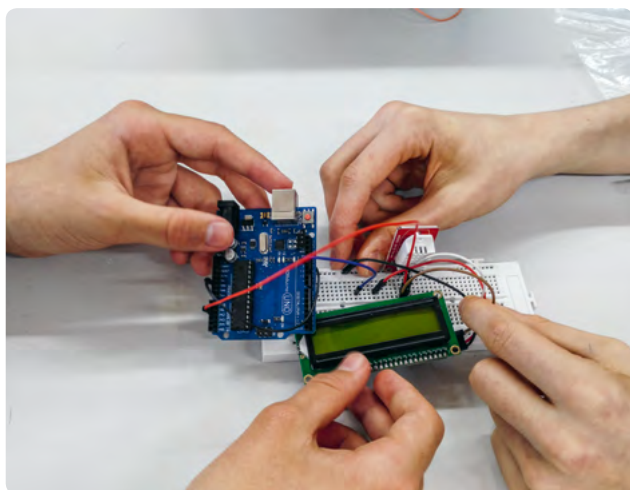
- ↳ È possibile registrare la temperatura massima e l'umidità relativa raggiunta all'interno dell'alambicco solare con i sensori di umidità e temperatura DHT11 o DHT22 e il vostro Arduino UNO.
- ↳ È possibile scrivere un programma Python 3 per determinare il tempo necessario affinché le prime gocce d'acqua si condensino con il sensore pioggia FR-04 e Raspberry Pi^[3]. Questo sensore può essere utilizzato anche con un Calliope mini e programmato con Snap!^[10] (linguaggio di programmazione a blocchi).



© Arduino con sensore di umidità



© Arduino con sensore di temperatura



© Arduino con display LCD e sensore di umidità

<Parte terza: Test del depuratore solare con sensori>

Il vostro team deve utilizzare i sensori programmati per testare e confrontare il vostro progetto con un altro. I depuratori solari devono lavorare nelle stesse condizioni e utilizzare lo stesso tipo di sensori perché ciò avvenga.

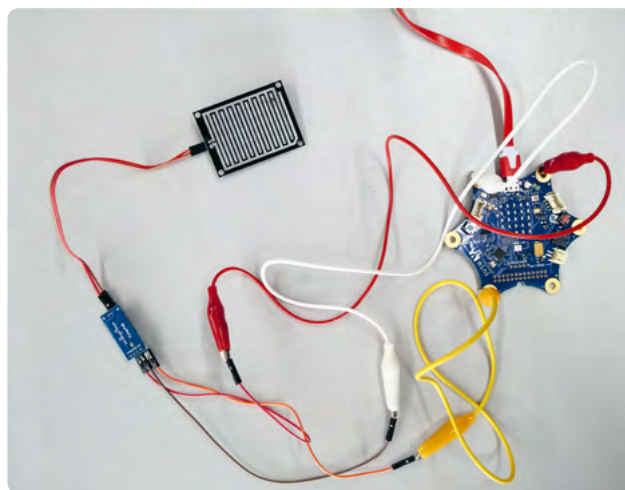
Si possono analizzare le ragioni per cui le efficienze sono diverse, prendendo nota di fattori chiave come le caratteristiche che influenzano l'evaporazione, ecc.

Come migliorare l'efficienza del depuratore solare:

- ↳ Provatelo in una giornata di sole attorno a mezzogiorno.
- ↳ Lasciate che il vostro apparecchio abbia il tempo sufficiente per raggiungere la temperatura più alta possibile.
- ↳ Assicuratevi che il depuratore sia ancora sigillato per evitare qualsiasi perdita di vapore.
- ↳ Rimuovete continuamente l'acqua pulita per evitare che evapori.
- ↳ Colorate l'acqua sporca per garantire il corretto funzionamento del depuratore.
- ↳ Scegliete un grande contenitore nero per l'acqua sporca.
- ↳ Utilizzate un ombrello coperto da un foglio di alluminio per riflettere la luce del sole nel depuratore solare.

<Risultati>

La prima parte dell'attività (costruzione dell'alambicco solare) ha portato a progetti che variavano enormemente nella loro efficienza. Se testato nelle giornate di sole di maggio/giugno in Spagna, il miglior progetto ha purificato il 95% dell'acqua sporca in 24 ore. Un tasso di efficienza del 54% è stato ottenuto anche in 4 ore. Tuttavia, alcuni depuratori non hanno purificato l'acqua a causa di carenze varie nella loro progettazione.



© Sensore pioggia con Calliope mini

Grazie poi all'utilizzo dei sensori, gli studenti hanno ottenuto i seguenti risultati:

- ↳ La temperatura massima raggiunta all'interno degli alambicchi solari dopo un'ora in una giornata di sole è stata di 65°C.
- ↳ È stata analizzata l'influenza del colore del contenitore di acqua sporca. Confrontando un contenitore bianco e uno nero lasciati al sole per alcuni minuti, la temperatura dell'acqua nel contenitore nero è risultata superiore di quasi 5°C rispetto a quella del contenitore bianco.
- ↳ Il rapporto tra la temperatura dell'acqua e il tasso di evaporazione è stato studiato con il sensore di umidità. L'umidità relativa ottenuta all'interno di un depuratore solare che conteneva acqua a temperatura ambiente è stata misurata al 55%. Quando l'acqua è stata riscaldata a 45°C, l'umidità relativa è salita al 98% in pochi secondi.
- ↳ La quantità di radiazione riflessa da una superficie metallica nell'alambicco solare è stata misurata con il sensore di fiamma. Un vecchio ombrello coperto con un foglio di alluminio è stato molto efficace nel riflettere le radiazioni IR.

<Conclusione>

Gli studenti possono utilizzare la loro creatività e le loro capacità di pensiero di ordine superiore (HOTS, Higher-Order Thinking Skills) per progettare il depuratore solare più efficiente. Questa unità darà loro anche l'opportunità di sviluppare le loro capacità di pensiero critico e di problem-solving, il che è molto utile perché gli studenti possono utilizzarle quotidianamente. Impareranno alcuni fatti fisici chiave mediante l'autoapprendimento, osservando, sperimentando, testando e analizzando i loro risultati, invece di limitarsi a leggerli nei loro libri di fisica.

Essi svilupperanno anche capacità di pensiero computazionale mentre codificano i loro sensori. Faranno calcoli fisici (cioè i loro codici interagiranno con il mondo fisico). In tutta l'unità, seguiranno le diverse fasi del metodo scientifico sopra de-

scritto per sviluppare il miglior progetto di depuratore solare e poi costruirlo con materiali opportuni.

È importante che ogni studente sia coinvolto nei compiti del progetto. Alcuni possono essere svolti individualmente (ricerca di informazioni, raccolta delle idee iniziali sul progetto....) per garantire che ciò avvenga.

Tuttavia, l'ostacolo principale è che alcuni studenti potrebbero non avere le competenze di programmazione appropriate (per questo motivo abbiamo incluso alternative di programmazione a blocchi), conoscenze sufficienti per costruire i circuiti (le risorse aggiuntive^[7] sono molto utili a questo proposito) o potrebbero non capire come funzionano i sensori. Inoltre, il materiale potrebbe non essere disponibile in tutte le scuole, per cui potrebbe essere necessario acquistarlo.

Ulteriori attività di sviluppo:

- ↳ I dati raccolti dai sensori potrebbero essere memorizzati su una scheda SD per ulteriori analisi.
- ↳ Per visualizzare le misure è possibile utilizzare uno schermo LCD.
- ↳ Internet of Things (IoT): i dati raccolti possono essere inviati via Internet in tempo reale, in modo che siano disponibili al pubblico.
- ↳ Possono essere utilizzati sensori aggiuntivi, come CO₂ o altri sensori di gas serra, un sensore di conducibilità per controllare se l'acqua pulita è ancora salata, un sensore di pH per misurare il pH dell'acqua sporca e pulita, ecc.
- ↳ La salinità può essere misurata confrontandola con campioni di acqua di mare.
- ↳ Potrebbe essere aggiunto un metodo per disinfettare l'acqua raccolta.

Gli studenti possono utilizzare i depuratori solari per studiare l'effetto serra, la fotosintesi, la respirazione cellulare, i gas ideali, ecc. Molti sensori potrebbero essere utilizzati in altri progetti per misurare parametri fisici o chimici, ad esempio per monitorare l'inquinamento atmosferico o la qualità dell'acqua.

<Attività cooperativa>

Il solare è più efficiente nei paesi con insolazione maggiore? Gli studenti di diverse scuole europee potrebbero condividere i loro risultati, utilizzando una mappa online per la localizzazione. La salinità dei diversi luoghi può essere confrontata prelevando campioni di acqua di mare.

<Riferimenti>

- [1] www.arduino.cc/
- [2] <https://calliope.cc/en>
- [3] www.raspberrypi.org
- [4] www.arduino.cc/reference/en/
- [5] www.python.org/
- [6] <http://snap4arduino.rocks/>
- [7] Tutti i materiali aggiuntivi sono disponibili all'indirizzo www.science-on-stage.de/coding-materials.
- [8] <https://scratch.mit.edu/>
- [9] www.raspberrypi.org/documentation/usage/python/
- [10] <https://snap.berkeley.edu/>



Il funzionamento dell'acqua

<Autore> Lúcio Botelho

<Autore> Lílíana Fernandes

<Autore> Jorge Reis

<Info>

<Parole chiave> acqua, elaborazione delle immagini, acquisizione dati, microclima, robot

<Discipline> matematica, biologia, studi sociali, robotica, arti

<Età degli studenti> 6–10, 11–15 e 16–18

<Hardware> <livello facile> Calliope mini^[1], LEGO We Do 2.0^[2], piccoli bot di apprendimento^[3], WeeeMake^[4]

<livello medio> LEGO EV3^[2] con sensori a ultrasuoni e sensori di colore LEGO, oppure Anprino^[5] con Arduino^[6] e relativi sensori a ultrasuoni^[7] e sensori di colore^[8]

<livello avanzato> computer con accesso a Internet

<Linguaggi> Snap!^[9], Scratch^[10], WeeeCode^[4], Open Roberta^[11], LEGO Blocks^[2]

<Livello di difficoltà di programmazione> facile, medio, avanzato

<Sommario>

Questa unità è stata progettata per essere svolta in maniera verticale tra livelli diversi del ciclo di istruzione e per facilitare il lavoro collaborativo tra studenti di età differenti: dalla scuola primaria a quella secondaria di primo e secondo grado. In alternativa, ogni parte può essere insegnata a uno specifico livello di istruzione. Gli studenti, con questa unità didattica, potranno scoprire argomenti differenti tutti collegati al tema principale dell'acqua: partendo da un semplice approccio al pensiero computazionale, passando alla codifica in Scratch^[10], fino alla programmazione vera e propria di un robot oppure di una piccola casa ecologica.

<Introduzione concettuale>

Questo progetto riguarda l'acqua, il suo ruolo nella nostra vita e il nostro compito nel preservarla. È suddiviso in tre livelli (facile per gli studenti della scuola primaria, medio per gli studenti della scuola media e avanzato per gli studenti della scuola secondaria), questo progetto può essere impiegato per lo svolgimento in un contesto didattico di tipo collaborativo, anche in situazioni di verticalità tra diversi livelli scolastici e per svolgere attività interscolastiche.

<Cosa fanno gli studenti/insegnanti>

<Livello facile: Da dove viene l'acqua?>

Gli studenti sono invitati ad indagare sulla provenienza dell'acqua. L'insegnante all'inizio della lezione porrà delle semplici domande per stimolare l'interesse degli studenti e poi... l'avventura potrà iniziare! Gli studenti faranno ricerche, impareranno prima e condivideranno poi i loro risultati con i compagni di classe; contemporaneamente inizieranno a sviluppare

le loro capacità di pensiero computazionale mediante facili sfide per la programmazione di semplici robot.

Dopo aver terminato la ricerca, gli studenti iniziano a lavorare in piccoli gruppi alla realizzazione di alcuni semplici progetti; a tale scopo utilizzeranno le modalità demo dell'app WeDo 2.0^[12] e, in particolare, le attività legate all'acqua.

Successivamente gli studenti realizzano una semplice ma innovativa soluzione ecologica per il risparmio idrico, utilizzando qualsiasi mattoncino LEGO (o set) che desiderano.

Nell'esempio che segue, gli studenti hanno costruito un modellino di casa ecologica^[13] e l'hanno combinata con alcuni mattoncini extra e con il set WeDo 2.0. Successivamente hanno aggiunto un collettore di acqua piovana collegato ad un filtro (codificato con l'app LEGO) che, a sua volta, convoglia l'acqua all'interno dell'allevamento in modo che gli animali possono bere acqua fresca (🐾1).



🐾 1: Una casa ecologica

Allo stesso tempo, gli studenti, che lavorano ancora in piccoli gruppi, inizieranno a pianificare e progettare nuove configurazioni legate al tema dell'acqua, suddivise per piccoli bot didattici e che potrebbero essere programmati senza computer. Presentando le loro configurazioni (tappetini) agli altri studenti, li motiveranno a codificare e allo stesso tempo a conoscere l'acqua. Gli studenti possono utilizzare vari robot di apprendimento a basso costo per svolgere il compito.^[3]

Le istruzioni complete su come stampare i tappetini sono disponibili online.^[14]

Troverete anche i riferimenti ad alcune attività didattiche aggiuntive descritte passo-passo.^[14]

<Livello medio: Costruire un robot per la pulizia delle dighe d'acqua>

Il progetto di pulizia della diga riguarda un robot che viaggia all'interno di un serbatoio d'acqua creato da una diga, rileva ed asporta i rifiuti solidi.

Questo progetto ha due versioni, che utilizzano due robot diversi. La versione LEGO [2] utilizza il kit didattico EV3 LEGO Education; il robot Anprino [5], stampato con una stampante 3D e poi assemblato, è la base della versione Arduino [3]. Il microcontrollore Arduino [6] e il set dei suoi accessori sono attaccati ad Anprino.



2: La versione LEGO

3: La versione Anprino

E' consigliabile iniziare a costruire il modello del serbatoio dell'acqua utilizzando carta o cartone. Dovrebbe misurare circa 2 m x 1 m ed essere dipinto di blu per simulare l'acqua in esso contenuta. Costruire le rive utilizzando cartone robusto per delimitare lo spazio di movimento del robot, e alcuni pezzi di cartone nero per simulare i rifiuti.



4: Modello del serbatoio della diga

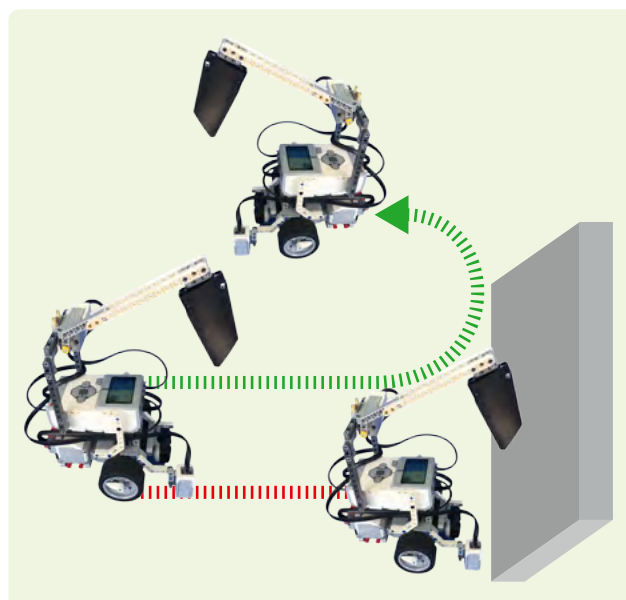
Sensore a ultrasuoni

Un sensore a ultrasuoni [7] genera onde sonore per rilevare e misurare la distanza dagli oggetti. Può anche inviare onde sonore per funzionare come un sonar o ricevere un'onda sonora che avvia una modalità di programmazione.

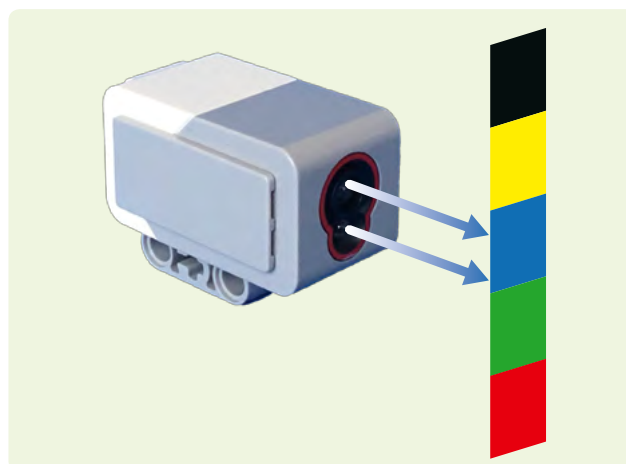
Utilizzando il sensore a ultrasuoni, il robot è in grado di rilevare gli ostacoli e, a seconda delle istruzioni del programma, reagire in modi diversi: il robot potrebbe essere programmato per fermarsi o cambiare direzione, ad esempio. Nel modello di diga, gli ostacoli sono i margini della diga, realizzati in cartone.



5: Un sensore LEGO ad ultrasuoni



6: Azioni di arresto del robot e di cambiamento di direzione



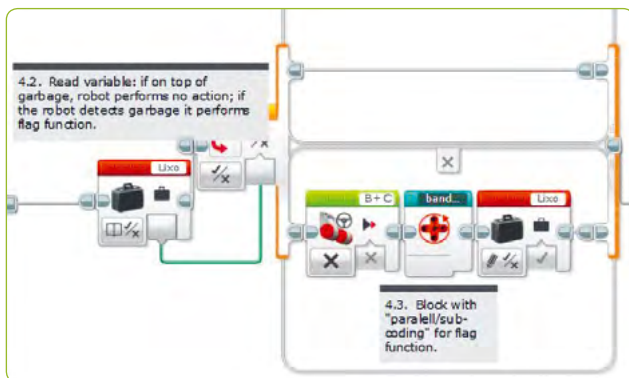
7: Esempio di un sensore di lettura dei colori: distingue i colori leggendo il loro codice RGB.

Sensore di colore

Il sensore di colore^[8] è in grado di rilevare colori diversi e l'assenza di luce. Funziona anche come sensore di luce, rilevando le diverse intensità della luce. Gli studenti possono costruire linee di diverso colore che devono essere seguite dal robot.

Realizzazione di un programma, utilizzando i blocchi di programmazione LEGO

Gli studenti devono costruire diversi modelli per simulare diversi tipi di rifiuti, come quelli domestici, industriali, turistici, biologici, organici, legati al turismo, ecc. L'obiettivo principale è che gli studenti diventino consapevoli dell'inquinamento dei fiumi e delle dighe. Gli studenti devono simulare un rilevatore di rifiuti da montare su un'imbarcazione e poi, in un secondo momento, progettare e costruire l'imbarcazione stessa necessaria per la raccolta dei rifiuti.



8: Un estratto dalla programmazione LEGO; il diagramma di programmazione completo è disponibile online^[14]

Il robot deve riprodurre un suono particolare per ogni tipo di rifiuti rilevato. Per raggiungere questo obiettivo, utilizziamo il sensore di colore e specifiche "macchie" di colore che simulano il tipo specifico di rifiuto.

Gli studenti possono consultare materiale bibliografico differenziato (pubblicato da enti del settore) per cercare di capire la qualità e composizione effettiva dei rifiuti raccolti nel proprio territorio e quindi progettare e costruire macchine diverse sulla base di tali dati.

Gli studenti possono recarsi in gita presso fiumi e dighe per esaminare la qualità dell'acqua e l'entità dell'inquinamento. Successivamente devono poi simulare e riprodurre i dati di queste osservazioni nei modelli che costruiscono. Dopo aver usato il robot, gli studenti dovrebbero analizzare i dati così acquisiti per verificare che siano coerenti con quelli reali (9).

Quando gli studenti hanno raccolto una quantità sufficiente di dati, devono presentare le loro ricerche alla classe. L'obiettivo è che essi sviluppino le loro capacità di pensiero critico, di indagine e di programmazione. Quando gli studenti guardano a ciò che sta accadendo nei nostri corsi d'acqua, potranno anche rendersi conto delle reali conseguenze della cattiva gestione delle questioni ambientali. Per potere comprendere pienamente, dovrebbero anche aver acquisito le competenze ambientali necessarie per capire come intervenire nella comunità di appartenenza.

Questa fase potrebbe consistere anche nell'illustrare alle persone la necessità di prevenire comportamenti che danneggiano le risorse idriche e l'ambiente più in generale. Dovrebbero anche essere in grado di progettare e illustrare delle soluzioni quando individuano i problemi. L'obiettivo generale è aumentare la loro partecipazione civica e il senso di responsabilità ambientale all'interno della propria comunità.

Nota bene: i nostri studenti hanno già costruito e testato la versione LEGO e sono ancora in fase di miglioramento della versione Arduino. Il codice completo utilizzato per programmare l'Arduino è disponibile online.^[14]

<Livello avanzato: Programmazione di giochi educativi per l'educazione ambientale>

L'obiettivo principale è che gli studenti diventino consapevoli dell'inquinamento idrico. Gli studenti utilizzeranno Scratch^[10] per programmare giochi che motivano i giocatori a contribuire a preservare e proteggere le risorse idriche e quindi incoraggiare nelle persone comportamenti di rispetto.

9: Tabella per la scansione dei rifiuti; realizzata dalle squadre di pulizia del club ambientale della scuola. Contiene dati relativi a due diverse escursioni e le quantità di tipi di rifiuti raccolti in ciascuna di esse.

Data	Tipo di rifiuti					Area Pulita
	Domestico	Industriale	indifferenziato	Organico	Altri	
Escursione Aprile 2018	3,450 kg			32 kg	8 kg	100 m ²
Escursione Maggio 2018	0,730 kg			6 kg		100 m ²

Il nostro primo gioco simula un pesciolino nell'oceano. Il pesce deve nutrirsi ed allo stesso tempo evitare altre creature marine (squali e granchi) ma anche la caduta di rifiuti (bicchieri, lattine, ecc.). Più il pesce mangia, più diventa grande, più punti vengono guadagnati dal giocatore durante il processo.

Il pesce non deve scontrarsi con gli scarti e nemmeno con gli altri pesci, altrimenti si fa male e riceve una benda. Quando il pesce ha tre bende, il gioco è finito. Questo gioco è divertente e sensibilizza bambini e adulti sul problema della crescente quantità di rifiuti presenti nei nostri corsi d'acqua (oceani, fiumi, ecc.).

Il secondo gioco è basato su un noto videogioco in cui una rana deve attraversare una strada. In questo caso però, il personaggio deve attraversare un fiume (usando dei tronchi galleggianti in un corso d'acqua che scorre velocemente) ed evitare sia

rifiuti che altri animali (pipistrelli e serpenti). Esso può anche mangiare mosche con cui guadagna punti extra.

In questo gioco, ci sono quattro diversi scenari; uno solo viene selezionato casualmente all'inizio di ogni round. La rana (sprite) ha tre vite, terminata la terza il gioco finisce.

La sezione seguente contiene dettagli sul programma.

10 mostra la parte del programma che controlla il movimento di alcuni dei nemici nelle varie modalità di gioco. Nell'esempio visualizzato, il nemico scompare quando tocca uno qualsiasi dei bordi. Finché non tocca i bordi, esso ripete lo stesso movimento, che aumenta anche di velocità di un fattore 0,04 all'aumentare del punteggio del giocatore. Questo è un accorgimento utile a rendere il gioco un po' più impegnativo all'aumentare del punteggio e del livello di difficoltà.

Inizio del gioco: scegliete una delle tre modalità di gioco (11). Attualmente sono stati sviluppati e disponibili soltanto due giochi e gli studenti ne stanno sviluppando un terzo chiamato modalità di gioco 2.

Ad esempio, la modalità di gioco 2 potrebbe svolgersi in uno stagno dove le anatre devono prendere del cibo.

Le anatre mangiano regolarmente piccoli pesci, uova di pesce, lumache, vermi, molluschi, gamberi, erba, foglie, piante acquatiche.

```

when I start as a clone
  go to AI Director
  show
  if Gamemode = 1 then
    if x position > 140 then
      point in direction -90
      repeat 10
        change x by Score * -0.04 + -6
      repeat until touching edge ?
        change x by Score * -0.04 + -6
      repeat 8
        change x by Score * -0.04 + -6
      delete this clone
    else
      point in direction 90
      repeat 10
        change x by Score * 0.04 + 6
      repeat until touching edge ?
        change x by Score * 0.04 + 6
      repeat 8
        change x by Score * 0.04 + 6
      delete this clone
  
```

10: Programma Scratch controllo del nemico

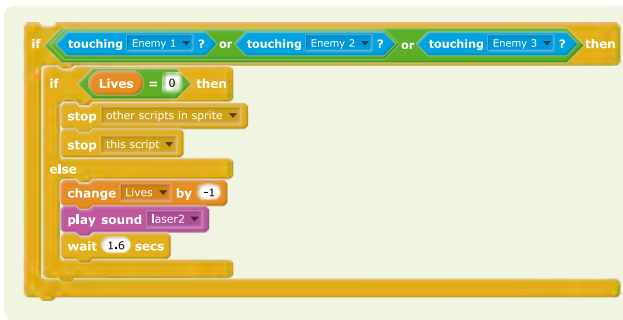
```

when I receive Load Game
  wait 0.8 secs
  if Gamemode = 1 then
    go to x: 0 y: 0
    switch costume to Fizz
    set size to 27 %
  if Gamemode = 2 then
    go to x: 0 y: -4
    switch costume to Nasus
    set size to 24 %
  if Gamemode = 3 then
    go to x: 0 y: -160
    switch costume to frog
    set size to 70 %
    point in direction 90
  show
  go back 5 layers
  
```

11: Programma Scratch di inizio del gioco

tiche, rane, salamandre. Inoltre, le anatre devono cercare di evitare sia le altre anatre che i rifiuti presenti nello stagno (oppure nei livelli avanzati, dei personaggi in veste di bracconieri casuali).

Se il pesce tocca uno qualsiasi dei nemici (1, 2 o 3), perde una vita e viene emesso un suono. Se il giocatore perde tutte le vite disponibili, il gioco è finito, tutti gli script vengono fermati (📄 12).



📄 12: Programma Scratch dei nemici 1 e 3

Si può utilizzare lo stesso codice per entrambe le versioni del gioco. Lo stesso sprite cambia il suo costume da “Squalo” a “Bat”. Gli studenti possono migliorare il programma con soluzioni di codice più efficaci.

Questo è possibile perché i giochi hanno obiettivi e regole simili:

- ↳ evitare i nemici
- ↳ prendere cibo / mosche
- ↳ avere 3 vite disponibili
- ↳ guadagnare punti (il pesce mangiando il cibo, la rana mangiando le mosche e raggiungendo un nuovo scenario)

Gli studenti utilizzeranno i cloni dello sprite nemico in modo da poter far comparire lo stesso sprite da diverse direzioni e assumere comportamenti (direzioni) diversi nel gioco.

Il codice del programma è interamente disponibile per il download.^[14]

<Conclusione>

In questa attività, gli studenti lavorano collaborando con i loro coetanei e la comunità in cui vivono. Possono imparare e condividere le loro conoscenze sui temi e problemi delle risorse idriche quali ad esempio: il ciclo idrologico, la scarsità d'acqua, l'inquinamento, ecc. Essi sviluppano anche risorse originali per monitorare, risparmiare e proteggere l'acqua. Gli studenti adottano metodi e strumenti investigativi e sviluppano competenze sia di programmazione che di robotica. Si consiglia di svolgere questa attività con metodo collaborativo tra pari con gli studenti più grandi che rivestono il ruolo di mentore e sostengono quelli più giovani; in questo modo tutti si motivano maggiormente e questo fatto contribuisce enormemente al successo del progetto.

Alla fine dell'anno scolastico, abbiamo notato che gli studenti non solo hanno migliorato le loro capacità di programmazione, ma sono anche diventati più consapevoli dei problemi legati alle risorse idriche, dei pericoli per gli animali e le piante che vivono in ecosistemi che dipendono strettamente dalla qualità dell'acqua pulita.

Non è facile realizzare più giochi basati su un unico modello di programmazione. I giochi infatti devono avere alcune somiglianze in modo tale che il codice di uno non possa essere adattato per realizzare altri giochi o versioni. Tuttavia, questo metodo viene usato in questa attività per risparmiare sulle risorse di programmazione.

Questa unità didattica, può anche essere utilizzata per sfidare gli studenti a sviluppare altre idee e progetti sul tema del risparmio idrico e contribuire così a migliorare i comportamenti ambientali nella comunità di appartenenza, riducendo così l'impronta di carbonio degli studenti e auspicabilmente dell'intera comunità.

<Attività cooperativa>

‘Science on Stage significa condividere le risorse tra gli insegnanti!’ Un altro risultato, conseguito da questo progetto, è stato rafforzare la comunità di insegnanti e di condividere risorse e idee, ciò ha contribuito a migliorare l'apprendimento degli studenti in tutta Europa. Condividere e collaborare è il modo migliore per sviluppare ulteriormente questi progetti e migliorare nell'insegnamento.

<Riferimenti>

- [1] <https://calliope.cc/en>
- [2] <https://education.lego.com>
- [3] Possibili bot: Bee Bots da tts, DOC da Clementoni, Jack da Imaginarium
- [4] www.weemake.com/
- [5] Anprino è un robot sviluppato dall'Associazione Nazionale Portoghese degli Insegnanti di Tecnologia dell'Informazione (ANPRI); informazioni e file di stampa 3D www.anpri.pt/anprino/index.php/anprino-luis [29/11/2018].
- [6] www.arduino.cc/
- [7] Abbiamo usato il sensore a ultrasuoni HC-SR04.
- [8] Abbiamo utilizzato il sensore di luce BE15000624.
- [9] <https://snap.berkeley.edu/>
- [10] <https://scratch.mit.edu/>
- [11] <https://lab.open-roberta.org/>
- [12] <https://education.lego.com/en-us/downloads/wedo-2/software> [29/11/2018]
- [13] LEGO SET 31068
- [14] Tutti i materiali aggiuntivi sono disponibili all'indirizzo www.science-on-stage.de/coding-materials.

SPDV

Salvavita per Piante Durante le Vacanze

<Autore> Andreas Meier

<Autore> Sonja van der Byl



<Info>

<Parole chiave> > impianto di irrigazione automatizzata, con l'utilizzo di un microcontrollore per gestire una pompa dell'acqua

<Discipline> informatica, scienze naturali, tecnologia

<Età degli studenti> 10–14

<Hardware> computer (uno per studente se possibile), Calliope mini^[1] con sensore di umidità e temperatura (uno per gruppo)

<Linguaggi> Scratch^[2] (online o offline), editor per Calliope^[3] (online)

<Livello di difficoltà di programmazione> facile

<Sommario>

Le piante in vaso negli edifici scolastici spesso muoiono durante la pausa estiva perché nessuno se ne prende cura. Per questo motivo abbiamo bisogno di un salvavita per le piante (Virtual Plant Saver, VPLS), nel periodo delle vacanze. In questa unità didattica svilupperemo un salvavita virtuale e reale per le piante della scuola.

<Introduzione concettuale>

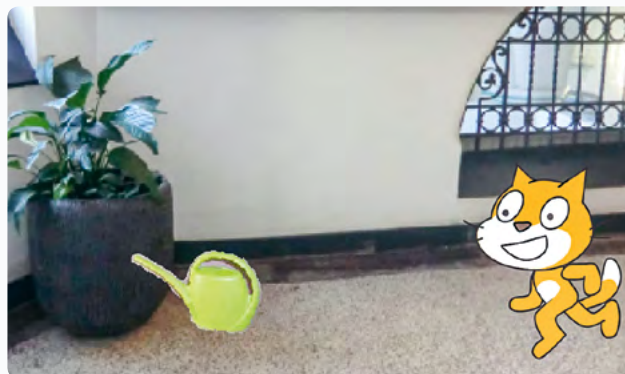
Il progetto è adatto a tutti gli argomenti STEM perché il livello di programmazione richiesto è di base.

Nella prima sezione, l'irrigazione delle piante ad intervalli regolari viene realizzata in termini virtuali. Ciò comprenderà varie strutture di controllo nell'informatica, come la programmazione ad oggetti, i loop e le condizioni, nonché l'uso di variabili. Gli studenti scriveranno per la prima volta codici di programmazione di base e lavoreranno attraverso un tutorial online in 'Getting Started with Scratch'^[4] (durata: 45 minuti).

Object Oriented Programming (OOP) è un tipo di programmazione che conferisce ad un oggetto proprietà (attributi) e capacità (metodi). Nel nostro caso, questi oggetti sono un gatto chiamato 'Sprite', un innaffiatoio e uno stage. Ogni oggetto appartiene ad una classe. Tutti gli oggetti della stessa classe hanno le stesse proprietà e capacità. Si può interpretare una classe come un progetto e un oggetto come un esempio, cioè una realizzazione concreta del progetto. In Scratch, le cifre sono esempi della classe 'Sprite' o, in breve, 'sprites'. Il gatto chiamato 'Sprite' è uno sprite, cioè un esempio della classe 'Sprite'.

Uno degli attributi di uno sprite è il suo "costume"; in questo caso, è l'immagine di un gatto. Un altro esempio della classe 'Sprite' potrebbe avere l'immagine di un essere umano come

costume ed essere chiamato Gunther. Il gatto 'Sprite' e l'umano 'Gunther' sono entrambi oggetti (esempi) della classe 'Sprite' o in breve: entrambi sono sprite. Ci sono solo due classi in Scratch: lo stage e gli sprite. In questo progetto, abbiamo due sprite (il gatto e l'innaffiatoio) e lo stage [1].



© 1: Il salvavita virtuale delle piante per le vacanze

La seconda parte del progetto può essere effettuata dopo il completamento della prima parte, oppure può essere realizzata anche autonomamente se gli studenti conoscono le strutture di controllo sopra menzionate e hanno già acquisito una prima esperienza di programmazione con Calliope mini^[1]. I sensori del microcontrollore che controllano la valvola di una pompa dell'acqua saranno utilizzati al posto delle variabili.

<Cosa fanno gli studenti/insegnanti>

Tutti i materiali e i fogli di lavoro richiesti sono disponibili per il download.^[5]

<Parte 1: Irrigazione virtuale regolata dell'impianto>

Fase 1: codificare un programma con una sola variabile: il tempo; imparare l'utilizzo delle condizioni e del loop (durata: 180 minuti).

Dopo aver analizzato il problema ("Come potrebbe funzionare un salvavita per piante da utilizzare nel periodo delle vacanze?"), verrà chiesto agli studenti di pensare alla struttura di base del programma. Poi prenderanno appunti sotto forma di una "ricetta" (algoritmo) e discuteranno le idee degli altri compagni di scuola. Solo queste fasi, gli studenti si accordano su una struttura di base comune per il programma (vedi Foglio di lavoro 1^[5]).

Questa fase aiuterà gli studenti a pensare alle strutture di base del programma che vogliono codificare. Le parole chiave "elenco di istruzioni", "ciclo" e "condizione" sono derivate dal contesto.

Ora gli studenti realizzano il programma in Scratch^[2] assemblando i singoli componenti^[5] forniti dal programma di lavoro.

Gli studenti, durante le attività, impareranno di più di Scratch e dei seguenti aspetti:

- ↳ programmazione orientata all'oggetto (ogni figura ha il proprio script, anche lo stage)
- ↳ struttura (Come si presenta la struttura di una condizione o di un ciclo in Scratch?)
- ↳ script/costume/suoni possono essere assegnati ad ogni singola figura.

Gli studenti impareranno di più sulla struttura di base del programma di irrigazione delle piante mentre pensano a come ordinare al meglio le singole parti del codice per far funzionare il programma. È molto importante decidere quali istruzioni devono essere all'interno del ciclo di conteggio (2 & 3); questo può essere fatto per tentativi ed errori.

2

3

Sulla base del programma assemblato nella sezione precedente, gli studenti saranno ora in grado di creare il proprio programma in cui il gatto 'Sprite' si sposta verso le piante e le annaffia a seconda della variabile tempo. L'unico file che verrà consegnato agli studenti è lo stage per la scena di partenza.^[5]

Gli studenti saranno incoraggiati a studiare il linguaggio di programmazione in modo autonomo per testare le proprie

idee ed essere creativi. È importante che gli studenti siano in grado di utilizzare il linguaggio di programmazione richiesto con fiducia (in base al loro rispettivo livello di conoscenza); in questo modo sarà più piacevole per loro.

Fase 2: Scrivere un programma che incorpora le variabili "livello dell'acqua" e "temperatura" (tempo richiesto circa 270 minuti).

Come introduzione alla seconda fase dell'unità didattica, gli studenti penseranno ad altri fattori che determinano la frequenza con cui una pianta da appartamento deve essere annaffiata. La "temperatura ambiente" e il "livello dell'acqua" nel contenitore dell'impianto giocheranno certamente un ruolo importante di variabili. (vedi Foglio di lavoro 2^[5]).

Gli studenti riceveranno un programma funzionante in cui gli script per il gatto 'Sprite' e l'innaffiatore sono quasi gli stessi di prima.

Tuttavia, ci sarà un nuovo script per la fase che controlla la variabile "livello dell'acqua" al posto del timer precedente: il gatto innaffierà la pianta solo una volta. Gli studenti saranno invitati a pensare a come la variabile "livello dell'acqua" può essere definita da un lato e come essa può controllare l'attività del gatto. D'altro canto, avranno il compito di risolvere il problema in modo che l'impianto venga irrigato una sola volta. Se necessario, è disponibile un file di aiuto.^[5]

Utilizzando una sola variabile, il programma rimarrà chiaramente strutturato. Potrebbe essere troppo impegnativo per un programmatore principiante coordinare più variabili contemporaneamente.

La struttura ad anello utilizzata è più complessa della precedente in quanto collegata ad una condizione (4). Gli studenti dovranno riflettere attentamente su cosa deve essere ripetuto, con quale frequenza e in quali condizioni.

4

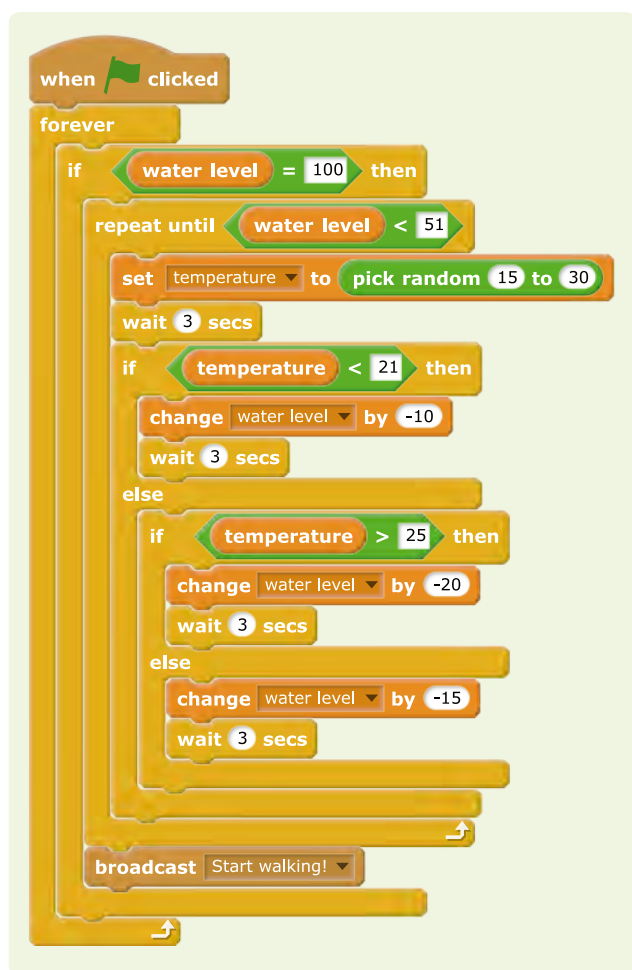
La capacità di strutturare è di fondamentale importanza nell'apprendimento di qualsiasi linguaggio di programmazione, in questa attività essa sarà insegnata in modo molto divertente e gli studenti saranno in grado di fare prove e tentativi senza avere troppo timore.

Gli studenti più veloci avranno anche l'opportunità di fare modifiche al programma e provarle al termine di questa fase di lavoro.

Nella fase successiva, la variabile "temperatura" ha un ruolo importante nel programma. Il suo valore in questo caso è determinato da un generatore di numeri casuali, che fornisce un numero compreso tra 15 °C e 30 °C. Il livello dell'acqua nel contenitore dell'impianto cambia a seconda di tale valore (Foglio di lavoro 3^[5]).

Poiché la struttura del "nuovo" programma di irrigazione è piuttosto complessa, il programma dato agli studenti sotto forma di moduli, dovrebbe essere riassembleato in modo da avere un programma funzionante^[5].

A questo punto sono introdotti il generatore di numeri casuali e le condizioni doppie. Inoltre, gli studenti potranno lavorare a più riprese con le variabili e le condizioni di interrogazione e approfondire la loro comprensione. Anche in questo caso, gli studenti dovranno riflettere attentamente su quali istruzioni devono essere ripetute, quante volte e in quali condizioni [5].



© 5

Come nella fase precedente, gli studenti avranno la possibilità di personalizzare il programma risultante agendo in base alle loro idee e alla loro capacità di programmazione.

Alla fine della prima parte del progetto, il Salvavita per piante è stato realizzato in termini virtuali, è necessario presentare ai ragazzi una grande varietà di risultati di programmazione, verosimilmente ottenuti da loro, per valutare le idee degli studenti e la loro bontà.

<Parte 2: Irrigazione regolare di un impianto controllato da un microcontrollore>.

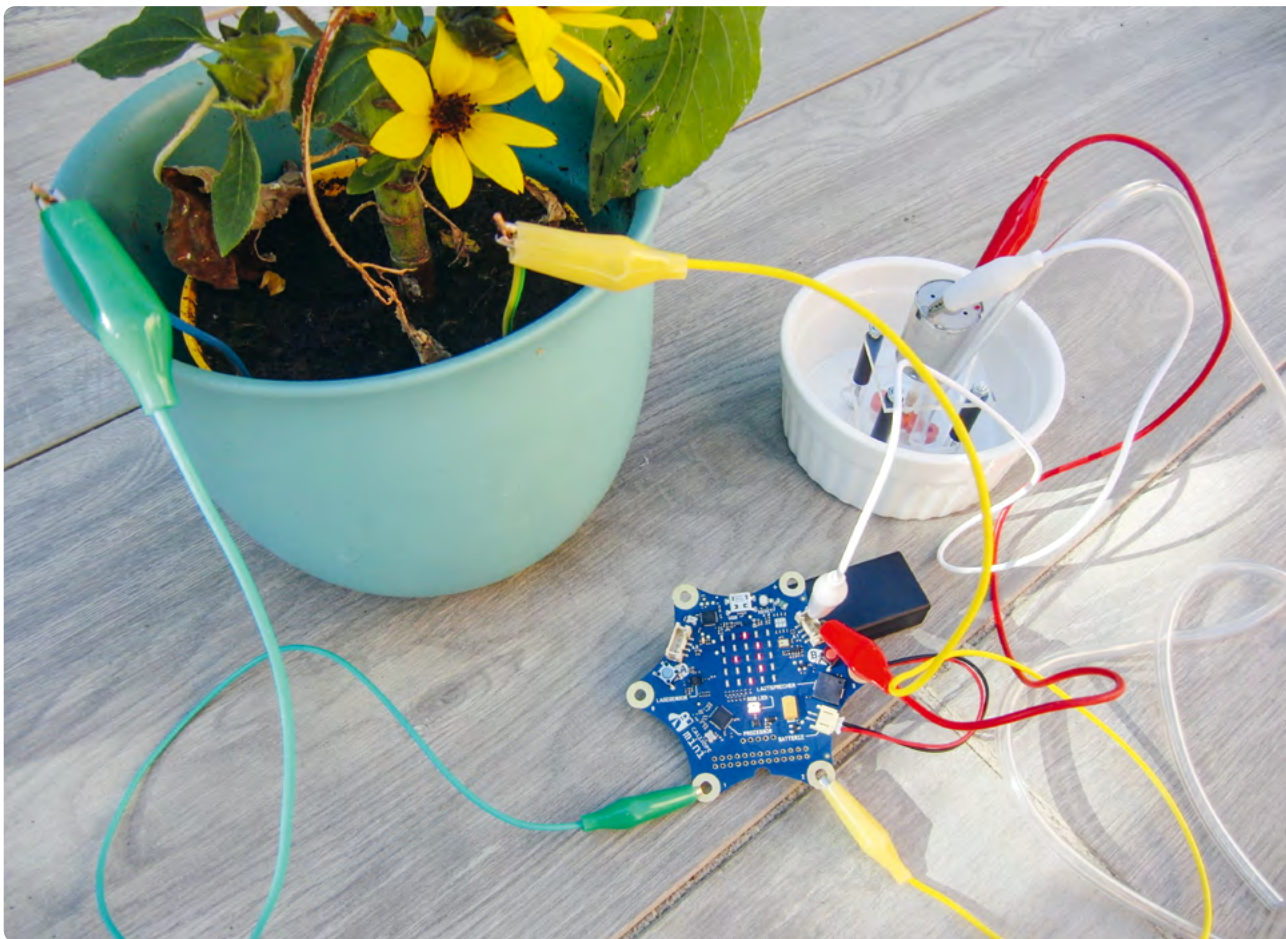
Durante il progetto, è apparso subito chiaro che alcuni studenti non erano soddisfatti della soluzione virtuale al problema dell'irrigazione. Hanno chiesto come innaffiare piante vere con l'aiuto del loro programma. Il programma di irrigazione virtuale è relativamente facile da trasferire ad un Salvavita reale utilizzando un microcontrollore, soprattutto perché molti di questi mini-computer possono essere programmati anche con Scratch^[2] o un'applicazione simile. Nel nostro progetto utilizziamo il Calliope mini^[4], un microcontrollore simile al BBC micro:bit^[6], che contiene anche "funzioni plug-and-play" di facile utilizzo come i sensori tattili e le connessioni per il motore. Tuttavia, il Salvavita può essere controllato anche con tutti gli altri microcontrollori comunemente usati nelle scuole, come LEGO EV3, LEGO NXT, Arduino, Raspberry Pi, Teensy, ecc.^[5]. Per esempio programmare il Calliope mini risulta un'attività semplice, poiché è sufficiente collegarlo al computer tramite un cavo USB e le attività di programmazione sono immediate. Open Roberta Lab^[7], che supporta vari microcontrollori (sono disponibili vari editor^[5]), è un'interfaccia di programmazione adatta a questo tipo di attività. L'interfaccia di programmazione è disponibile in diverse lingue ed è possibile cambiare la lingua cliccando sull'icona a "globo", dopo aver selezionato un microcontrollore, nel nostro caso il Calliope mini.

Una versione semplice del Salvavita per piante potrebbe funzionare come segue:

1. L'umidità del suolo viene costantemente misurata.
2. Se il terreno è troppo secco, una certa quantità di acqua viene pompata fino a quando il terreno è sufficientemente umido.

Pertanto, il microcontrollore deve essere in grado di misurare l'umidità del terreno e controllare il motore di una pompa dell'acqua.

Il Calliope mini ha quattro sensori di contatto. La fisica alla base del loro funzionamento risiede nel fatto che essi misurano differenti conducibilità elettriche tra i punti di connessione. Poiché l'acqua conduce l'elettricità, il suolo umido ha una con-



© 6: VPLS controllato da un Calliope mini

ducibilità maggiore rispetto al suolo secco. Si usano semplicemente due fili di rame come sensori, che poi si mettono nel vaso di fiori ad una certa distanza l'uno dall'altro. Vengono poi collegati con morsetti a coccodrillo al Calliope mini con i contatti ai connettori [-, P1]. Il valore di uscita del sensore P1 (pin analogico) sarà compreso tra 0 e 1023. Se la conducibilità scende sotto un certo valore, si attiva la pompa che innaffia l'impianto (vedi ©6).

Un componente essenziale della pompa è un piccolo motore elettrico, che viene collegato al Calliope mini direttamente oppure tramite un driver motore aggiuntivo, a seconda del livello di potenza richiesta. Nel menu Action/Move sono disponibili due porte motore (A, B). La quantità d'acqua della pompa può essere regolata modificando il valore 'speed %'.

Naturalmente, per costruire la pompa e l'azionamento del motore è necessario un certo livello di esperienza. Nel materiale disponibile online sono forniti i manuali di costruzione necessari. Il costo della pompa è di pochi euro.

Mentre lavorano con il vero Salvavita, gli studenti si porranno numerose domande su come ottimizzarlo, che consentiranno

loro di approfondire le loro conoscenze sulla problematica, ad esempio:

- ↳ Le nostre piante hanno bisogno di molta acqua? Quanto deve essere grande il serbatoio?
- ↳ Qual è il momento migliore per innaffiare le piante? E' meglio innaffiare molto solo una volta al giorno oppure di meno, ma più volte al giorno?
- ↳ A quale profondità devono essere collocati i sensori di umidità nel terreno per fornire misurazioni ottimali? Qual è la distanza ideale tra i sensori?
- ↳ Quanto dura l'alimentazione elettrica del Salvavita per piante? È possibile aumentare l'efficienza energetica della pompa in modo che il Salvavita funzioni durante l'intera pausa scolastica?

Come si può vedere, il progetto Salvavita offre agli studenti diversi approcci provenienti dai campi della biologia e della fisica, anche per futuri sviluppi o progetti. Un'altra opzione interessante sarebbe quella di collegare il Salvavita a Internet per il monitoraggio online (Internet of Things). Anche se questo va ben oltre le nostre intenzioni, ciò dimostra cosa si può ottenere da una esigenza molto semplice come ad esempio quella di innaffiare le piante.

<Trasferibilità ad altri linguaggi di programmazione>

Il progetto può essere facilmente portato nell'ambiente Snap! 8], che è un ulteriore sviluppo di Scratch^[2]. Gli esempi di programmazione sono forniti online^[5]. Questi due linguaggi sono particolarmente adatti per un progetto didattico rivolto ai programmatori principianti, perché sono facili da capire e incoraggiano gli studenti a provare le proprie idee utilizzando la modalità veloce di "drag and drop".

<Conclusione>

Con questo progetto didattico gli studenti a digiuno di programmazione hanno l'occasione di imparare le basi di un linguaggio di programmazione e di applicarle a una situazione di vita reale. L'attenzione qui non è focalizzata sull'apprendimento della sintassi e del vocabolario di un linguaggio di programmazione, ma piuttosto sulla sperimentazione del funzionamento di alcune strutture logiche: "Come funziona e perché?"

Questa attività sperimentale mette gli studenti nelle condizioni di commettere, molto probabilmente, degli errori che sono però essenziali per costringerli alla comprensione profonda del linguaggio di programmazione.

Consegnare subito agli studenti la struttura del programma significa dargli un elemento di sicurezza; successivamente possono limitarsi ad assemblare le varie parti. Al contrario, se la struttura non viene consegnata inizialmente, allora gli studenti (magari quelli meno riflessivi) saranno spinti a compiere uno sforzo di riflessione e creatività.

Dal progetto sono emerse alcune idee, ad esempio: costruire un "vero" robot di irrigazione o sviluppare un gioco di irrigazione. In ogni caso, gli studenti hanno acquisito un'esperienza di programmazione iniziale positiva che, si spera, avrà su di loro un effetto permanente.

Le tempistiche dedicate al progetto nella nostra scuola a volte sono state di difficile gestione. Abbiamo avuto la disponibilità di slot di soli 45 minuti per ogni lezione svolta in questo progetto. La sola parte organizzativa delle lezioni (accesso al computer, apertura dei file, salvataggio dei file, disconnettersi dal computer) ha richiesto 15 minuti per periodo, quindi spesso non c'è stato il tempo sufficiente per la programmazione e il lavoro vero e proprio. È possibile però richiedere l'accesso a Scratch^[2] in modalità "insegnante", che vi permetterà di creare una classe virtuale e memorizzare il materiale prodotto.

<Attività cooperativa>

Poiché il Salvavita è un'attività introduttiva alla programmazione, le possibilità di cooperazione sono molto limitate. Tuttavia non appena il progetto viene trasferito alla gestione di un microcontrollore, la cooperazione tra studenti risulta utile, poiché il grado di difficoltà del problema aumenta utilizzando i componenti aggiuntivi (sensori, pompa). Ad esempio, gli studenti più anziani possono aiutare a costruire la pompa, oppure scuole di diversi paesi potrebbero lavorare insieme al progetto Salvavita e confrontare i loro risultati e le loro soluzioni.

Si potrebbe creare una banca dati comune per diverse piante per adattare il Salvavita alle caratteristiche individuali di una determinata specie vegetale. Se il dispositivo è collegato ad Internet, le scuole potrebbero adottare piante di un'altra scuola e farsi anche carico dell'irrigazione da remoto.

<Riferimenti>

- [1] www.calliope.cc/en
- [2] www.scratch.mit.edu/
- [3] www.calliope.cc/en/los-geht-s/editor
- [4] https://scratch.mit.edu/projects/editor/?tip_bar=getStarted [29/11/2018]
- [5] Tutti i materiali aggiuntivi sono disponibili all'indirizzo www.science-on-stage.de/coding-materials.
- [6] www.microbit.co.uk/home
- [7] <https://lab.open-roberta.org>
- [8] <https://snap.berkeley.edu/>



Il guanto magico

<Autore> Annamária Lőkös

<Autore> Camelia Ioana Rațiu

MAGIC



<Info>

<Parole chiave> esperimento, ambiente, temperatura, umidità, luminosità, campo magnetico

<Discipline> fisica, chimica, biologia, ecologia, informatica

<Età degli studenti> 10–18

<Livello 1> per la scuola primaria (età: 10–11 anni) e secondaria (scuola media, età: 12–15 anni)

<Livello 2> per la scuola secondaria (scuola superiore, età: 15–18 anni)

<Hardware> Arduino UNO^[1], sensori compatibili con Arduino (es. sensore di luce, sensore di temperatura, sensore di campo magnetico, sensore di umidità, sensore di gas), schermo per tasti LCD, cavi di collegamento, batteria esterna

<Linguaggi> C^[2], Arduino 1.8.5^[3], Snap!^[4]

<Livello di difficoltà di programmazione> medio

<Sommario>

I giovani sono appassionati di tecnologia, quindi una lezione che combina la scienza con l'informatica è destinata ad avere successo. Seguendo questa unità gli studenti costruiranno e utilizzeranno un guanto dotato di un sensore diverso per ogni dito. Questo permetterà loro di effettuare esperimenti diversi collegando solo il sensore necessario.

<Introduzione concettuale>

Il vantaggio di utilizzare un dispositivo (il guanto) dotato di più sensori per misure diverse è duplice. Gli studenti delle scuole primarie e secondarie possono usare il guanto "magico" per misurare la temperatura, la luminosità, l'umidità, la presenza di un campo magnetico, l'intensità di un suono, ecc. Tutto quello che devono fare è selezionare il sensore desiderato, dopodiché saranno pronti per trovare vari utilizzi del guanto in diversi campi di studio e materie scolastiche. Il guanto può essere utilizzato anche in ambiente aperto, in quanto è alimentato da una batteria; questo dà agli studenti la possibilità di fare misure anche al di fuori del laboratorio. Gli studenti delle scuole superiori possono costruire da soli il guanto per effettuare determinati rilevamenti. Conoscendo le nozioni teoriche delle varie discipline (fisica, chimica, biologia, ecologia), gli studenti hanno la grande opportunità di sperimentare praticamente con indagini in diversi campi.

All'insegnante è richiesto di presentare i concetti di base per la codifica di un programma in C^[2] o in qualsiasi altro linguaggio di programmazione supportato da Arduino, incluso Snap!^[4], affinché gli studenti siano in grado di scrivere codice per la piattaforma Arduino^[1]. Per acquisire una conoscenza di base

del C, se viene scelto come linguaggio di programmazione, gli studenti possono utilizzare i tutorial su Internet. Questo li aiuterà a capire meglio come scrivere il codice e aumenterà anche la loro fiducia, data la facilità di scrittura del programma.

<Cosa fanno gli studenti/insegnanti>

<Livello 1>

Il guanto ha un display LCD con pulsanti. Gli studenti indossano il guanto e selezionano il sensore desiderato con i tasti UP/DOWN; quindi premono il tasto SEL per avviare la misurazione. Sullo schermo viene visualizzato il valore misurato. Gli studenti possono ripetere le misure in qualsiasi momento. Per tornare alla schermata iniziale, premere il pulsante "BACK". Come esempio, si può vedere la determinazione dei poli di un magnete @1a–1c.



@ 1a-c: La determinazione dei poli di un magnete

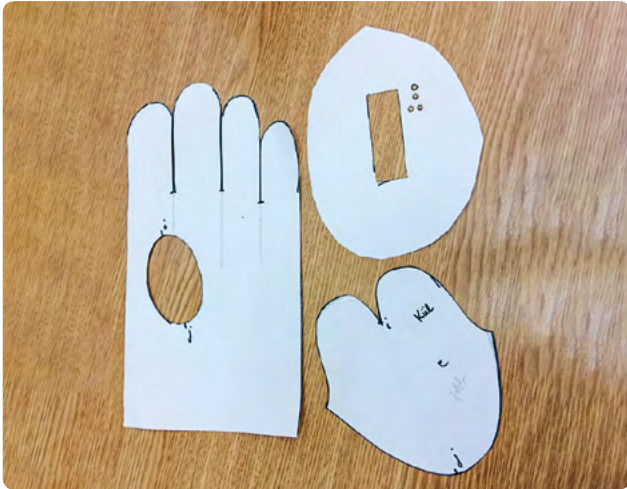
<Livello 2>

Gli studenti di una classe possono essere divisi in quattro gruppi. Un gruppo taglia e cuce il guanto, il secondo gruppo fa il circuito, il terzo scrive il programma e l'ultimo calibra i sensori.

<Costruzione del guanto>

Gli studenti hanno realizzato un modello (@2) dopo aver cercato idee su diversi siti web^[5]. Hanno piegato il materiale (in pelle nel nostro caso, ma altri materiali possono essere utilizzati) in tre e lo hanno tagliato utilizzando una sagoma. Per

completare il guanto, gli studenti hanno cucito due facce insieme. Hanno terminato la cucitura del materiale con l'ultimo pezzo dopo aver montato l'Arduino con LCD e sensori sul dorso del palmo. Gli studenti hanno ritagliato l'apertura per il display LCD e i pulsanti su questo terzo pezzo.



© 2: Modelli per il guanto

<Costruzione del circuito>

Gli studenti hanno realizzato il circuito, partendo da uno schema che era stato discusso e analizzato in precedenza con l'insegnante. Il circuito può essere fisso (saldato) o no. Si deve tenere conto del corretto collegamento dei sensori alla scheda Arduino, ovvero il GND dal sensore al GND sulla scheda Arduino, il VCC dal sensore a 5V sull'Arduino, e l'OUT sul sensore ad uno degli ANALOG IN (A0, A1, A2, A2, A3, A4 o A5) sulla scheda. Se un sensore deve essere collegato al DIGITAL IN, fare attenzione a non utilizzare uno degli ingressi utilizzati per il display LCD perché si verificherebbero errori di funzionamento. Nel nostro esempio, abbiamo collegato i seguenti sensori: temperatura [A1], luce [A2], umidità [A3], magnetismo [A4] e prossimità [A5] (vedi ©3a-3e).^[6]

<Scrittura del codice>

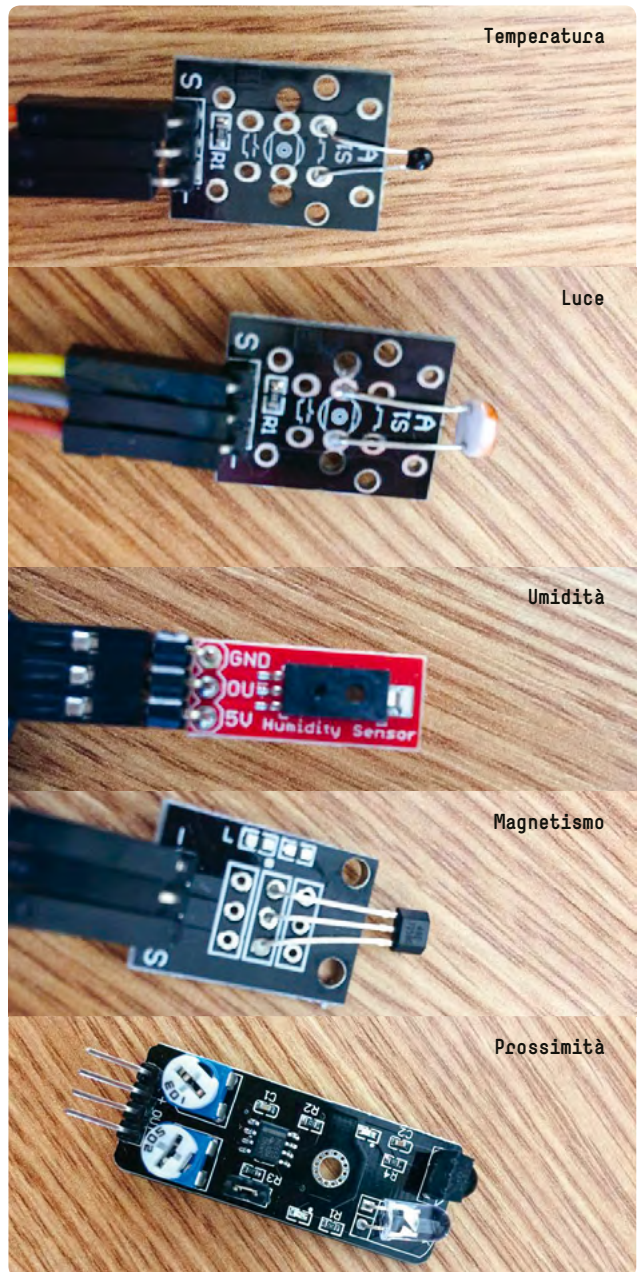
Gli studenti delle scuole superiori che stanno studiando il linguaggio di programmazione C^[2] possono facilmente programmare Arduino^[4]. Ci sono molti tutorial disponibili online in varie lingue. I nostri studenti hanno utilizzato un sito web in rumeno^[7]. Tutorial in inglese sono offerti sul sito web di Arduino o sui siti web dei distributori.^[8] Gli studenti possono trovare tutorials in molti altri siti.

L'insegnante può guidarli nella scrittura del programma per Arduino, ma è anche possibile trovare online il codice già completo che abbiamo usato.^[9]

<Calibrazione dei sensori>

Ci sono sensori calibrati disponibili, ma ci sono anche sensori non calibrati; è comunque molto istruttivo per gli studenti trovare un modo per calibrarli. I nostri studenti hanno trovato formule e modalità di calibrazione per alcuni dei sensori su Internet. Ad esempio, esiste una formula particolare per il sensore di umidità^[6], perché la funzione con cui variano i valori visualizzati non è lineare.

Per quanto riguarda la calibrazione del sensore di temperatura, gli studenti hanno messo in grafico i valori visualizzati dal sensore. Hanno usato un termometro calibrato in laboratorio e hanno associato il valore visualizzato con il valore del termometro. Hanno scoperto che i valori letti da questo sensore variano linearmente e sono riusciti a scrivere la formula di cali-



© 3a-e: Sensori

brazione. Sono comunque disponibili vari esempi di formule di calibrazione per i sensori di umidità e temperatura nel materiale aggiuntivo fornito online.^[9]

Una volta che gli studenti hanno calibrato i sensori, completato il programma e controllato il display per assicurarsi che i sensori sulle dita del guanto siano correttamente correlati con i dati che appaiono sullo schermo, il guanto viene terminato e chiuso. L'ultimo passo è quello di cucire/fissare lo strato esterno del materiale. Gli studenti hanno usato alcuni anelli di fissaggio per ogni sensore su ogni dito [14] per fissare meglio i sensori.



© 4: Fissaggio dei sensori

<Algoritmo da usare con altri linguaggi di programmazione>

Se si desidera utilizzare un altro ambiente di programmazione, è disponibile online un diagramma con tutti gli elementi necessari per la struttura logica del programma.^[9]

<Conclusione>

Agli studenti piace scoprire cose nuove e possono essere molto creativi. Amano fare misure sperimentali, per di più il guanto sembra provenire da un film di fantascienza. Gli studenti hanno la possibilità di comprendere meglio gli elementi di programmazione e di realizzare immediatamente un autentico strumento di misura.

Questa esperienza è stata per noi senza precedenti e gli insegnanti e gli studenti hanno potuto apprendere molte cose insieme.

La maggior difficoltà incontrata: non è facile trovare i sensori giusti^[6] e calibrarli, ma comunque si riesce sempre a trovare una soluzione. Se non è possibile scrivere ed implementare una formula di calibrazione, una soluzione sta nell'acquisto di sensori calibrati, anche se sono più costosi.

Questo guanto potrebbe anche essere costruito utilizzando un Calliope mini, che lo renderebbe più leggero e più piccolo. I nostri studenti si sono mostrati interessati a realizzare un guanto di questo tipo pur essendo diverso il linguaggio di programmazione.

<Attività cooperativa>

Studenti provenienti da scuole e paesi diversi potrebbero realizzare guanti utilizzando diversi microcontrollori e sensori, per poi discutere e confrontare i risultati. Si potrebbe chiedere a un insegnante di arte di contribuire alla progettazione del guanto. Inoltre, potrebbe essere organizzato un concorso tra le scuole, in cui gli studenti stessi propongono diversi design del guanto.

Il guanto è facile da spedire per posta, in modo che gli studenti di diverse scuole possano sperimentare con i guanti fatti dai loro coetanei in altre scuole e in altri paesi.

<Riferimenti>

- [1] www.arduino.cc
- [2] [https://en.wikipedia.org/wiki/C_\(programming_language\)](https://en.wikipedia.org/wiki/C_(programming_language))
- [3] www.arduino.cc/en/Main/Software
- [4] <https://snap.berkeley.edu/>
- [5] Diversi siti web con documentazione e tutorial su come cucire i guanti:
<http://ofdreamsandseams.blogspot.ro/2012/04/1950s-hand-sewn-leather-gloves.html>,
<https://so-sew-easy.com/easy-gloves-pattern-winter-comfort/>,
<http://sew-ing.com/make/gloves.html>,
www.glove.org/Modern/myfirstgloves.php,
www.instructables.com/id/How-to-Make-Gloves/ [tutti dicembre 2018]
- [6] Abbiamo usato i sensori nel 'kit sensori 37 in 1' per Arduino. Il sensore di umidità è il sensore di umidità HIH-4030, marca: Sparkfun, codice: SEN-VRM-09.
- [7] www.robofun.ro (Tutorial in rumeno. Ogni prodotto ha istruzioni su come collegarlo al circuito e sul linguaggio di programmazione da utilizzare. Il sito web contiene schemi e disegni, sensori o altri componenti riconoscibili dalle foto e il programma è molto facile da seguire. Questo significa che agli studenti non è necessaria alcuna conoscenza della lingua rumena.)
- [8] <http://mthackathon.info/resources/37-SENSOR-KIT-TUTORIAL-FOR-UNO-AND-MEGA.pdf> (i tutorial sono in inglese; contengono istruzioni su come collegare i sensori del kit ad Arduino) [dicembre 2018].
- [9] Tutti i materiali aggiuntivi sono disponibili all'indirizzo www.science-on-stage.de/coding-materials.

Science Friction

<Autore> Ilia Mestvirishvili

<Autore> David Shapakidze



<Info>

<Parole chiave> forza d'attrito, distanza d'arresto, sistema frenante antibloccaggio (ABS), programmazione app, acquisizione dati

<Discipline> fisica, informatica, matematica

<Età degli studenti> 14+

<Hardware> Arduino^[1], servo, motore, modulo Bluetooth, scudo motore, fotogate

<Linguaggi> Ambiente di programmazione Arduino^[2], AppInventor^[3], Snap4Arduino^[4], Blockly^[5]

<Livello di difficoltà di programmazione> facile, medio

<Sommario>

Un'indagine sui fattori che determinano la forza di attrito può essere trasformata in un interessante e divertente esperimento costruendo un'auto a basso costo controllata da Bluetooth, dotata di un semplice sistema frenante. Questo permetterà agli studenti di osservare dati reali come la velocità dell'auto (valore assoluto della velocità) prima di azionare i freni, la sua distanza di frenata, e valutare come la massa di un'auto e il tipo di superficie su cui viaggia influenzano la forza di attrito. Gli studenti effettueranno esperimenti per indagare le relazioni esistenti tra i fattori che influenzano la distanza di arresto, con la precisione necessaria per verificare le proprie ipotesi o quelle suggerite dall'insegnante.

<Introduzione concettuale>

L'attrito è una forza molto importante nella vita di tutti i giorni e viene insegnata in fisica sia a livello di scuola media sia a quello di scuola superiore. Tuttavia, gli esperimenti tradizionali legati al tema dell'attrito sono limitati e poco divertenti. Questo progetto trasformerà l'esplorazione dell'attrito in un emozionante progetto di gruppo che comporta:

1. costruire e mettere a punto un modellino di auto
2. programmazione di un microcontrollore Arduino^[1] per misurare la velocità istantanea e la distanza di arresto
3. programmazione di uno smartphone mediante AppInventor^[3] per inviare, ricevere e visualizzare i dati reali sullo schermo del cellulare

<Cosa fanno gli studenti/insegnanti>

Poiché queste tre attività possono essere svolte inizialmente contemporaneamente, si consiglia all'insegnante di dividere la classe in gruppi di due o tre studenti, che vi lavorano separatamente, e poi si riuniscono per discutere e rivedere il loro lavoro.

Se la forza d'attrito è contenuta nel vostro curriculum, l'insegnante può abbinare questo progetto alla parte teorica, per aumentare la motivazione degli studenti e migliorare la loro comprensione dei concetti teorici. Il modo migliore per iniziare l'attività è quello di porre le seguenti domande agli studenti e dare loro il tempo di fare attività di brainstorming e formulare le proprie idee, previsioni e ipotesi:

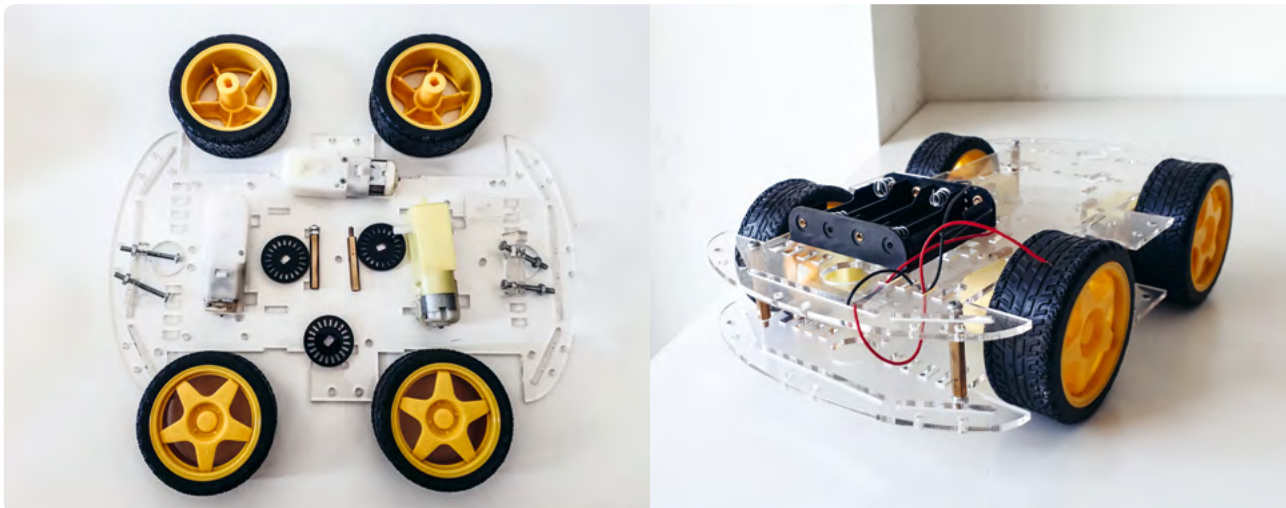
- ↳ Qual è la relazione tra la velocità di un'auto e la distanza di arresto? (Le risposte possono variare naturalmente, cioè "lo spazio di arresto è proporzionale alla velocità prima di azionare i freni", oppure alcuni potrebbero "ricordare" dalla fisica che lo spazio di arresto è effettivamente proporzionale al quadrato della velocità.)
- ↳ In che modo l'aumento della massa di un'auto influisce sulla distanza di arresto, a condizione che le altre grandezze siano immutate? (Una risposta comune è che aumentando la massa dovrebbe aumentare la distanza di arresto.)
- ↳ Come dobbiamo frenare per fermare un'auto il più rapidamente possibile? (Possibili risposte: il modo migliore è fermare completamente le ruote; se le facciamo ruotare in direzione opposta al movimento, questo fermerà l'auto più velocemente; ecc.)
- ↳ Se i freni anteriori e posteriori sono identici, fermeranno l'auto allo stesso tempo? (Gli studenti possono riflettere sulla propria esperienza con la bicicletta.)
- ↳ Qualsiasi altra domanda che potrebbe venire dall'insegnante o dagli studenti.

Dopo che gli studenti hanno scritto le loro idee iniziali, il passo successivo sarà quello di pensare a come costruire una semplice auto e ai dati necessari per potere verificare e sviluppare ulteriormente le loro idee iniziali. L'insegnante può facilitare questo processo e suggerire agli studenti di costruire un'auto in grado di raccogliere e inviare i dati rilevati ad un telefono, che a sua volta può controllare l'auto e ricevere e visualizzare i dati.

In base ai loro interessi, competenze e preferenze, in questa fase del progetto gli studenti possono, se lo desiderano, suddividersi in gruppi. Tuttavia, anche un solo gruppo di studenti può svolgere tutti questi compiti. Le prossime fasi dello sviluppo del progetto saranno le stesse per entrambi gli scenari.

<Costruire un telaio e montare i componenti elettrici>

Questo approccio prevede la costruzione di un'auto a partire dal kit di telai Arduino^[1] a basso costo e visibili in [@1](#). Insegnanti e studenti sono incoraggiati a provare diversi approcci alla progettazione e all'implementazione: ad esempio si possono provare diversi modi per raccogliere e inviare dati, per



© 1: Un kit telaio completo

controllare l'auto a distanza, così come diversi software e linguaggi per scrivere il codice necessario.

Dopo aver montato l'auto e aver deciso dove montare il controller Arduino e il motore, gli studenti dovranno pensare ai possibili modi per misurare la velocità dell'auto. L'approccio consigliato è quello di fare brainstorming e dare agli studenti l'opportunità di proporre alcune loro idee. Gli studenti, opportunamente aiutati dall'insegnante, potrebbero utilizzare un sensore photogate di moto rotatorio, per misurare in maniera precisa ed efficace, la velocità di rotazione di una ruota posteriore libera. In questo modo, gli studenti saranno in grado di misurare sia la velocità istantanea che la distanza percorsa. Questo metodo può essere adottato semplicemente utilizzando i materiali del kit telaio 4 ruote Arduino, oppure se l'insegnante preferisce, anche sviluppando il tutto in modo autonomo.

Per trasformare il numero di eventi conteggiati dal photogate in velocità o distanza percorsa sarà necessaria, ovviamente, un po' di matematica. Il kit che abbiamo usato comprende un disco con 22 fori e una ruota del diametro di $5,1 \pm 0,1$ cm. Non è difficile calcolare che 1 impulso dal photogate, cioè una rotazione di $1/22$ di un giro completo, corrisponde ad una distanza $d = 0,72$ cm. Allo stesso tempo, il photogate misura e invia un intervallo di tempo t in millisecondi tra due impulsi consecutivi. La velocità istantanea può essere calcolata dividendo $0,72$ cm per questo intervallo di tempo.

I seguenti passi possono essere utilizzati, indipendentemente dal fatto che gli studenti lavorino in gruppi diversi o in uno unico. Un unico gruppo esegue tutte le fasi una dopo l'altra, mentre gruppi diversi si divideranno le tre attività.

<Programmazione di Arduino>

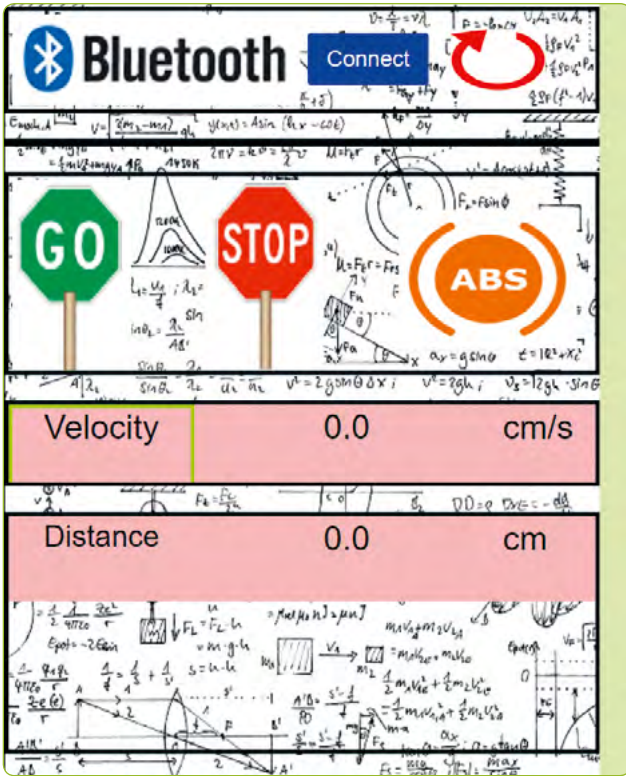
Il gruppo di programmazione Arduino può lavorare sulla codifica seguendo questo approccio:

1. Definire le azioni e di conseguenza i metodi o le funzioni per raccogliere e inviare i dati richiesti via Bluetooth.
2. Scrivere e testare ogni funzione separatamente.
3. Assemblare il tutto.

I principianti possono iniziare il design dei circuiti utilizzando TinkerCad^[6], un software di CAD design, che permette la progettazione e la simulazione di circuiti Arduino online, evitando così di bruciare componenti e di avere cortocircuiti nella fase di prototipazione.

Le sezioni seguenti descrivono ogni parte di attività in modo più dettagliato:

1. Le azioni richieste sono: avviare e fermare un motore, azionare e rilasciare i freni, misurare la distanza, misurare la velocità, inviare e ricevere i dati via Bluetooth.
2. La parte cruciale è quella di scrivere un codice per misurare la velocità e la distanza percorsa durante lo stesso esperimento. Entrambe le misure utilizzano impulsi dal photogate e si attivano quando viene ricevuta la cifra '2' dall'applicazione telefonica via Bluetooth:
 - ↳ Per misurare la distanza, c'è un contatore che inizia a contare gli impulsi inviati all'Arduino da una ruota posteriore e rotante liberamente, dopo che i freni sono stati applicati alle ruote anteriori.
 - ↳ Gli intervalli di tempo tra gli impulsi vengono utilizzati per misurare la velocità istantanea. Una ruota posteriore gira di $0,72$ cm durante un impulso, quindi questo deve essere diviso per l'intervallo di tempo tra gli impulsi.
 - ↳ Si può anche decidere di implementare la funzionalità di un sistema antibloccaggio (ABS) facendo intervenire i freni e rilasciandoli ad intervalli di tempo compresi tra



© 2: Interfaccia utente dell'applicazione

50 e 200 ms (valore ottimizzato sperimentalmente): nella maggior parte dei casi si avrà uno spazio di frenata più breve.

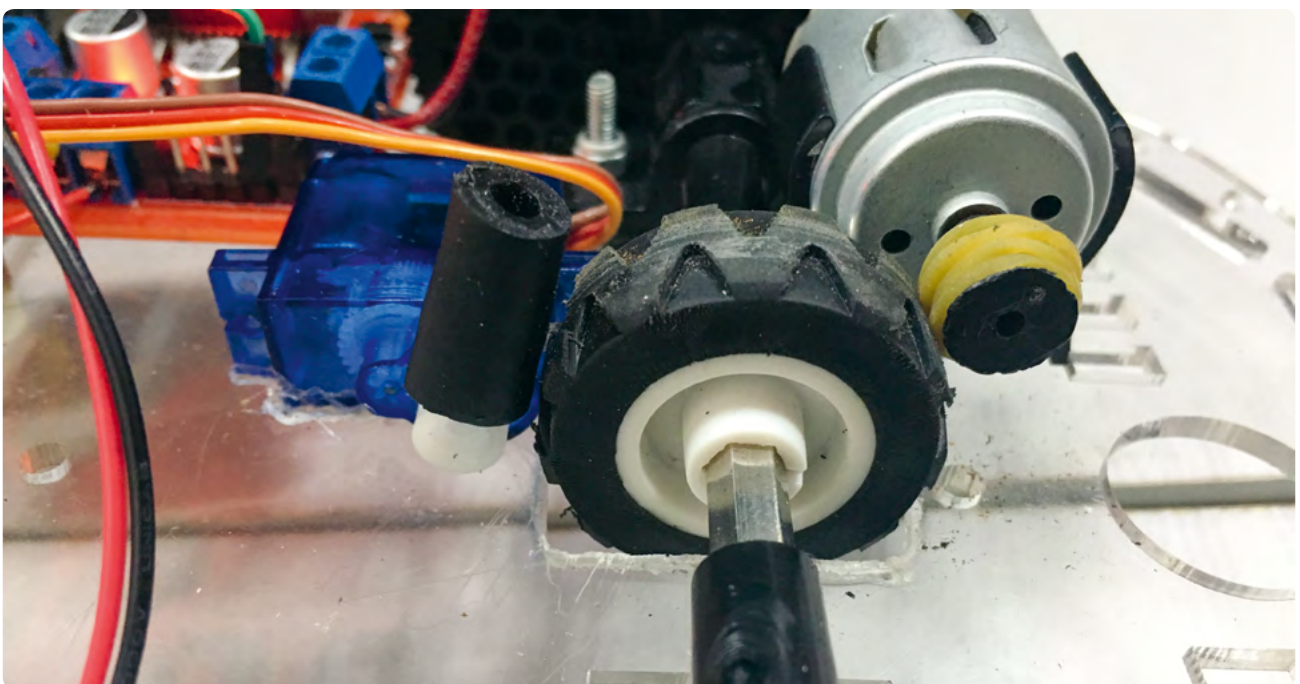
3. Quando si assembla un programma per Arduino, è necessario assicurarsi che tutto avvenga nel ciclo principale. Se il programma viene interrotto in un particolare punto, influirà su tutti i passaggi successivi.

Il codice di esempio e i riferimenti ad altre fonti per ciascuna delle funzionalità descritte sono disponibili online^[7] ma, con un piccolo aiuto da parte degli insegnanti, gli studenti dovrebbero essere incoraggiati a provare a scrivere il proprio codice in autonomia.

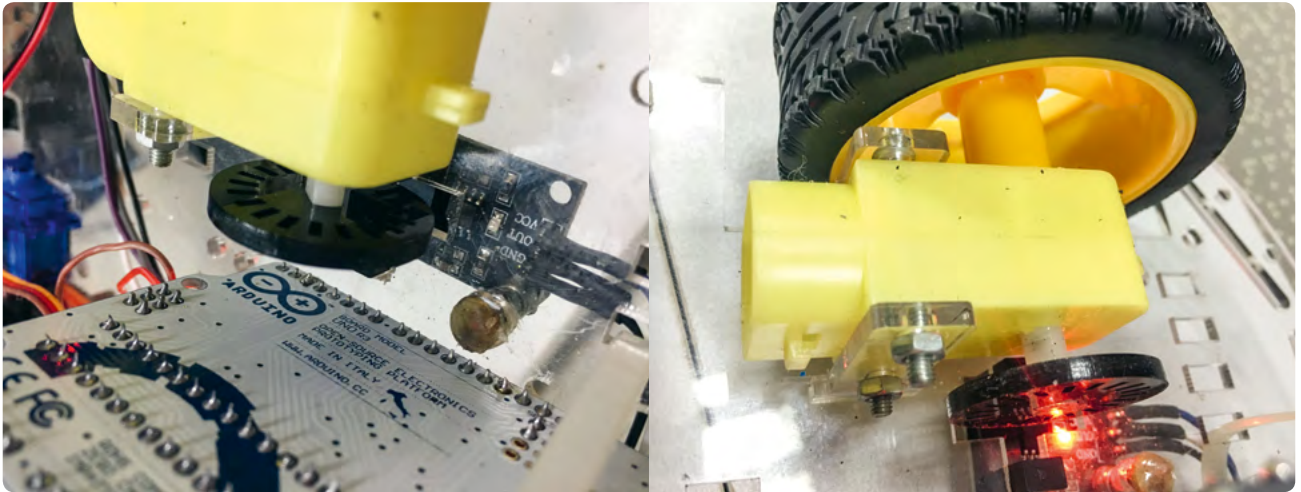
<Programmazione Android>

Il gruppo di programmazione Android può provare l'utilizzo di AppInventor^[3] e pensare a come visualizzare i dati sullo schermo (interfaccia utente). Gli studenti decideranno dove e come disporre i pulsanti per il controllo dell'auto, nonché i pannelli e le etichette per visualizzare i dati su uno schermo [2]. Il codice del programma AppInventor è fornito online^[7] e ha le seguenti funzionalità:

1. Premendo il pulsante **START** invia '1' all'Arduino^[1] via Bluetooth, che avvia il motore in auto.
2. Premendo il tasto **STOP** si invia '2' all'Arduino via Bluetooth, che ferma il motore e poi aziona i freni.
3. Premendo il pulsante **ABS** invia '3' all'Arduino via Bluetooth, che ferma il motore e poi frena ad intervalli regolari (simulando una funzione **ABS**).
4. Dopo aver premuto il pulsante **STOP** o il pulsante **ABS**, i dati ricevuti relativi alla velocità istantanea prima dell'arresto e sulla distanza percorsa dall'auto dopo aver azionato i freni, cioè lo spazio di arresto, saranno visualizzati in due spazi corrispondenti con le etichette "velocità" e "distanza" rispettivamente.
5. Il pulsante **RESET** invia '0' all'Arduino via Bluetooth, cancella i dati di velocità e distanza, quindi resetta l'Arduino.



© 3: Un primo piano dell'impianto frenante



© 4: Il photogate

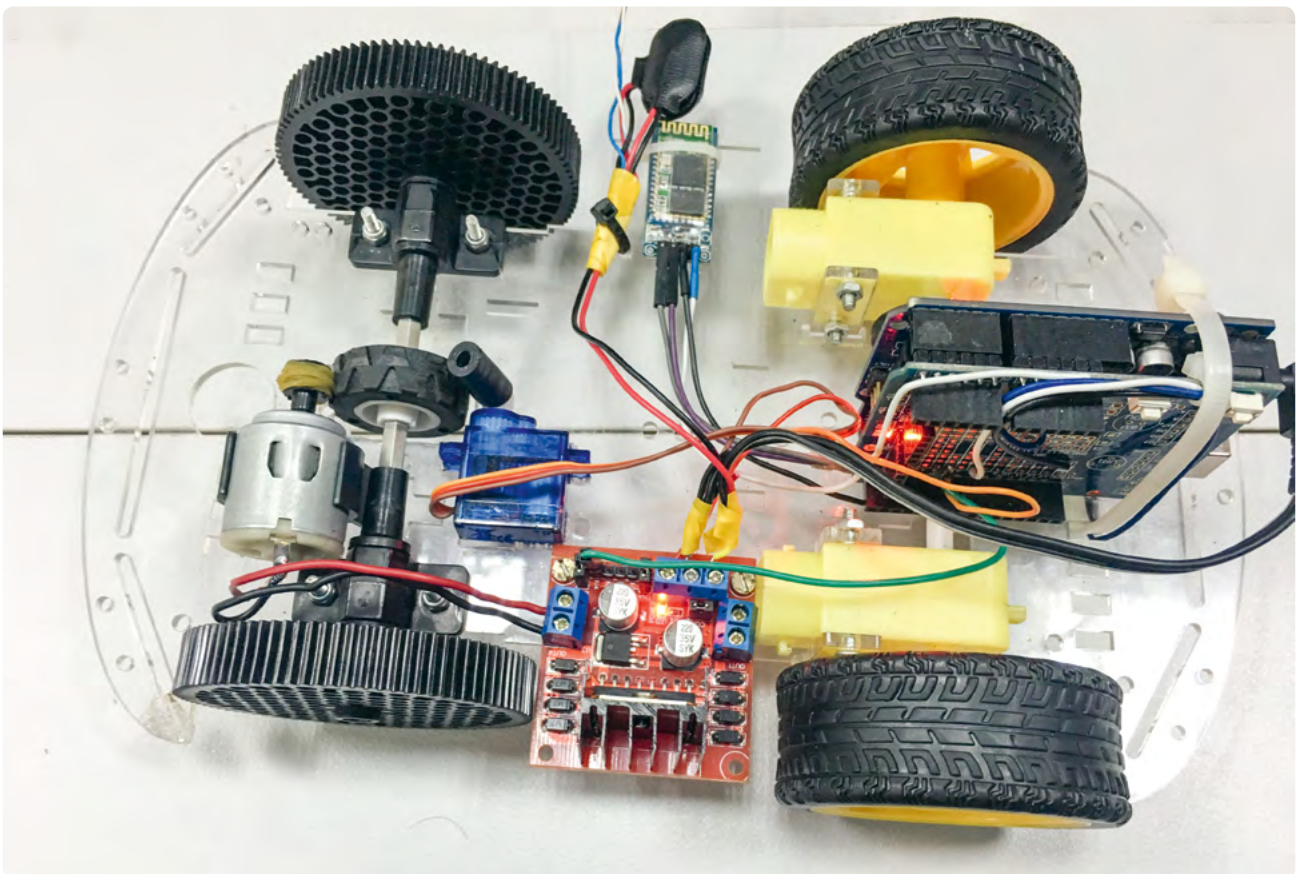
Consigliamo agli insegnanti di utilizzare il codice fornito come riferimento, ma di dare agli studenti l'opportunità di esplorare AppInventor^[3] e di scrivere il proprio codice in base alle funzionalità sopra descritte.

<Costruzione auto>

Al gruppo di costruttori di auto viene chiesto di trovare i modi migliori per collegare e collocare i componenti come il motore, il portellone, il servo, le batterie, il modulo Bluetooth, lo schermo motore e, infine, la scheda Arduino stessa. Quando il servo

gira, è importante che eserciti la pressione giusta su una levetta che si trova sul disco rotante, con un tubetto di gomma, che è montato sull'asse della ruota anteriore [©3].

Questo può fornire una forza sufficiente per fermare le ruote anche immediatamente. Altrettanto importante è la posizione del sensore photogate. Assicuratevi che conti correttamente tutti gli impulsi: il sensore photogate Arduino consigliato è dotato di un LED incorporato, che lampeggia alla rilevazione di un transito attraverso di esso. Per questo, assicuratevi che il



© 5: L'auto assemblata

photogate conti correttamente le rotazioni quando le ruote posteriori ruotano [4].

Inoltre, controllate bene che le ruote posteriori ruotino sempre il più liberamente possibile. Velocità e distanza, infatti, sono calcolate utilizzando la libera rotazione delle ruote posteriori. L'auto una volta costruita dovrebbe assomigliare all'esempio rappresentato in [5].

Si consiglia di dare come riferimento agli studenti linee guida dettagliate, ma di dare loro l'opportunità di fare brainstorming e di implementare soluzioni originali.

<Conclusione>

Questo progetto è un modo divertente per gli studenti di imparare concetti fisici fondamentali come la forza di attrito cinetico e statico, e, allo stesso tempo, tecnologie relativamente moderne come l'ABS, dove fisica, elettronica, programmazione e design si uniscono per esplorare i fattori che influenzano la distanza di arresto di un'auto. Interpretare e analizzare dati sperimentali reali ha sempre il sapore di una sfida. Sono sempre coinvolte nozioni teoriche molto importanti come l'errore, la validità, la riproducibilità e la visualizzazione dei dati con un numero di cifre limitato. Il progetto permette agli studenti di sperimentare e comprendere la forza d'attrito mediante una lezione pratica e stimolante.

<Attività cooperativa>

Questo progetto offre un grande potenziale di collaborazione, poiché i suoi tre elementi indipendenti – il design della vettura, la codifica di Arduino^[1] e la codifica con AppInventor^[3] – potrebbero essere ulteriormente sviluppati e migliorati. Tutti i partner coinvolti trarrebbero vantaggio dal contributo di ciascuno di essi a una qualsiasi di queste componenti.

Un'altra opzione di cooperazione potrebbe essere quella di svolgere una competizione tra squadre scolastiche basata su chi realizza più rapidamente un modellino di auto fornendo a tutti gli stessi materiali come ruote e massa, ma anche le stesse condizioni come superficie "stradale" e velocità prima dell'arresto.

I nostri ringraziamenti vanno ai nostri colleghi greci: Astrinos Tsoutsoudakis per aver fornito importanti suggerimenti sui contenuti di fisica di questo progetto e Georgios Georgoulakis per i suoi utilissimi consigli sulla programmazione. Vorremmo anche ringraziare per l'ampio feedback e il supporto ricevuto da Jörg Gutschank, che ha reso questo progetto più interessante e trasferibile.

<Riferimenti>

- [1] www.arduino.cc
- [2] www.arduino.cc/en/Guide/Environment
- [3] <http://appinventor.mit.edu>
- [4] <http://snap4arduino.rocks/>
- [5] <https://developers.google.com/blockly/>
- [6] www.tinkercad.com/circuits
- [7] Tutti i materiali aggiuntivi sono disponibili all'indirizzo www.science-on-stage.de/coding-materials.

Rolling Sounds

<Autore> Georgios Georgoulakis

<Autore> Astrinos Tsoutsoudakis



<Info>

<Parole chiave> scienza fondamentale, acquisizione dati, moto circolare, onde sonore, geometria, trigonometria

<Discipline> fisica, matematica, informatica

<Età degli studenti> 14–17

<Hardware> Microcontrollore Arduino^[1] o simile (con i driver appropriati installati), microfono, high output buzzer (cicalino, generatore di suono), trapano elettrico con la sua base, materiali per la lavorazione del disco in legno

<Linguaggi> Snap4Arduino^[2]

<Livello di difficoltà di programmazione> medio

<Sommario>

Questa unità didattica interdisciplinare e combina la fisica con l'informatica. Può essere utilizzata in laboratorio di informatica o di fisica e comporta, insieme al calcolo di altre grandezze fisiche relative al moto circolare uniforme, il calcolo della velocità lineare di rotazione in due metodi diversi.

Il primo di questi metodi rileva la frequenza con cui il segnale di un sensore di prossimità a infrarossi, viene bloccato da una piccola striscia metallica collegata ad un particolare punto di un disco rotante, e misura un periodo. Il secondo metodo sfrutta l'effetto Doppler di una sorgente sonora in movimento, posta sopra il disco.

<Introduzione concettuale>

Lo studio sperimentale delle grandezze fisiche (periodo T , frequenza f , velocità lineare v e velocità angolare ω) del moto circolare uniforme si basa sulle conoscenze che si acquisiscono nelle scuole medie superiori greche (età degli studenti: 14–17 anni) e nei curricula di altre scuole secondarie europee, dove si apprende anche l'effetto Doppler. La frequenza e l'intensità della velocità angolare e lineare sono ottenute utilizzando formule ben note:


$$f = \frac{1}{T}, \omega = \frac{2\pi}{T} \text{ and } v = \frac{2\pi r}{T}$$

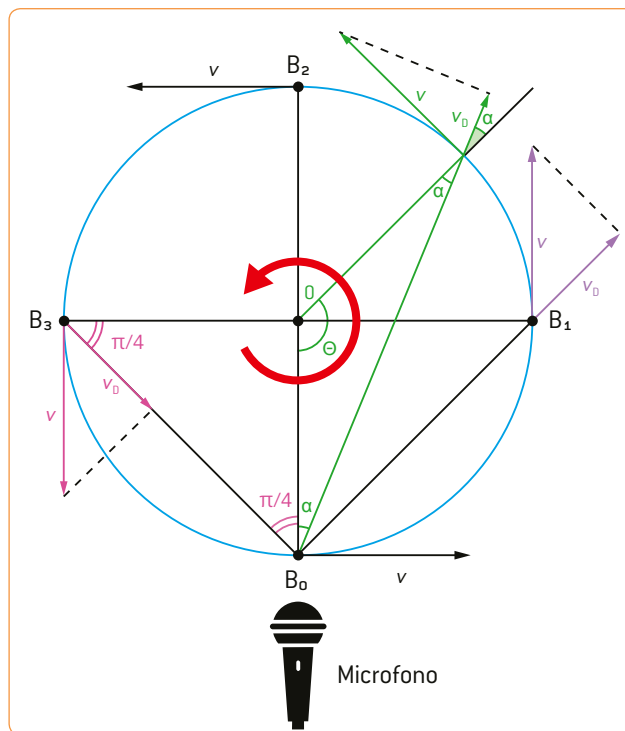
T è ricavato grazie all'oscillatore interno del microprocessore (il tempo tra due rilevazioni del segnale), e il raggio r rappresenta la distanza tra la striscia metallica, o il buzzer, e il centro del disco.


<Esperimento dell'effetto Doppler>


L'effetto Doppler è il cambiamento della frequenza o della lunghezza d'onda di un'onda sonora che avviene se la sorgente si muove relativamente ad un osservatore. Un esempio comune di questo fenomeno è il cambiamento di frequenza di una sirena su un'ambulanza in movimento. Quando l'ambulanza si avvicina, il suono percepito è più acuto di quello originale, mentre,

quando l'ambulanza si allontana, è più grave. Il suono udito ha la stessa frequenza di quello emesso solo nel momento in cui l'ambulanza passa davanti all'osservatore.

Nel nostro esperimento utilizziamo un buzzer (cicalino) come sorgente e collocato su disco rotante, e un microfono statico come osservatore (vedi  1).



 1: Lo schema dell'esperimento

Man mano che il disco ruota in senso antiorario ( 1), la componente della velocità nella direzione della linea della corda (in modulo), ottenuta unendo il punto B_0 con il punto in cui si trova ogni volta il buzzer B , aumenta da zero al massimo nel punto B_1 e successivamente scende a zero nel punto B_2 . Questa componente di velocità è la velocità effettiva di arretramento per l'effetto Doppler. Dal punto B_2 a B_3 , la componente di velocità, che ora rappresenta la velocità di avvicinamento, aumenta da zero al massimo nel punto B_3 e poi diminuisce nuovamente a zero nel punto B_0 .

Calcoliamo ora la velocità lineare applicando la formula dell'effetto Doppler per una sorgente in movimento nel punto B_3 e per un osservatore a riposo (il microfono). La velocità lineare rimane in ogni istante perpendicolare al raggio del cerchio e l'angolo di $\frac{\pi}{4}$ è determinato dalle proprietà geometriche del triangolo retto e isoscele formato B_3OB_0 .

$$f = f_0 \cdot \left(\frac{v_s}{v_s - v_D} \right) \Rightarrow f = \frac{f_0 \cdot v_s}{v_s - v_D} \Rightarrow f \cdot v_s - f \cdot v_D = f_0 \cdot v_s$$

$$\Rightarrow f \cdot v_D = (f - f_0) \cdot v_s \Rightarrow v \cdot \cos\left(\frac{\pi}{4}\right) = \left(\frac{f - f_0}{f} \right) v_s$$

$$\Rightarrow v = \left(\frac{f - f_0}{f} \right) \frac{v_s}{\cos\left(\frac{\pi}{4}\right)}$$

- f: frequenza misurata
- f₀: frequenza emessa
- v: velocità
- v_s: velocità del suono
- v_D: velocità di allontanamento/avvicinamento

Poiché l'implementazione di una Trasformata di Fourier (FFT) per estrarre il contenuto in frequenza del suono prodotto va ben oltre le capacità di codifica degli studenti, il software di editing audio gratuito Audacity^[3] può fornire un aiuto utile per ottenere un file di testo contenente tutti i dati necessari.

<Altri materiali a disposizione>

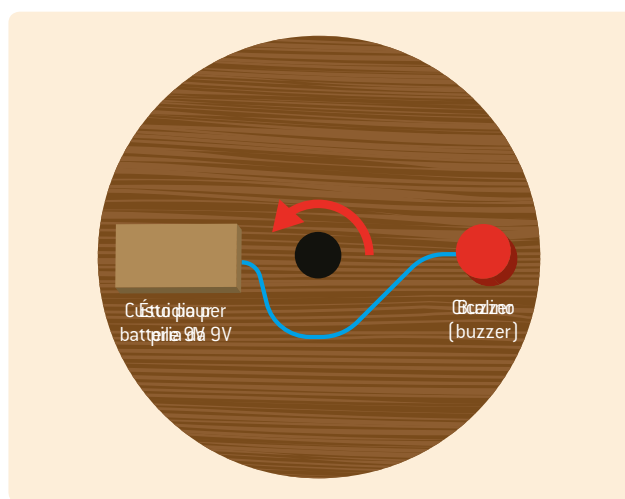
L'attività qui descritta è in una versione semplificata rispetto a quella effettivamente svolta e non include gli esperimenti fatti con un sistema a tubo statico di Pitot e un sensore di pressione online^[4], invece, sono liberamente disponibili tutte le informazioni necessarie. Il materiale online contiene una descrizione dettagliata dell'impianto sperimentale, idee costruttive alternative, insieme alla documentazione teorica e all'analisi passo-passo delle procedure utilizzate.

<Cosa fanno gli studenti/insegnanti>

Nella sezione inerente alla fisica dell'unità didattica, gli studenti misurano le grandezze fisiche del moto circolare per diversi raggi e fanno pratica sull'effetto Doppler. In ogni caso essi devono, prima di tutto, progettare ed assemblare un dispositivo sperimentale di base.

<La costruzione del disco in legno>

Gli studenti allestiscono lo strumento costituito da un disco di legno azionato da un trapano elettrico e dotato di un cicalino (buzzer) collegato ad una batteria da 9V. Un sensore di prossimità a infrarossi, indipendente ma collocato vicino, fornisce al microcontrollore una misurazione di segnale per ogni rotazione completa, mentre un microfono (se ne possono usare an-



© 3: Disco visto dall'alto



© 2: Configurazione sperimentale di base

che di molto economici) registra il suono prodotto. La variazione di frequenza Doppler dovrebbe essere idealmente udibile per la velocità di rotazione scelta, che viene mantenuta piuttosto bassa per motivi di sicurezza. (📷2 & 3)

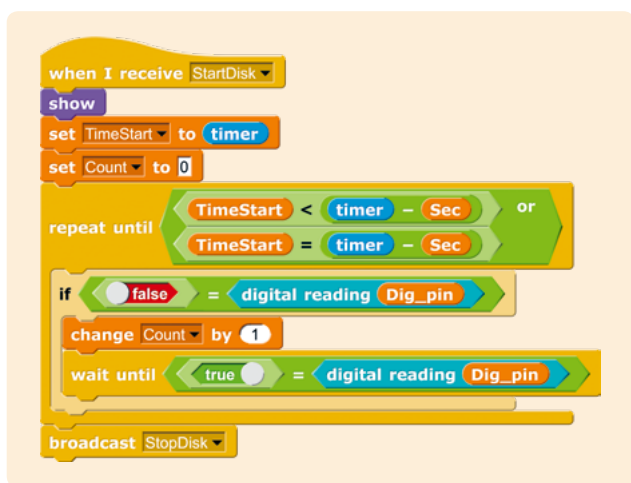
Come mostrato in 📷4 è stata sviluppata un'interfaccia attraente e di facile utilizzo per gli studenti dove poter inserire tutti i parametri richiesti ed estrarre i valori calcolati.



📷 4: L'interfaccia utente per i parametri

Gli studenti hanno bisogno di competenze di programmazione di base e di una certa esperienza nell'uso di linguaggi di programmazione a blocchi (come Scratch o Snap!). Qui di seguito diamo agli studenti un modello di base per lavorare con il loro codice in modo che si concentrino sugli obiettivi dell'unità didattica e non sull'interfaccia utente e sull'aspetto del programma.

Viene quindi fornito il modello di base in forma di progetto Snap!^[5] in un file .xml e un foglio di lavoro per dare agli studenti le istruzioni di base sul modello di esperimento e dare un'idea di ciò che ci aspettiamo da loro. Sia il modello che le richieste per gli studenti sono disponibili online.^[4]

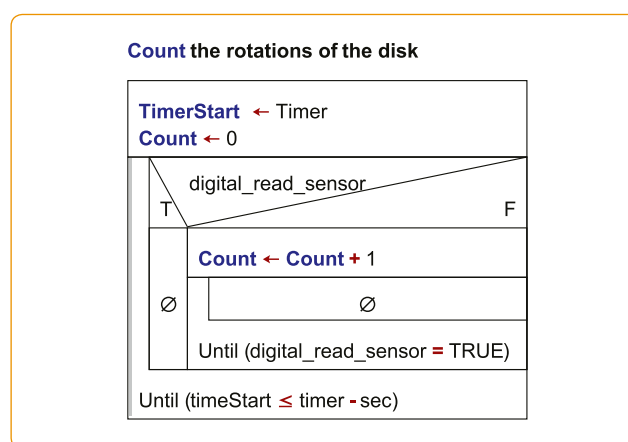


📷 5: Il calcolo del periodo di rotazione

A questo punto gli studenti controllano e convalidano i dati acquisiti, si collegano e comunicano con i dispositivi esterni, ricevono

ed elaborano i dati dai sensori e scrivono un semplice algoritmo di ricerca sequenziale.

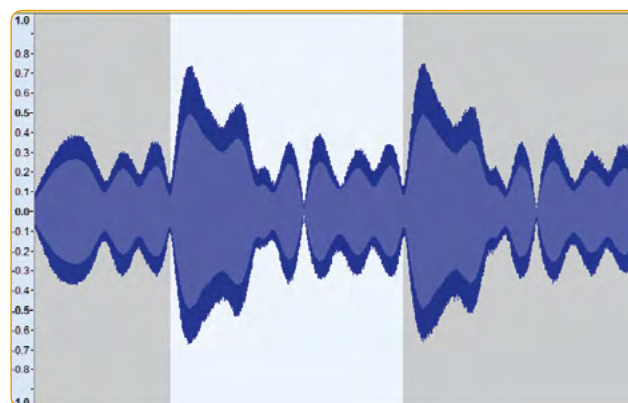
Il programma finito è disponibile anche per l'insegnante per il download nelle risorse di riferimento^[4]. 📷5 mostra una schermata di esempio dell'ambiente di programmazione Snap4Arduino^[2]. 📷6 mostra il corrispondente diagramma di Nassi-Shneiderman.



📷 6: Diagramma di Nassi-Shneiderman per il periodo

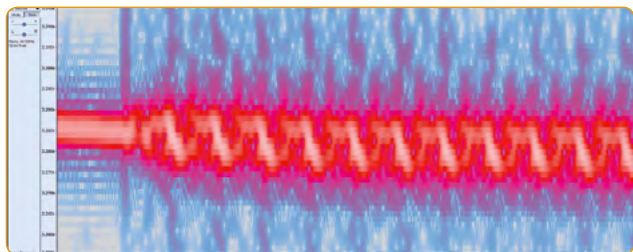
Come accennato in precedenza, si consiglia di utilizzare un software di editing audio gratuito come Audacity^[3] per estrarre il contenuto in frequenza del suono prodotto. Il software fornisce un adeguato file di testo con tutti i dati necessari agli studenti per imparare ad elaborare un segnale sonoro attraverso un software specializzato.

Il segnale importato e l'elaborazione di base sono illustrati nelle 📷7-9, mentre 📷10 mostra una parte dell'esportazione dei dati finali. Per evitare analisi approfondite e per consentire una migliore comprensione da parte degli studenti, è necessario fare un'ipotesi di approssimazione. La frequenza evidenziata in giallo, che presenta il massimo livello di segnale, è la frequenza del buzzer a riposo o quella misurata nei punti B₀ e B₂, mentre le linee blu e verde rappresentano i picchi più vicini delle nostre prime frequenze ottenute tra quelle variate dall'effetto Doppler (📷10).

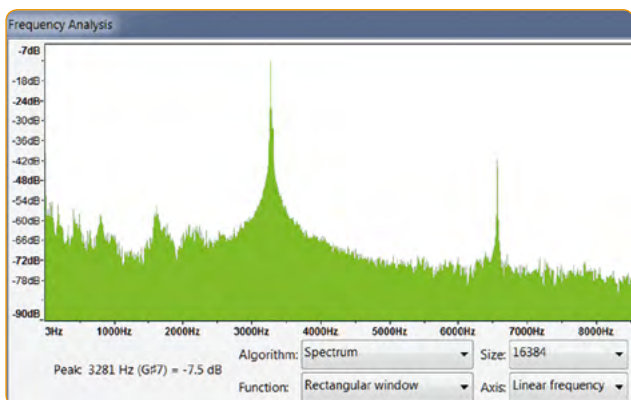


📷 7: Una forma d'onda sonora registrata

In ogni caso, il codice presentato trova solo il segnale della frequenza evidenziata in verde, ma per una migliore precisione può essere facilmente modificato per trovare entrambi. Per velocizzare le cose, la sezione dati potrebbe anche essere ridotta a soli 50–100 valori al di sopra e al di sotto della frequenza del livello massimo.

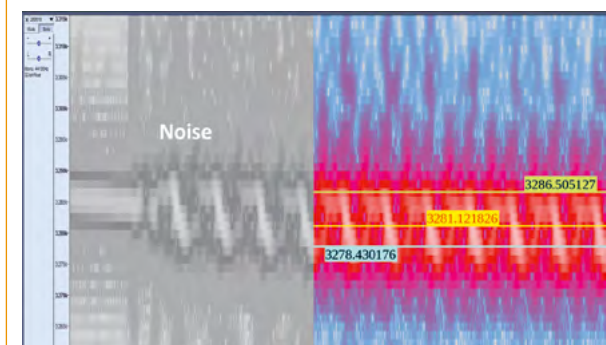


8: Uno spettrogramma che mostra lo spostamento Doppler



9: Analisi di frequenza con la trasformata veloce di Fourier

Frequenza [Hz]	Livello [dB]
3273.046875	-27.597595
3275.738525	-22.331339
3278.430176	-12.437067
3281.121826	-7.5547090
3283.813477	-10.041918
3286.505127	-9.7750780
3289.196777	-16.848948
3291.888428	-26.916197



10: Dati esportati

L'elaborazione dei dati dello spettro sonoro è mostrata nelle 11 & 12. Tutte le informazioni dettagliate, per esempio sulle variabili utilizzate, sono disponibili online.^[4]

<Algoritmo da usare con altri linguaggi di programmazione>

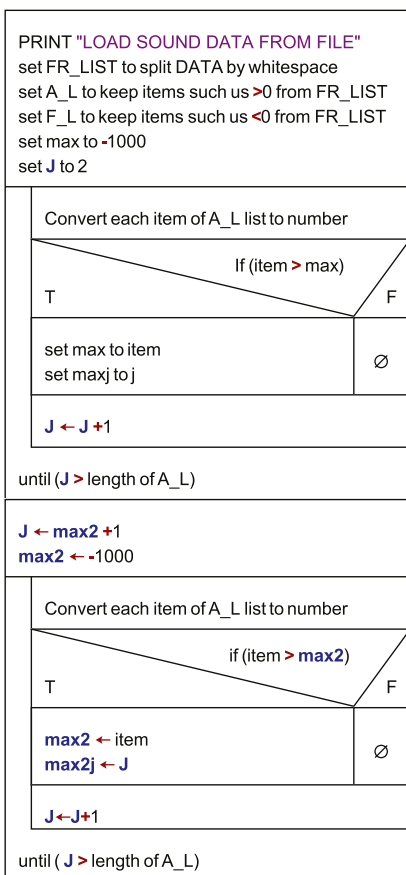
Il modello di metaprogramma di base permette un facile trasferimento a qualsiasi altro linguaggio di programmazione, purché dotato di una libreria di base per la comunicazione con il microcontrollore. In questo modo, la scelta del microcontrollore non avrà alcun effetto significativo sul progetto.

```

when I am clicked
  say Don't forget to Right-click string and load data for 2 secs
  set FR_LIST to split String by whitespace
  set F_L to keep items such that [ ] > 0 from FR_LIST
  set A_L to keep items such that [ ] < 0 from FR_LIST
  set max to 1000
  set j to 2
  repeat until j > length of A_L
    set Id_num to join
      item 1 of split item j of A_L by [ ]
      item 2 of split item j of A_L by [ ]
    if Id_num > max
      set max to Id_num
      set maxj to j
    change j by 1
  set j to maxj + 1
  set max2 to -1000
  repeat until j > length of A_L
    set Id_num to join
      item 1 of split item j of A_L by [ ]
      item 2 of split item j of A_L by [ ]
    if Id_num > max2
      set max2 to Id_num
      set max2j to j
    change j by 1
  
```

11: La parte di spostamento Doppler dell'esperimento^[4]

Sound Data Processing



© 12: Diagramma Nassi-Shneiderman per l'elaborazione dei dati sonori

<Conclusion>

Il progetto presentato è a basso costo, facile da assemblare ed utilizzare; la nostra speranza è che sia interessante e stimolante per altri colleghi e per gli studenti.

<Attività cooperativa>

La piattaforma Science on Stage si basa sullo scambio di idee didattiche e sull'implementazione di approcci educativi innovativi. La cooperazione didattica con Ilia Mestvirishvili e David Shapakidze, un superbo team di partner dalla Georgia, potrebbe rivelarsi impegnativo a causa della distanza e dei problemi di tempistiche lavorative, ma in ogni caso ci ha già permesso di sviluppare tecniche innovative. Nonostante la mancanza di un background comune nella didattica per alunni con bisogni educativi speciali, sarebbe una buona idea modificare il progetto per renderlo accessibile a tutti gli studenti.

<Riferimenti>

- [1] www.arduino.cc
 - [2] <http://snap4arduino.rocks>
 - [3] www.audacityteam.org
 - [4] Tutti i materiali aggiuntivi sono disponibili all'indirizzo www.science-on-stage.de/coding-materials.
 - [5] <https://snap.berkeley.edu>
- ↳ www.physicsclassroom.com/mmedia/circmot/ucm.cfm
 ↳ <https://education.pasco.com/epub/PhysicsNGSS/BookInd-904.html>
 ↳ <http://hyperphysics.phy-astr.gsu.edu/hbase/Sound/dopp.html>
 ↳ http://newton.phys.uaic.ro/data/pdf/Doppler_experiment.pdf
 ↳ <https://manual.audacityteam.org/man/tutorials.html> (tutti dicembre 2018)

La Fisica animata

<Autore> Mihaela Irina Giurgea

<Autore> Corina Lavinia Toma



<Info>

<Parole chiave> animazione, sprite, blocchi, loop, grafici, leggi gravitazionali, collisioni, caduta libera, forza di attrito, lancio obliquo, movimento, momento, operatori, variabili

<Discipline> informatica, fisica, matematica

<Età degli studenti> 14–16

<Hardware> Il computer

<Linguaggi> Scratch^[1]

<Livello di difficoltà di programmazione> facile, medio

<Sommarío>

Cosa penseresti se ti dicessimo che i tuoi studenti potrebbero imparare due materie apparentemente molto diverse, fisica e informatica, più facilmente e allo stesso tempo? In questa unità, il “motore”, cioè l’ambiente di programmazione Scratch^[1], è lo strumento magico che aiuterà gli studenti a creare interessanti applicazioni sui fenomeni naturali quotidiani per comprendere meglio le leggi della fisica e migliorare le loro capacità di programmazione.

<Introduzione concettuale>

Perché abbiamo usato Scratch^[1]? Scratch è un ambiente di programmazione visiva che anima gli “sprite” (oggetti della programmazione) utilizzando blocchi sullo schermo del computer e aiuta gli studenti a creare applicazioni più facilmente rispetto ai classici ambienti di programmazione (C++, Java, ecc.). Inoltre, il nostro “motore” ci aiuta ad insegnare due materie che gli studenti considerano difficili: la fisica e l’informatica. Eliminando i principali ostacoli, l’impossibilità di immaginare (vedere) come funziona un fenomeno e la complicata sintassi del codice, abbiamo creato un nuovo modo piacevole di insegnare.

Gli studenti che sono stati coinvolti nello sviluppo di questa unità didattica avevano già una certa esperienza di programmazione, così sono stati in grado di capire come utilizzare Scratch. Lo usano sia nelle loro normali lezioni di informatica, sia in quelle facoltative. Inoltre, le conoscenze di fisica richieste fanno parte del curriculum standard, ed è sempre utile rivedere e applicare quanto appreso.

<Cosa fanno gli studenti/insegnanti>

L’unità si basa sull’alternanza di sequenze di apprendimento che coinvolgono la programmazione e la fisica.

In primo luogo, l’insegnante di informatica ha presentato le basi per realizzare un progetto in Scratch^[1].

Gli studenti hanno familiarizzato con diverse parole chiave relative all’ambiente di Scratch: stage, sprite, costumes e movements. Puoi seguire le istruzioni e le spiegazioni (fornite nei riferimenti di questa unità) della prima applicazione insegnata in Scratch, un’applicazione senza formule fisiche.^[2]

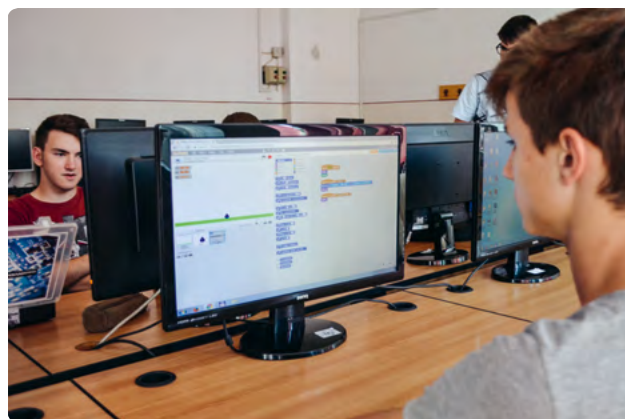
Gli studenti dovevano inizialmente capire le interazioni tra gli sprite e la loro sincronizzazione e come funziona un sistema di coordinate. È possibile trovare un tutorial completo per Scratch online.^[3]

Per comprendere meglio i principali algoritmi di Scratch, gli studenti hanno esaminato applicazioni interessanti ed entusiasmanti. Quando hanno visto il codice che sta dietro a queste applicazioni, a volte sono rimasti sorpresi scoprendo che possono crearle da soli.

Per la parte fisica di questa unità, gli studenti hanno applicato i principi teorici alla base dei fenomeni del mondo che li circonda. Per questo motivo, l’insegnante di fisica ha suggerito un’ampia varietà di argomenti^[4], che gli studenti hanno poi discusso: formule necessarie, possibili animazioni, il design, ecc.

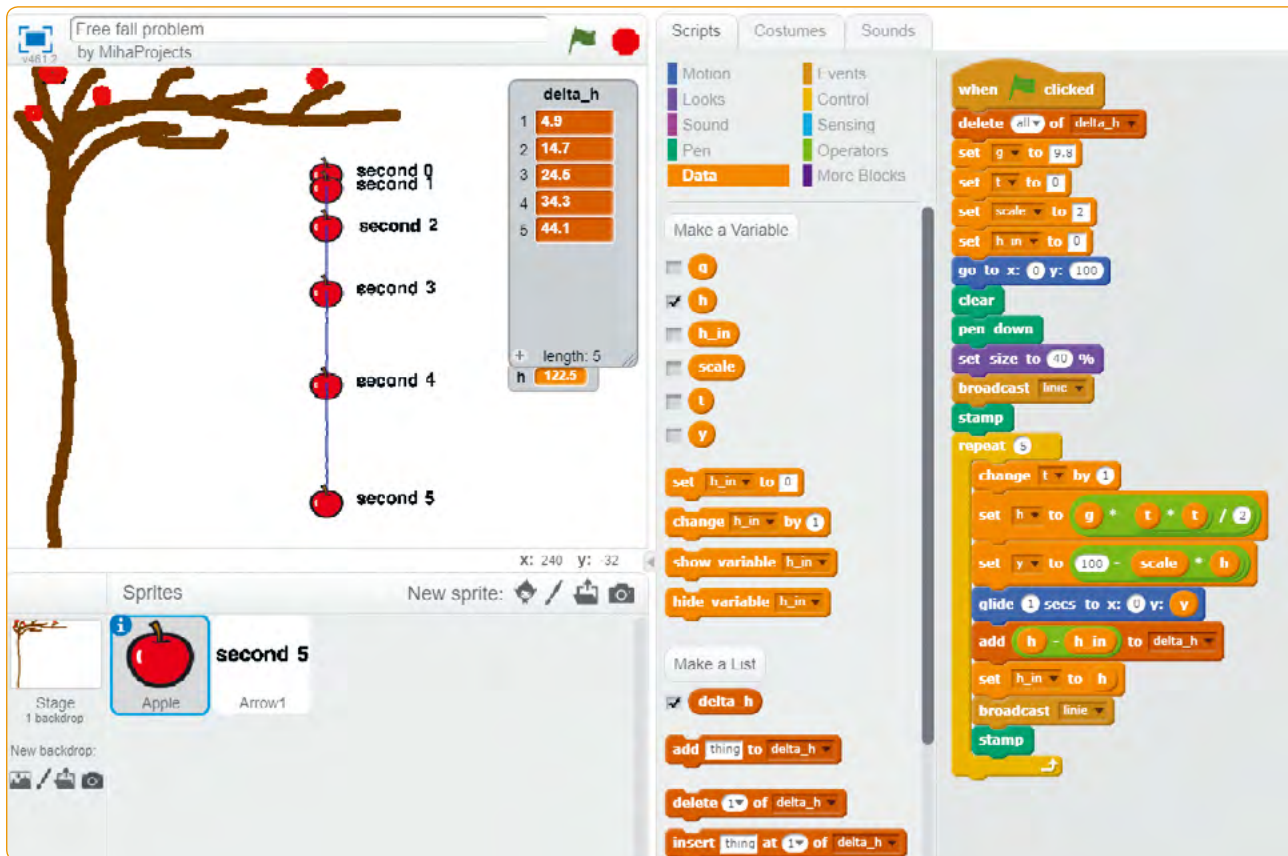
Gli studenti hanno scelto tra questi argomenti; una settimana dopo, abbiamo ricevuto applicazioni sul lancio obliquo di una palla; la caduta libera di una mela; la caduta, più complessa, di una goccia d’acqua oppure la collisione tra due palle, ma anche il movimento dei pianeti nel sistema solare o anche piccoli giochi.

All’inizio, gli studenti hanno lavorato individualmente con l’aiuto dei loro insegnanti. Gli studenti sono anche stati guidati dagli insegnanti e dai loro compagni di classe nelle fasi in cui i progetti richiedevano dei miglioramenti. [📷 1]



📷 1: Lavoro individuale

In un momento finale tutti gli studenti hanno presentato il loro progetto alla classe e hanno ricevuto feedback dai loro coetanei. In questo modo è stato più facile per gli studenti capire quali parti dei progetti dovevano essere migliorate sia della programmazione che della parte di fisica.



Ⓒ 2: Problema di caduta libera

Le sezioni seguenti contengono esempi di come ci siamo avvicinati alla fisica e alle parti di programmazione.

<Applicazione 1: Problema di caduta libera (mela di Newton)>

Ogni studente ha sentito parlare della caduta libera di questo oggetto storico: la mela di Newton.

L'applicazione in Ⓒ 2 si ispira ad un problema classico: qual è la distanza percorsa ogni secondo dalla mela di Newton in caduta libera?

Teoria fisica

Consideriamo un moto lineare con accelerazione gravitazionale costante $g = 9,8 \frac{m}{s^2}$.

Dopo un tempo t , la distanza $h(t)$ percorsa dalla mela è: $h(t) = \frac{gt^2}{2}$.

Il punto iniziale $h(0)$ è fissato sul ramo dell'albero da cui la mela si è staccata.

Poi calcoliamo la distanza percorsa per un periodo più lungo, $t + \Delta t$: $h(t + \Delta t) = \frac{g(t + \Delta t)^2}{2}$.

La formula generale per la distanza $\Delta h(t)$ percorsa dalla mela durante Δt , è:

$$\Delta h(t) = h(t + \Delta t) - h(t) = \frac{g(2t\Delta t + \Delta t^2)}{2}$$

Poi usiamo i dati del problema specifico per applicare la formula generale; in questo caso, $1s$ per Δt . Per il primo secondo, $t = t_{in} = 0$ si ottiene $\Delta h_1 = 4,9$ m, per il secondo successivo, $t = 1s$ risulta $\Delta h_2 = 3 \cdot 4,9$ m = 14,7 m e così via. Attraverso l'induzione matematica, possiamo calcolare la distanza percorsa nell'ennesimo secondo considerando $t = (n - 1)$ s:

$$\Delta h_n = \frac{9,8(2n - 1)}{2} \text{ m.}$$

Poi gli studenti possono calcolare, e anche vedere nell'animazione realizzata, che la distanza percorsa aumenta della stessa quantità pari a 9,8 m, ogni secondo.

Come lo programmiamo?

Variabili utilizzate:

g : accelerazione gravitazionale

t : contatore dei secondi (con valori: 0, 1, 2, 3, 4, 5)

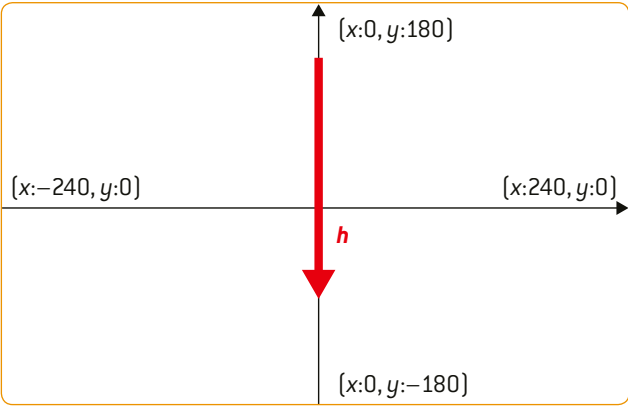
h : distanza percorsa dopo t secondi

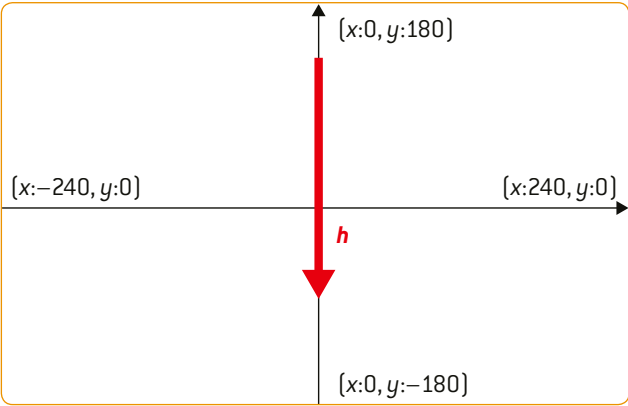
h_{in} : posizione iniziale della mela


$delta_h$: un vettore (elenco, array) con tutte le distanze percorse in ogni secondo

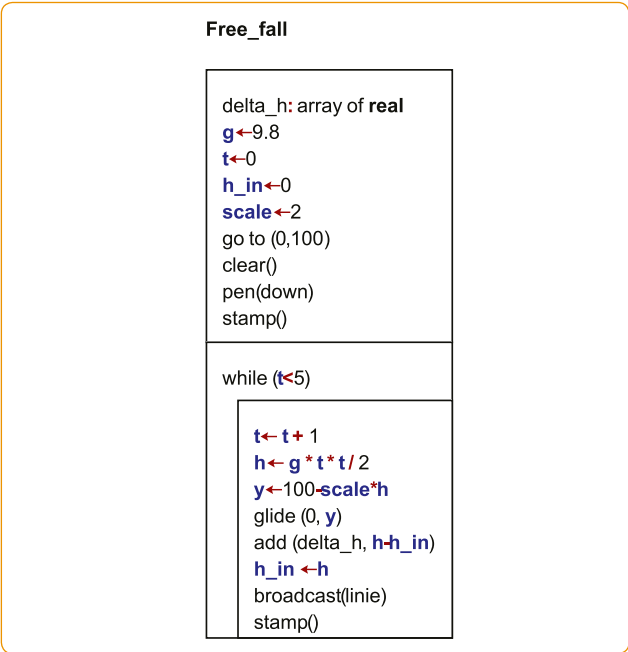
y : coordinata "y" della mela

Osservazione: in questa applicazione, la coordinata x rimane costante = 0, in modo da poter spostare più facilmente la traiettoria a sinistra o a destra sulla scena.

All'inizio, la mela si trova nel punto con le coordinate $[0,180]$. Il punto iniziale e la direzione della distanza percorsa $h(t)$ sono indicati nella .




 3: Orientazione del sistema di coordinate

Utilizzando un loop per 5 volte, abbiamo ricalcolato la distanza che la mela ha percorso dopo ogni secondo, calcolando la nuova coordinata y e considerando le caratteristiche dello schermo.  illustra l'algoritmo di codifica.

```

Free_fall
delta_h: array of real
g ← -9.8
t ← 0
h_in ← 0
scale ← -2
go to (0,100)
clear()
pen(down)
stamp()

while (t < 5)
    t ← t + 1
    h ← g * t * t / 2
    y ← 100 - scale * h
    glide (0, y)
    add (delta_h, h - h_in)
    h_in ← h
    broadcast((linie)
    stamp()
    
```

 4: Problema di caduta libera

Sfida

Gli studenti modificano il Δt e il tempo di percorrenza t ; il nostro melo dovrebbe essere molto alto – probabilmente è meglio disegnare una torre oppure spostare il problema su un altro pianeta con un'altra accelerazione gravitazionale. Per creare un progetto complesso, potrebbero aggiungere la forza

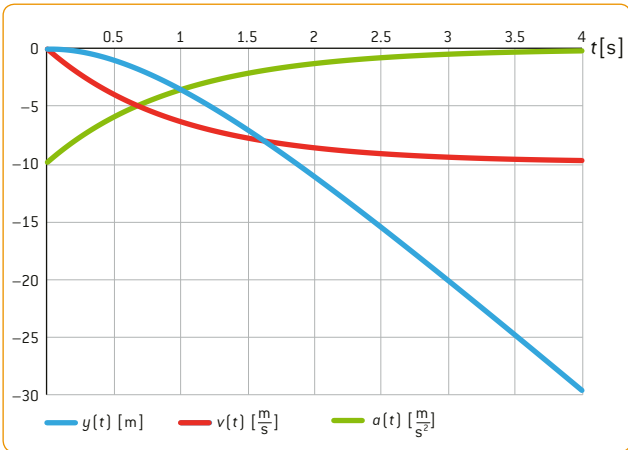
d'attrito dell'aria e considerare un'accelerazione gravitazionale variabile facendo cadere la mela da un pallone meteorologico ad una quota più alta.

Applicazione 2: Goccia d'acqua in caduta

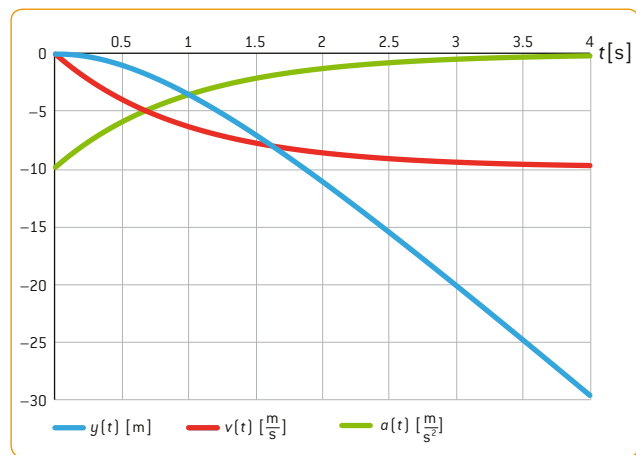
In un giorno di pioggia, tutti possono osservare la caduta di gocce d'acqua. Gli studenti hanno analizzato il movimento lineare di una goccia con la nostra simulazione. All'inizio, hanno visto che la caduta della goccia d'acqua è accelerata, ma con un'accelerazione decrescente. Dopo qualche istante, la velocità di caduta ha raggiunto il suo limite, la velocità terminale v_t , quando l'accelerazione ha raggiunto lo zero. Poi la goccia d'acqua ha continuato a muoversi a questa velocità costante. Come puoi spiegarlo?


Teoria fisica

Nella parte accelerata del movimento, due forze agiscono in direzioni opposte sulla goccia in caduta: la forza gravitazionale $G = mg$ (m : massa della goccia, g : accelerazione gravitazionale) e la forza di attrito $F_f = kv$ (k : costante di proporzionalità, v : velocità istantanea). L'accelerazione della goccia diventa: $a = g - \frac{k}{m} v$.

Nella nostra simulazione, consideriamo una grande goccia che misura circa 5 mm di diametro con la velocità terminale $v_t = 9,8 \frac{m}{s}$.^[5] In questo caso, la costante $\frac{k}{m} = \frac{1}{s}$. L'accelerazione diminuisce all'aumentare della velocità (vedi ). I valori iniziali sono $a = 9,8 \frac{m}{s^2}$, $v = 0$ and $y = 0$.

Usiamo un piccolo programma in C++^[4] per calcolare l'accelerazione e la velocità istantanea, considerando l'accelerazione e la velocità costanti per intervalli di tempo Δt (like 0,05 s). In questo caso, la velocità aumenta con $\Delta v = a\Delta t$ e la distanza percorsa con $\Delta y = v\Delta t$ per ogni Δt scelto (metodo passo a passo).



 5: Relazione tra distanza percorsa, velocità e accelerazione

Come lo codifichiamo?

1. Dipingere uno sprite goccia d'acqua.
2. Dipingere una linea orizzontale verde in basso sullo sfondo.
3. Fai uno sprite con il messaggio 'accelerazione=0', che appare quando l'accelerazione ha un valore di circa 0.
4. Scrivi il codice dello sprite goccia. La caduta inizia nel punto (0, y_{init}). Utilizzando un ciclo, si ricalcolano $a(t)$, $v(t)$, $y(t)$. Utilizziamo la distanza raggiunta dalla caduta dopo ogni Δt , elaborando la nuova *coordinata y* e tenendo conto delle caratteristiche dello schermo. L'accelerazione diminuisce e quando è circa 0, il ciclo è finito. A questo punto, il messaggio sprite viene visualizzato sullo schermo. Successivamente, lo sprite goccia cade a velocità costante fino a toccare la linea verde dello sfondo.

📄 6 fornisce una chiara spiegazione del codice.

Sfida

Gli studenti possono migliorare questa applicazione se aggiungono una variabile per la massa della goccia d'acqua (il diametro della goccia d'acqua può di solito assumere valori da 1 mm a 5 mm)^[6] e un altro tipo di forza di attrito: $F_f = \frac{kv^2}{2}$.

Gli studenti potrebbero anche aggiungere più gocce di diverse masse e confrontare come cadono.

<Applicazione 3: Urti elastici>

Ci sono molti esempi di corpi che si scontrano intorno a noi. Questi urti sono complicati; noi però consideriamo gli urti elastici di

Drop

```

g ← 9.8
kOverM ← 1
deltaT ← 0.05
eps ← 0.17
v ← 0
t ← 0
y ← 0
a ← -g + kOverM * v
vf ← -9.8

(abs(a) > eps)

t ← t + deltaT
y ← y + deltaT * v
v ← v + deltaT * a
a ← -g + kOverM * v

y ← y + deltaT * vf

until (touch (ground))
    
```

📄 6: Caduta della goccia

palle da biliardo o d'acciaio, applicabili nella vita reale, o urti teorici di molecole, come quelle studiate nel modello di gas ideale.

Teoria fisica

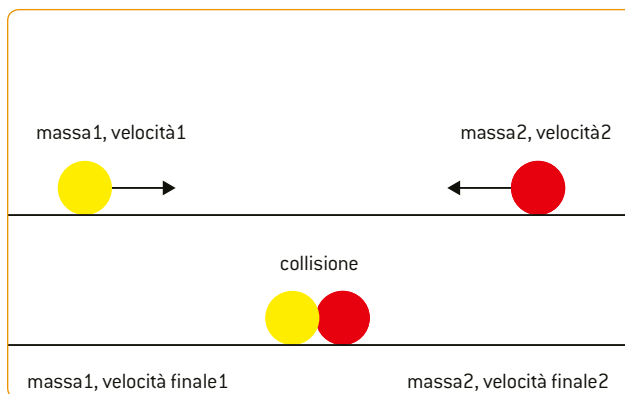
La quantità di moto lineare e l'energia cinetica sono conservate per due sfere con la massa m_1 e m_2 , le velocità iniziali (\vec{v}_1) e (\vec{v}_2) e le velocità finali (\vec{v}_{1f}) e (\vec{v}_{2f}). [7]

$$m_1 \vec{v}_1 + m_2 \vec{v}_2 = m_1 \vec{v}_{1f} + m_2 \vec{v}_{2f} \text{ and}$$

$$\frac{1}{2} m_1 v_1^2 + \frac{1}{2} m_2 v_2^2 = \frac{1}{2} m_1 v_{1f}^2 + \frac{1}{2} m_2 v_{2f}^2$$

Se tutti i movimenti avvengono lungo la stessa linea (movimento sull'asse x), possiamo usare i segni + o - per indicare le direzioni. La notazione vettoriale non è necessaria per il caso di urto in linea retta, e le velocità finali possono essere calcolate con le seguenti equazioni:

$$v_{1f} = 2 \frac{m_1 v_1 + m_2 v_2}{m_1 + m_2} - v_1 \text{ and } v_{2f} = 2 \frac{m_1 v_1 + m_2 v_2}{m_1 + m_2} - v_2.$$



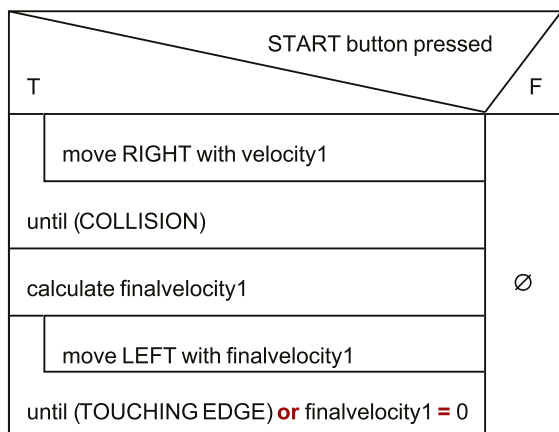
📄 7: Collisione elastica

Come lo codifichiamo?

1. Scegliere due sprite per le palle (Ball1 e Ball2) e uno sprite per il pulsante **START** (Start sprite).
2. Variabili usate: *massa1*, *massa2*, *velocità1*, *velocità2* (la massa e la velocità iniziale) per ogni oggetto. Rendere visibili i cursori delle variabili e impostarne il valore minimo e massimo.
3. Inserire la massa e le velocità iniziali per ogni oggetto.
4. Premere il pulsante **START** (Avvio). In questo momento, lo sprite trasmette un messaggio per gli sprite palla. Quando ricevono il messaggio, ogni pallina si muove verso l'altra utilizzando la ben nota formula $distanza = velocità \times tempo$.
5. Calcolare le velocità finali delle palle e usarle per spostare le palle nella giusta direzione fino a quando una palla o tocca il bordo e lascia la scena o rimane al suo posto perché la sua nuova velocità è 0.

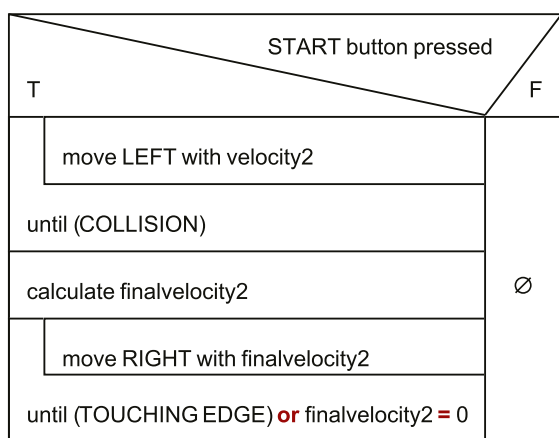
📄 8 e 9 forniscono una chiara panoramica di come le due sfere sono animate in Scratch^[1].

Elastic_Collision_Ball1



© 8: Urto elastico per Ball1

Elastic_Collision_Ball2



© 9: Urto elastico per Ball2

Due esempi di utilizzo di questa applicazione:

1. Scegliete una velocità 0 e la massa uguale per le palle; dopo questo urto, noterete che la palla in movimento si ferma e l'altra si muove con la stessa velocità che la prima palla aveva prima dell'impatto.
 2. Le sfere hanno velocità diverse e massa uguale; dopo l'urto, vedrete che gli oggetti si scambiano la velocità.
- In entrambi gli esempi, le sfere scambiano la loro quantità di moto.

Sfida

Gli studenti potrebbero cambiare la dimensione delle palle e renderla direttamente proporzionale alla loro massa; potrebbero fare un'applicazione per una collisione elastica bidimensionale (simulazione dell'effetto Compton) o programmare una simulazione per la collisione di una palla con una parete (legge della riflessione in meccanica).

Si potrebbe continuare lo studio con un'altra applicazione, per esempio una collisione anelastica.^[4]

<Conclusione>

<Per gli studenti>

Vantaggi

Gli studenti hanno imparato la fisica in modo più divertente e sono stati in grado di comprendere meglio i fenomeni naturali utilizzando le simulazioni in Scratch. Hanno approfondito le loro conoscenze informatiche e fisiche allo stesso tempo. Anche se i loro progetti non erano tutti perfetti, gli studenti hanno chiaramente migliorato le loro capacità di programmazione e di pensiero algoritmico come risultato.

Svantaggi

Gli studenti lavoravano da soli e di più a casa. Hanno ricevuto feedback a scuola.

<Per gli insegnanti>

Vantaggi

Abbiamo osservato un reale interesse nel creare un'applicazione originale e nell'imparare più che nelle lezioni classiche.

Svantaggi

È stato difficile per noi coordinare l'intera classe a causa della grande varietà di argomenti fisici e dei bug molto specifici di ogni applicazione. Riteniamo che sarebbe meglio dare a tutti gli studenti lo stesso argomento e incoraggiarli a migliorarlo il più possibile e al meglio dei loro livelli di abilità.

<Attività cooperativa>

Studenti provenienti da scuole e paesi diversi potrebbero affrontare le sfide dei progetti e crearne di nuovi con altre idee legate all'argomento originale. Tutte queste applicazioni potrebbero essere messe nello stesso posto sulla piattaforma Scratch, e quindi si potrebbe organizzare un concorso per selezionare le migliori. Gli insegnanti devono anche tenere conto della complessità della programmazione e della fisica quando valutano il lavoro dei loro studenti.

<Riferimenti>

[1] <https://scratch.mit.edu/>
 [2] Tutti i materiali aggiuntivi sono disponibili all'indirizzo www.science-on-stage.de/coding-materials.
 [3] <https://en.scratch-wiki.info/>
 [4] https://scratch.mit.edu/users/SonS_Coding
 [5] <http://hypertextbook.com/facts/2007/EvanKaplan.shtml> (29/11/2018)
 [6] <https://journals.ametsoc.org/doi/pdf/10.1175/1520-0450%281969%29008%3C0249%3ATVO-RA%3E2.0.CO%3B2> (29/11/2018)

Science Magic Box

<Autoce> Luc Ivacca

<Autoce> Marco Nicolini



<Info>

<Parole chiave> microcontrollore, trasduttore, sensore, attuatore, segnale, grandezza fisica, ciclo, diramazione logica, processo sequenziale, calibrazione, ingresso, uscita, lettura, scrittura, analogico, digitale, linearità, conversione di grandezze fisiche, breadboard (basetta), pin, saldatura, interfaccia uomo-macchina

<Discipline> fisica, elettronica, matematica, informatica, logica, biologia

<Età degli studenti> 14–18

<Hardware> Arduino UNO^[1] with Arduino DUE^[2] e/o TI-Nspire CX CAS con TI-Innovator Hub

<Linguaggi> C++ (using Arduino IDE^[3]) e/o TI-Basic

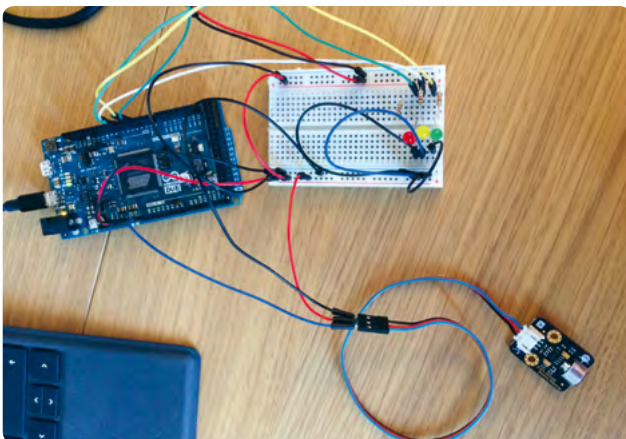
<Livello di difficoltà di programmazione> medio, con una sezione audio per studenti avanzati

Un elenco delle abbreviazioni, dei termini speciali e degli acronimi utilizzati in questa unità è disponibile online.^[4]

<Sommario>

Gli studenti imparano a programmare in un ambiente hardware-software autocostuito (basato su Arduino) e con un computer tascabile pronto all'uso (calcolatrice TI-Nspire CX CAS con la sua estensione, il TI-Innovator Hub). Entrambi sono utilizzati come dispositivi per la raccolta, la conversione e la trasduzione dei dati raccolti da sensori per gestire, leggere, convertire ed attuare grandezze fisiche.

Utilizzando la piattaforma Arduino, gli studenti codificano un programma in un ambiente dotato di sensori che acquisiscono grandezze fisiche come segnali di ingresso, e attuatori che reagiscono all'acquisizione e producono un segnale di uscita come grandezza fisica dopo che un microcontrollore ha elaborato il segnale rilevato per impostare l'uscita opportunamente (vedi ①).



①: Scheda Arduino

Il TI-Innovator Hub è un box "pronto all'uso" che consente agli studenti di apprendere le basi della programmazione. Deve essere interfacciato ad una calcolatrice TI-Nspire CX CAS. Ha una buona interfaccia I/O, che include un sensore di luminosità, due LED e un buzzer integrato, capace di produrre un suono di una frequenza voluta (vedi ②).



②: Calcolatrice TI-Nspire e TI-Innovator Hub

<Introduzione concettuale>

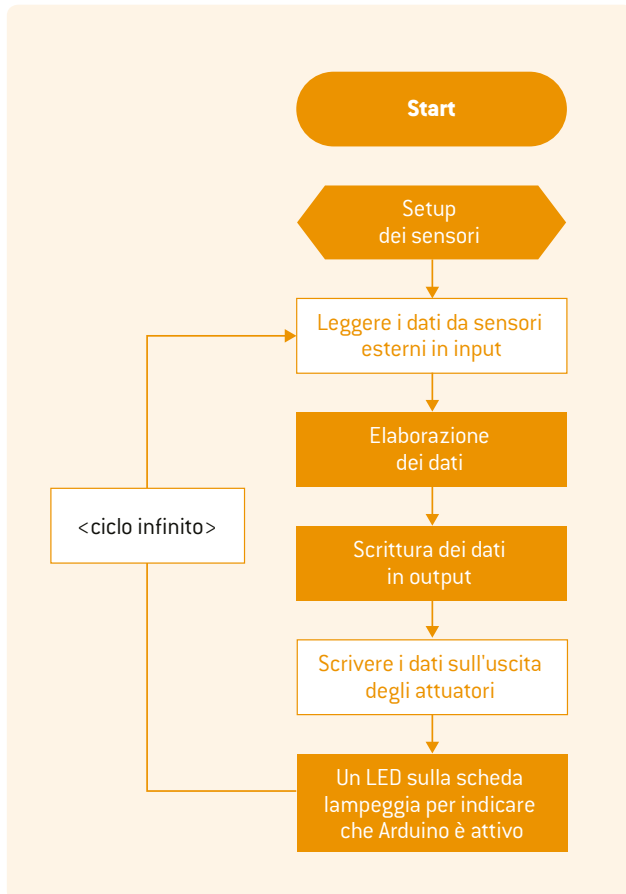
L'unità introduce gli studenti al mondo della codifica dei problemi fisici, basato sul rilevamento e la misurazione delle grandezze fisiche, l'elaborazione dei dati, la reazione e la decisione di eseguire un'azione mediante gli attuatori a disposizione.

Il codice è normalmente basato su un loop infinito (la macchina è di solito 'viva' per tutto il tempo in cui è alimentata, e deve funzionare sempre), dove le tre azioni di lettura/misurazione, elaborazione dati ed azione sugli output sono eseguite nell'ordine.

Gli studenti impareranno che possono scrivere qualsiasi codice con

1. istruzioni sequenziali
2. loops (mentre ... fare; ripetere ... fino a che)
3. ramificazione (se... allora... altrimenti)

come indicato nel teorema di Böhm-Jacopini (cfr. 'Informazioni aggiuntive'^[4]).



© 3: Diagramma di flusso

Il secondo obiettivo dell'unità è quello di presentare i microcontrollori agli studenti. Impareranno a configurare i sensori di ingresso e gli attuatori di uscita, utilizzando porte digitali o analogiche, e a scrivere un semplice programma che legge gli ingressi, elabora i dati e scrive gli output. Possono scegliere se emettere segnali sonori o luminosi. Questi possono essere interpretati come "allarmi" che emettono un avviso in base all'ingresso letto dai sensori.

Gli studenti utilizzano l'ambiente di sviluppo integrato Arduino^[1] (Arduino IDE^[3]) per programmare in C++ con librerie di funzioni pronte all'uso che rendono il processo di codifica più facile e veloce.

Deve essere disponibile una breadboard (vedi 'Informazioni aggiuntive'^[4]) per consentire agli studenti di prototipare e collegare facilmente i pin dei sensori agli I/O Arduino, all'alimentatore a 5V e al pin GND.

La struttura di qualsiasi programma, in metalinguaggio, è mostrata in ©3.

Notate che la meta-istruzione '*While [TRUE]*' è un trucco per dire a qualsiasi processore di ripetere le istruzioni incluse a tempo indeterminato (finché il microcontrollore è alimentato).

Il terzo obiettivo principale è quello di imparare a convertire una grandezza fisica acquisita in un'altra quantità (ad esempio, l'intensità della luce in suono), pronta per la trasmissione all'ambiente esterno, seguendo questi passaggi:

1. Il segnale fisico (luce, suono, forza, energia) viene acquisito dal sensore e convertito in un segnale elettrico.
2. Il segnale elettrico viene trasformato in un numero a disposizione del processore.
3. Il numero viene elaborato e trasformato dal processore in un altro numero e quindi utilizzato per eseguire un'azione con un trasduttore attuatore.
4. Gli attuatori convertono il numero in segnali elettrici pronti per essere emessi.
5. Il segnale elettrico viene infine trasformato in un segnale fisico (es. suono, luce).

In 1 e 5, i segnali devono essere convertiti da una forma all'altra. In queste fasi, la linearità della trasformazione, o la dipendenza "prossima alla linearità", è estremamente importante (vedi 'Informazioni aggiuntive'^[4]).

Vedere 'Informazioni aggiuntive'^[4] per una spiegazione dettagliata dell'ultima riga della metacodifica ('LED lampeggiante a bordo').

Il segnale di ingresso è solitamente chiamato "stimolo" e proviene dall'ambiente in cui sono collocati i sensori per ottenere i dati. Il processore e il codice sono progettati per "reagire" allo stimolo con operazioni matematiche/logiche (eseguite dalle istruzioni del codice, elaborate dal microcontrollore) e per fornire questa "risposta" all'ambiente. Nelle nostre attività, l'ambiente è lo spazio che circonda l'Arduino, che è in grado di "vedere", "sentire" e "sentire le forze" grazie ai sensori.

Lavorare invece con TI-Innovator Hub permette agli studenti di concentrarsi maggiormente sulla parte di codifica del loro lavoro, poiché i microcontrollori e i sensori sono già impostati e pronti all'uso.

<Cosa fanno gli studenti/insegnanti>

Vi consigliamo di iniziare con un brainstorming per raccogliere le idee originali dei vostri studenti sui sensori e sul controllo automatico della macchina. Raccogliere queste idee, fare esperienza pratica con i sensori e il controllo automatico della macchina e poi confrontare i risultati con le loro idee precedenti (anche se errate) è un buon modo per aiutare gli studenti a comprendere veramente il processo.

L'insegnante potrebbe preparare un modulo con domande come:

- ↳ Sai come un termostato controlla la temperatura in una stanza?

- ↳ A cosa serve un dispositivo di parcheggio auto basato sull'allarme acustico? Come reagisce il conducente quando il suono viene emesso dal sistema?
- ↳ Hai un piano cottura a induzione in cucina? Che cosa significa un LED illuminato?

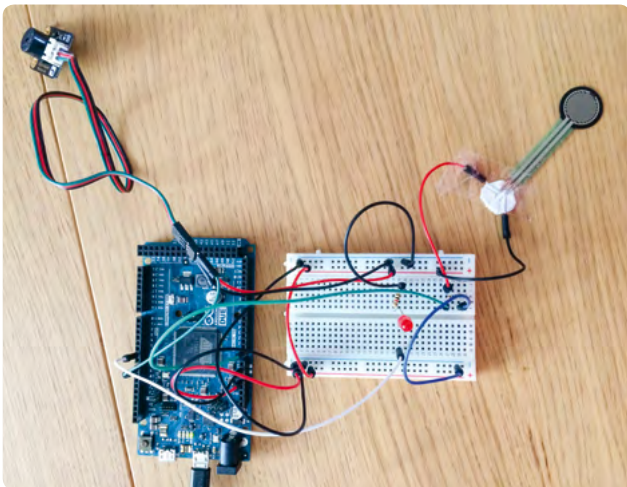
Si veda 'PEC ed elenco di domande'^[4] per un esempio di un elenco completo di domande e riferimenti alla metodologia di insegnamento della PEC (Prevision, Experience, and Correction).

<Fase teorica con Arduino^[4]>

L'insegnante introdurrà la programmazione C++^[3] con la struttura e le istruzioni di base in modo che gli studenti possano scrivere un semplice ciclo usando le istruzioni *analogRead*, *digitalRead*, *analogWrite*, *digitalWrite*, *if...then...else*, *loop*, *while*.

Parte hardware

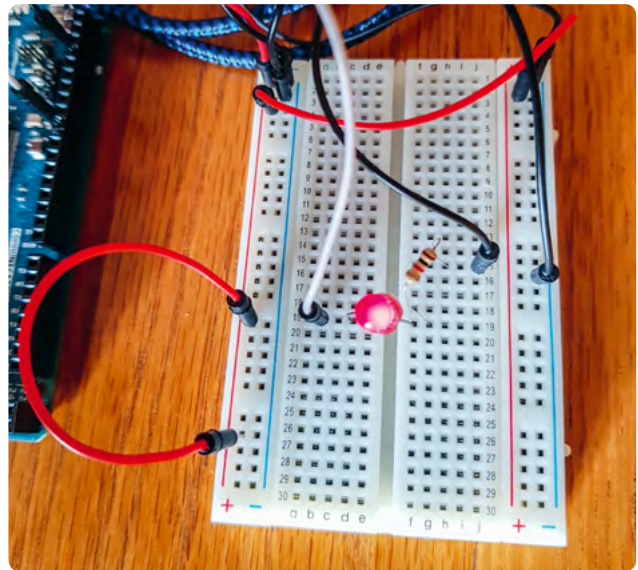
L'insegnante presenterà il layout del microcontrollore, mostrando il microprocessore, i pin degli I/O analogici (connessioni) e i pin di ingresso/uscita digitale (connessioni).



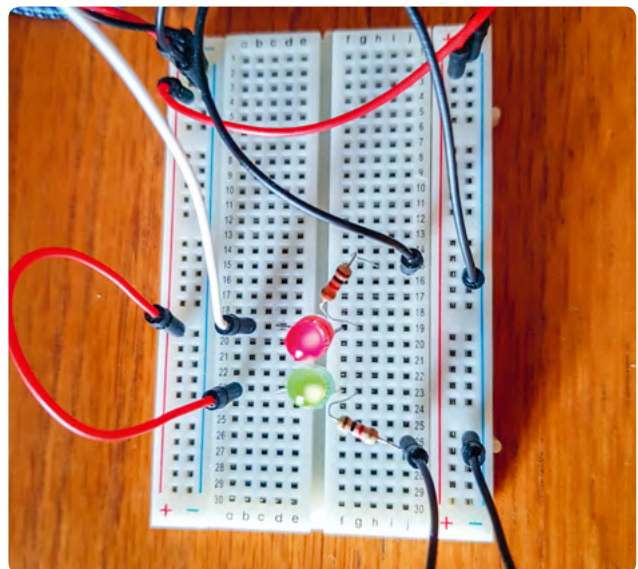
© 4: Arduino, breadboard e sensori

Gli studenti impareranno che ogni sensore/attuatore ha solitamente connessioni multiple:

- ↳ all'uscita 5V o 3.3V di Arduino per ottenere l'alimentazione elettrica
- ↳ al segnale GND (massa) in modo che una corrente possa fluire e
- ↳ ad un altro pin di ingresso digitale o analogico se viene utilizzato per leggere (ottenere) dati esterni oppure
- ↳ ad un altro pin di uscita digitale o analogico se viene utilizzato per compiere azioni che generano un' uscita, ad esempio emettono un suono o una luce, o fare qualsiasi altra cosa che segnali una situazione (un'operazione di "scrittura").



© 5: Dettaglio breadboard



© 6: Breadboard con LED

Parte software

L'insegnante presenta un semplice programma che legge un sensore e scrive un attuatore, dove una chiara associazione tra i pin fisici e l'indirizzo fisico a bordo del microcontrollore può essere facilmente stabilita dagli studenti.

Un esempio di istruzioni pronte all'uso è disponibile online [Program example 1^[4]].

Gli studenti devono tenere presente che il processore esegue le istruzioni del codice una dopo l'altra e nell'ordine in cui sono scritte. Solo l'istruzione "loop" altera questo ordine in quanto dice al processore di ripetere continuamente le istruzioni tra parentesi finché il microcontrollore è alimentato.

<Fase pratica con Arduino>

Gli studenti fanno un'esperienza pratica con il microcontrollore, la breadboard e i sensori. Il docente dovrebbe presentare la struttura della breadboard, mostrando tutte le connessioni disponibili e come gli studenti possono portare i segnali 5V, GND e I/O dall'Arduino^[4] alla breadboard. Gli studenti saranno invitati a copiare l'esempio di codifica e a provare, testare ed eseguire il debug del codice con gli I/O collegati.

Parte hardware

Gli studenti hanno bisogno del microcontrollore, dei sensori e di cavetti (10 cm) per consentire facili collegamenti tra i pin dei sensori e i "buchi" della breadboard. A volte potrebbe essere necessario saldare cavi aggiuntivi ai sensori, ma molti sensori non lo richiedono.

Gli studenti utilizzeranno i cavetti per collegare l'alimentazione a 5V e il segnale GND alla breadboard, e i pin analogico/digitale dell'Arduino^[4] ad alcuni "buchi" sulla breadboard. In questo modo i sensori possono essere facilmente posizionati sulla breadboard e ricevere i segnali elettrici necessari (vedi 6).

Parte software

Dopo aver verificato che le connessioni tra microcontrollore e breadboard siano state impostate correttamente, con l'esatta corrispondenza tra il numero logico del pin del microcontrollore e il pin del sensore sul breadboard, gli studenti proveranno a scrivere il semplice codice fornito ('Program example 1'^[4]).

Algoritmi con Arduino

Negli esempi abbiamo preparato diverse conversioni di segnali da una forma fisica ad un'altra.

Conversione di un segnale luminoso analogico in un segnale luminoso digitale in un LED (modulato con funzione PWM) e un suono: la frequenza sonora emessa aumenta con l'intensità della luce. Applicazione pratica: sveglia, assistenza per non vedenti. (Vedi 7)



7: Un sensore di luce

Conversione di una forza rilevata attraverso un sensore di forza in un segnale luminoso digitale in un LED (modulato con funzione PWM) e un suono: l'intensità luminosa aumenta con l'intensità della forza. Applicazione pratica: allarme di carico del peso. (Vedi 8)



8: Un cicalino digitale

Conversione di un segnale di rumore esterno in un segnale luminoso digitale in un LED. Più alto è il suono, più alta sarà la frequenza della luce. Applicazione pratica: controllo dell'inquinamento acustico. Conversione di una distanza misurata con un sensore di distanza in un suono. Applicazione tecnica: sensori di parcheggio auto. (Vedi 9)



9: Un sensore di distanza

Conversione di un segnale di temperatura in un segnale sonoro e un segnale luminoso. Applicazione tecnica: controllo della temperatura del forno.



10: Un sensore di umidità del suolo

Vedere 'Program example 2'^[4] per il codice utilizzato per queste applicazioni sulla scheda Arduino.

Tutti questi algoritmi servono come sistemi di controllo del segnale e di allarme, e monitorano un ambiente selezionato su una grandezza fisica emettendo un avviso in base all'ingresso (stimolo) letto.

<Fase teorica con il TI-Innovator Hub>

Parte hardware

L'insegnante può mostrare agli studenti quanto sia facile collegare i sensori.

Parte software

La programmazione di base può essere effettuata con la sola calcolatrice; gli studenti devono solo padroneggiare le istruzioni TI-Basic fornite in precedenza. Poi l'hub può essere collegato e gli studenti impareranno a comunicare con esso, cioè ad usare le istruzioni 'leggi' e 'ricevi' per acquisire dati e 'imposta' per controllare le uscite.

<Fase pratica con il TI-Innovator Hub>

Parte hardware e software

Gli studenti inizieranno con esempi di base per familiarizzare con l'hub prima di lavorare su problemi più aperti. Gli studenti impareranno a controllare le diverse uscite con piccoli esercizi, ad esempio controllando il colore del LED, facendolo lampeggiare, controllando la durata del lampeggio e producendo suoni di una data frequenza. Il loop infinito sarà di nuovo la struttura di base per continuare le operazioni a tempo indeterminato.

Algoritmi con il TI-Innovator Hub

A questo punto gli studenti possono risolvere come compito due problemi aperti: la programmazione di un interruttore automatico che accende la luce solo se l'intensità della luce ambientale è inferiore ad una certa soglia, e la programmazione di una sveglia, che emette un suono di frequenza crescente all'aumentare della luce ambientale. Sono possibili ulteriori sviluppi, ma sarà necessario acquistare e inserire sensori supplementari nell'Hub.

<Acquisto dei sensori>

Informazioni su dove e come acquistare i sensori sono disponibili online.^[4]

<Conclusione>

Alla fine di queste attività, abbiamo notato che la comprensione della codifica di un programma da parte dei nostri studenti era migliorata in modo significativo, così come la comprensione della struttura generale del programma, della logica e degli algoritmi.

<Attività cooperativa>

Una meravigliosa attività di cooperazione potrebbe essere quella di promuovere l'autoimprenditorialità. Gli studenti potrebbero cercare di inventare un'originale interfaccia uomo-macchina (HMI) tecnicamente utile. Questo HMI dovrebbe leggere uno stimolo proveniente da un ambiente (atmosfera, casa, corpo umano, ecc.) e reagire producendo un altro segnale che emette un avvertimento, esegue un'azione o segnala una situazione. Le scuole partner all'estero potrebbero effettuare un'indagine di mercato per valutare la domanda di mercato e il valore che il dispositivo può avere nei loro paesi. Ogni scuola che partecipa a questo scambio potrebbe inventare un dispositivo e fare ricerche di mercato per il prodotto assemblato dalle altre scuole. Alla fine del progetto, lo strumento più popolare potrebbe essere prodotto su piccola scala da un'azienda partner e venduto. L'autoimprenditorialità è molto apprezzata in tutto il mondo, in quanto offre una grande opportunità per insegnare insieme scienza, tecnologia e materie finanziarie.

<Riferimenti>

- [1] www.arduino.cc
- [2] www.arduino.cc/en/Guide/ArduinoDue
- [3] www.arduino.cc/en/Main/Software
- [4] Tutti i materiali aggiuntivi sono disponibili all'indirizzo www.science-on-stage.de/coding-materials.

CoALA-Code a Little Animal

<Autore> Mirek Hančl

<Autore> Julia Winckler



<Info>

<Parole chiave> simulazione, modello IPO (input-processing-output), misura, pensiero computazionale

<Discipline> scienza, biologia, informatica

<Età degli studenti> 9-13

<Hardware> Calliope mini^[1] o BBC micro:bit^[2]

<Laboratorio A> morsetti-clip a coccodrillo, plastica rossa artigianale (red craft plastic), cavo USB e batteria per il Calliope mini, nastro di rame autoadesivo (5 mm), cartone, colla, forbici, piccolo bicchiere d'acqua, poster con immagini di animali

<Laboratorio B> morsetti-clip a coccodrillo, cavo USB e batteria per il Calliope mini, sensore di umidità Grove, sensore tattile Grove I2C, Grove NFC, hub Grove I2C^[3], cartone, plastica rossa artigianale, piccolo bicchiere d'acqua, poster con immagini di animali

<Linguaggi> MakeCode^[4]

<Livello di difficoltà di programmazione> facile

<Sommario>

E' difficile trovare un bambino che non sia desideroso di possedere un animale domestico; con questa attività chiunque avrà la possibilità di realizzarne uno. Per scoprire qual è l'animale migliore, gli studenti costruiranno un simulatore controllato da un computer a scheda singola e che utilizza sensori esterni per imitare le esigenze di un animale domestico.

<Introduzione concettuale>

L'argomento degli "animali da compagnia" non solo fa parte del curriculum nelle scuole primarie ma si trova anche in quelli delle scuole secondarie di biologia, dove ad esempio gli studenti imparano come in passato siano stati ottenuti i cani dai lupi, oppure i bisogni di base di un animale domestico, le sue necessità e come soddisfarle. In genere, gli studenti analizzano testi nei loro libri di scuola oppure i video su Internet perché per le scuole non è facile fornire gli animali domestici necessari. Pertanto, un simulatore elettronico capace di replicare i bisogni e i comportamenti di base di un animale domestico (in termini di cibo, acqua, esercizio fisico, coccole-carezze e temperatura corporea) potrebbe essere sia illustrativo che istruttivo.

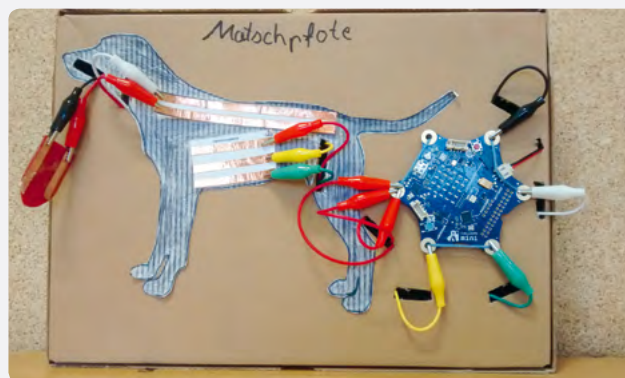
Il progetto CoALA non utilizza dispositivi elettronici già pronti all'uso e realizzati da produttori commerciali di materiale didattico, che consentono di svolgere solo programmi specifici. Né viene utilizzato un semplice giocattolo, come il Tamagotchi ad esempio, che ha riscosso un successo mondiale negli anni Novanta. In questa attività gli studenti progettano, costruiscono e programmano il proprio simulatore sotto forma del pro-

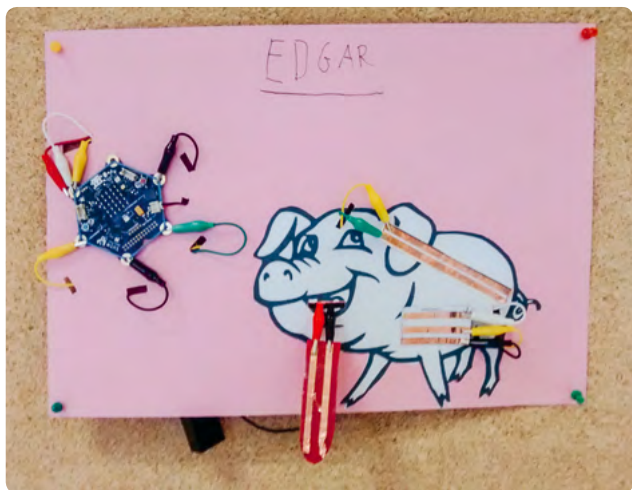
prio animale domestico preferito dotato di un'immagine, e composto da un computer a scheda singola (nel nostro caso, un Calliope mini^[1] o un BBC micro:bit^[2]) e materiali d'uso comune come cartone, nastro di rame e sensori esterni. Gli studenti programmano un algoritmo per registrare e valutare i bisogni fondamentali dell'animale scelto. A seconda dell'algoritmo, il simulatore dell'animale mostra differenti smiley (per mostrare come si sente l'animale) o suona melodie composte dagli studenti.

Il concetto del progetto didattico 'CoALA-Code a Little Animal' è quello di un laboratorio. Il materiale didattico OER (Open Educational Resources) è composto da tre parti. La prima parte introduce agli studenti le basi degli algoritmi di programmazione e la gestione del computer a scheda singola Calliope mini^[1]. Nella seconda parte, gli studenti esplorano i bisogni di base di un animale domestico e come valutarli. Nella terza parte del workshop, gli studenti costruiscono il loro animale domestico preferito con del cartone, installano sia il computer a scheda singola che gli opportuni sensori e creano, infine, gli algoritmi necessari con l'ausilio di un linguaggio grafico di programmazione.

Per soddisfare i requisiti del curriculum scientifico di biologia delle scuole primarie e delle scuole secondarie, presentiamo i materiali del laboratorio in due versioni. Per le scuole elementari (versione A), i tassi di consumo di cibo, acqua e coccole-carezze sono misurati e registrati utilizzando del nastro di rame conduttivo ed adesivo. Per le scuole secondarie (versione B), gli studenti utilizzano sensori esterni per misurare l'umidità (bere), per le interazioni multi-touch (coccole-carezze) e per la lettura wireless dei chip di Near Field Communication (mangiare). In entrambe le versioni, i sensori integrati misurano sia il movimento che la temperatura.

Tutti i materiali del laboratorio e gli esempi di codice per l'ambiente di programmazione MakeCode^[4] sono disponibili per il download online.^[5]

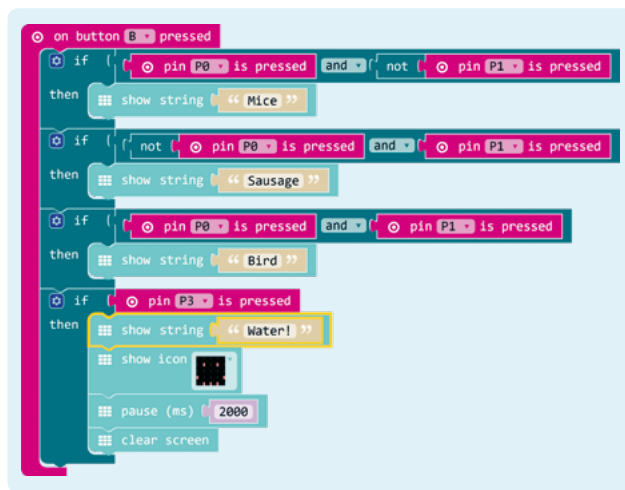




<Cosa fanno gli studenti/insegnanti>

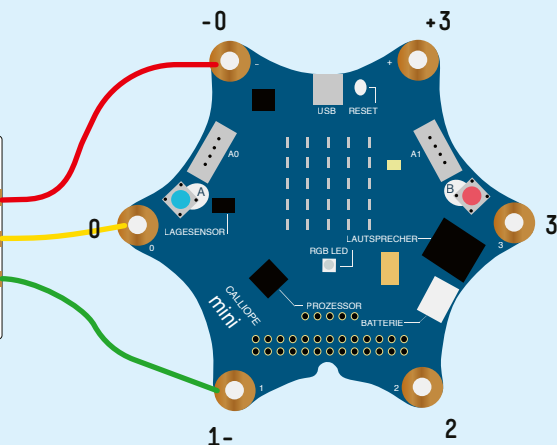
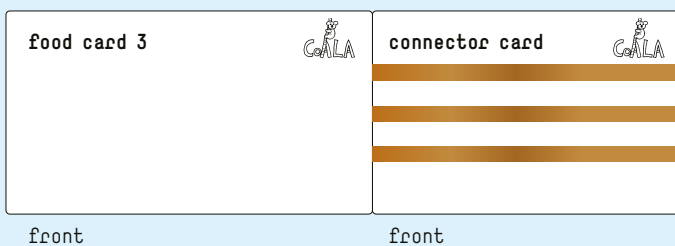
Per costruire il simulatore di animali domestici, gli studenti cercano un'immagine del loro animale domestico preferito oppure scattano una foto. L'immagine stampata viene incollata sul cartone e dotata di nastro adesivo in rame (versione A) o di sensori esterni (versione B) nei punti appropriati. Il nastro adesivo in rame o i sensori esterni sono collegati alle connessioni del computer a scheda singola e sul computer è presente un programma idoneo a renderlo "intelligente". Nell'esempio che segue, la necessità di base "cibo" viene utilizzata per spiegare come si differenziano le due versioni del workshop e come viene utilizzato l'ambiente di programmazione.

Quando l'animale domestico simulato si nutre, i sensori di gusto ovviamente non possono essere utilizzati. Invece, il sensore appropriato "legge" l'alimento offerto e l'algoritmo è controllato da una opportuna sequenza di istruzioni condizionata, in modo che l'uscita corrisponda al comportamento atteso dell'animale domestico. Pertanto, il display del simulatore di gatto mostra, ad esempio uno smiley sorridente quando viene alimentato con un topo e uno triste quando riceve un osso. Questi rami sono gli stessi per entrambe le versioni di questo laboratorio.

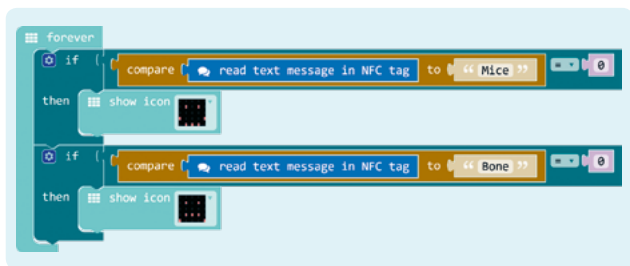


Tuttavia, i sensori alimentari sono completamente diversi. Nel laboratorio A, le immagini dei diversi tipi di cibo sono riportate su delle carte di cartone. Sull'altro lato di ogni carta vi è incollato del nastro di rame in modo che la "lingua" del simulatore legge un numero di codice binario quando viene tenuta in mano. Poiché le connessioni del "lettore", cioè della "lingua", sono collegate ai diversi pin del computer a scheda singola, l'algoritmo può verificare direttamente se i pin sono cortocircuitati o meno: le food card cortocircuitano combinazioni diverse di pin.

Nella versione B, un sensore esterno con chip NFC e antenna radio collegata viene utilizzato per leggere in modalità wireless le stringhe da un tag NFC. Questo tag può essere posto su un'etichetta adesiva oppure in una chip card. A differenza del laboratorio A, non viene letto alcun numero di codice binario, ma il nome di un alimento come "pesce" oppure "osso". Ciò aumenta notevolmente le opzioni di attuazione e la complessità. Nell'algoritmo, il ramo di istruzioni condizionale è controllato confrontando il valore letto con le stringhe fornite. Nel progetto CoALA, i tag NFC vengono scritti su un'applicazione per smartphone; la lettura del tag NFC viene didatticamente ridotta ad



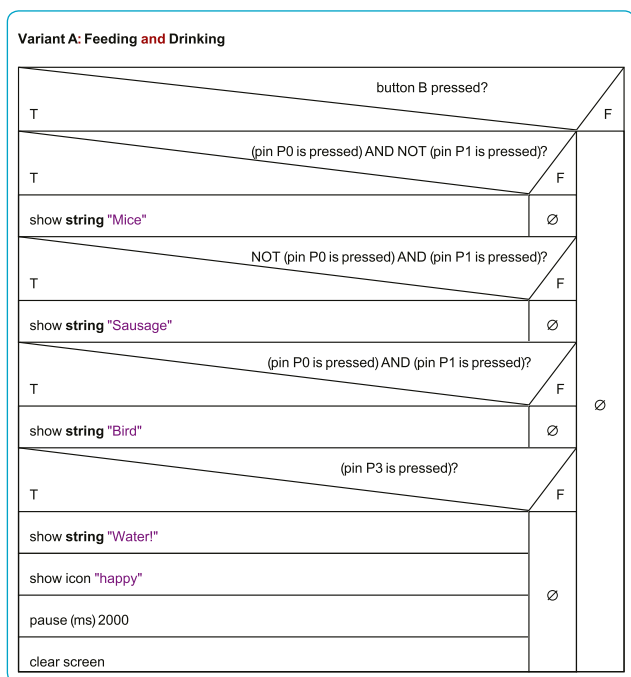
un unico blocco di codice in MakeCode^[4] e quindi caricata come estensione dell'ambiente di programmazione.



<Algoritmo da usare con altri linguaggi>

Gli esempi di codice disponibili^[5] possono essere caricati nell'editor di programmazione online MakeCode^[4] e quindi utilizzati direttamente. Passando dalla vista a blocchi alla vista testo, il codice sorgente viene convertito in JavaScript e può quindi essere facilmente utilizzato in altri linguaggi di programmazione per Calliope mini^[1] o BBC micro:bit^[2]. Le estensioni di programmazione per MakeCode utilizzate nei materiali di laboratorio per il controllo del multitouch e i sensori NFC funzionano anche per il BBC micro:bit.

Infine, gli esempi di programmazione sono forniti sul sito web come diagrammi di struttura in modo che gli algoritmi possano essere facilmente compresi e trasferiti su altre piattaforme e ambienti di programmazione come Arduino.



<Conclusione>

Il progetto CoALA offre agli studenti l'opportunità di familiarizzare con i concetti fondamentali degli algoritmi –dichiarazioni, sequenze, ramificazioni condizionali, loop e variabili. Non li imparano semplicemente memorizzandoli e riproducendoli,

ma lavorando ad un progetto educativo entusiasmante con applicazioni nella vita reale. Entrambe le versioni del laboratorio presentano parti che possono essere mescolate.

I materiali del laboratorio sono stati testati con successo con un Calliope mini^[1] e con il computer a scheda singola BBC micro:bit^[2]. Per utilizzare i sensori esterni Grove per la misura dell'umidità, per NFC o per il multitouch della versione B con il BBC micro:bit^[3] è necessario disporre di una aggiuntiva ed economica scheda di estensione.

<Attività cooperativa>

Il laboratorio CoALA può essere utilizzato in varie forme di cooperazione didattica. Siccome il materiale è destinato sia alle scuole primarie che alle secondarie, si potrebbe anche svolgere attività di scambio tra diversi tipi di scuole. Sarebbe gratificante non solo per gli studenti, ma anche per gli insegnanti coinvolti. I materiali della versione A del laboratorio sono progettati per insegnare semplici condizioni logiche Sì/No, mentre i materiali della versione B sono progettati per insegnare condizioni, variabili e operazioni di stringa più complesse e combinate.

I materiali di entrambi i laboratori possono anche essere combinati in vario modo e in base alle necessità didattiche per promuovere la cooperazione all'interno di gruppi di apprendimento eterogenei. Gli studenti che hanno bisogno di maggiore supporto potrebbero, ad esempio, utilizzare i semplici sensori del laboratorio A, mentre quelli più capaci potrebbero spiegare i sensori del laboratorio B ai loro compagni di classe e mettere così in pratica le loro abilità comunicative.

Durante la realizzazione del progetto CoALA si è svolta una cooperazione transnazionale tra due scuole secondarie in cui due gruppi di studenti –uno tedesco e uno spagnolo –hanno discusso le loro esperienze con i simulatori di animali domestici tramite videoconferenza. In queste situazioni i due gruppi si sono scambiati i suggerimenti per la risoluzione dei problemi, il vocabolario dei nomi e delle esigenze di base dei loro animali oltre a indicazioni ed istruzioni per la programmazione.

<Riferimenti>

- [1] <https://calliope.cc/en>
- [2] www.microbit.co.uk/home
- [3] Se si utilizza un BBC micro:bit, è necessario anche un Grove Shield per micro:bit.
- [4] <https://makecode.calliope.cc/?lang=en> o <https://makecode.microbit.org/?lang=en>
- [5] Tutti i materiali aggiuntivi sono disponibili all'indirizzo www.science-on-stage.de/coding-materials.

Dati Liquididi

<Autore> Eleftheria Karagiorgou

<Autore> Sevasti Tsiliki



<Info>

<Parole chiave> fisica computazionale, acidità, acqua, liquido, temperatura, pH, registrazione dei dati

<Discipline> chimica

<Età degli studenti> 16

<Hardware> Arduino starter kit^[1], interfaccia di raccolta dati (data-logging shield), sensore di temperatura, sensore di pH, scheda SD

<Linguaggi> Arduino IDE – Wiring C^[2]

<Livello di difficoltà di programmazione> medio

<Durata del progetto> 7 ore di insegnamento

<Sommario>

Questa unità utilizza un approccio interdisciplinare per la didattica dell'informatica e della chimica. Gli studenti assumono il ruolo di ricercatori e conducono un esperimento per determinare se esiste una relazione tra l'acidità e la temperatura dell'acqua. La risposta richiederà l'impiego del controllore Arduino e delle loro conoscenze di chimica.

<Introduzione concettuale>

Questa attività didattica è stata realizzata per dimostrare agli studenti che l'informatica può essere integrata nell'educazione STEM, e più specificamente nella chimica, utilizzando metodi di insegnamento innovativi. È stata realizzata come attività extrascolastica durante lo svolgimento del club di Robotica e STEM della nostra scuola, che si tiene per due ore ogni domenica pomeriggio.

Gli studenti hanno assunto il ruolo di ricercatori e sono stati incaricati di dimostrare il ruolo della temperatura nella misurazione del pH. Con l'aumento della temperatura, infatti, aumentano anche le vibrazioni molecolari, che permettono all'acqua di ionizzarsi e di formare quindi più ioni idrogeno; di conseguenza diminuisce anche il valore di pH.

<Metodo di insegnamento>

Educazione scientifica basata sull'indagine (IBSE Inquired Based Science Education): motivata dal fatto che volevamo coinvolgere i nostri studenti in un progetto di apprendimento attivo, basato su domande che poi a loro volta generano altre domande man mano che gli studenti progrediscono nel progetto di ricerca. In questo modo, gli studenti diventano ricercatori e imparano svolgendo un'attività pratica.

<Pre-requisiti - conoscenze di base>

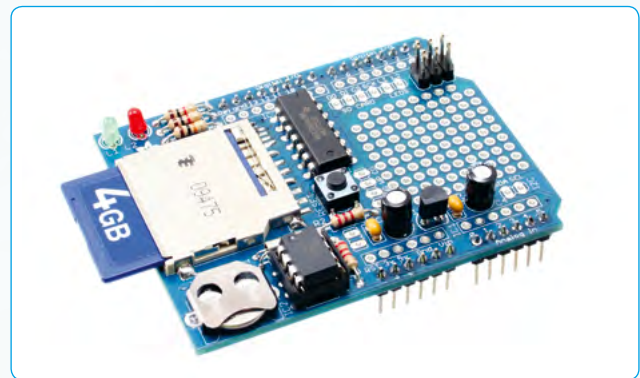
Secondo il curriculum didattico valido in Grecia:

- ↳ conoscenze di base della programmazione che sono state acquisite durante il 3° anno di scuola media e il 1° anno di scuola media superiore
- ↳ conoscenze di base di acidità e teoria del pH acquisite durante il terzo anno di scuola media superiore

<Materiali didattici/spazio utilizzati>

Il laboratorio di Robotica Educativa e STEM utilizzato dagli autori disponeva di:

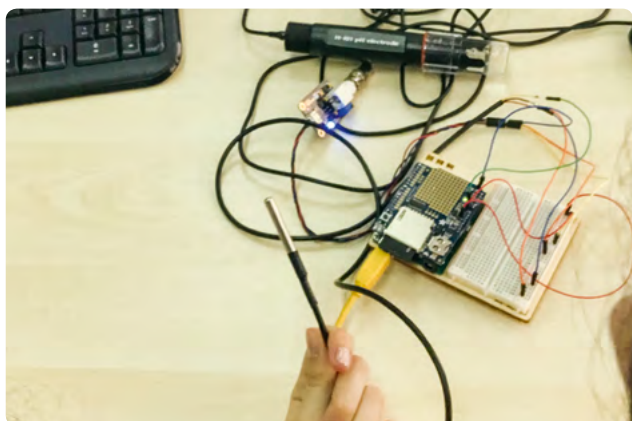
- ↳ Arduino^[1] starter kit (include una scheda Arduino, cavi, schermo LCD, ecc.)
- ↳ Adafruit Data Logger Shield per Arduino (📷1)
- ↳ Misuratore di pH analogico Pro Kit Pro per Arduino (📷2)
- ↳ Sensore di temperatura impermeabile (📷3)
- ↳ Scheda SD
- ↳ Un computer con porta SD, come ad esempio un computer portatile, è necessario per la programmazione e per la registrazione dei dati di questa unità didattica.
- ↳ Acqua demineralizzata (acqua purificata che ha subito l'eliminazione della maggior parte o di tutti gli ioni minerali e salini)
- ↳ Sacchetti con gel refrigerante e un congelatore per conservare i cubetti di ghiaccio



📷 1: Adafruit Data Logger Shield per Arduino^[3]



📷 2: Misuratore analogico di pH per Arduino



© 3: Sensore di temperatura impermeabile

<Quesito di partenza per la ricerca>

C'è una relazione tra l'acidità di un liquido e la sua temperatura?

<Indagini e domande per la risoluzione dei problemi>

1. Come possiamo collegare i sensori alla scheda Arduino?
2. Come faremo la registrazione dei dati?

<Cosa fanno gli studenti/insegnanti>

<Fase preparatoria: Introduzione - teoria - lavoro di gruppo>

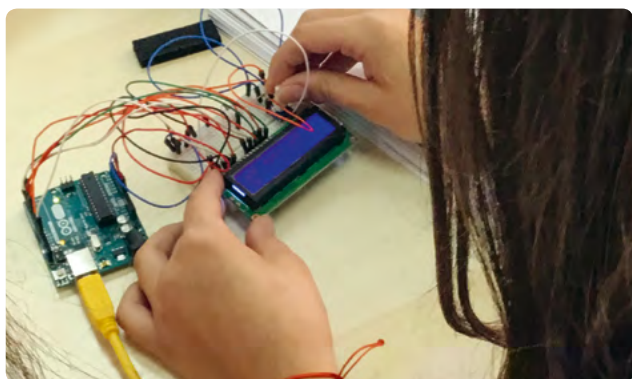
Durata: 1 ora

Gli studenti vengono divisi in gruppi di lavoro e seguono una breve spiegazione sul microcontrollore Arduino^[4], sui sensori utilizzati (di pH e di temperatura) e sul loro funzionamento. Viene illustrata anche la teoria dell'acidità, del pH-metro e la relazione tra la reazione di una soluzione e la sua temperatura. Agli studenti verrà poi chiesto di progettare un esperimento per misurare l'influenza della temperatura sulla reazione acido/base di un liquido.

<Fase 1: Introduzione ad Arduino e alla programmazione>

Durata: 1 ora

Gli student apprendono le basi di programmazione di Arduino e osservano i circuiti, collegano lo schermo LCD, e imparano a visualizzare un semplice messaggio. (©4 & 5).

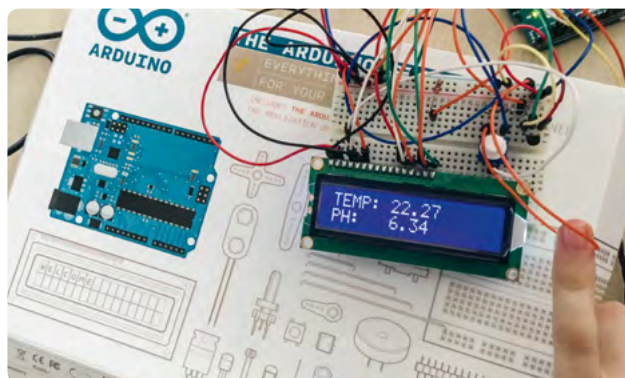


© 4: Collegamento dello schermo LCD

<Fase 2: Collegamento dei sensori>

Durata:1 ora

Gli studenti imparano il funzionamento del sensore di pH (©6) e del sensore di temperatura (©3) collegandoli all'Arduino^[4] e lo programmano per visualizzare i dati sullo schermo LCD.



© 5: Visualizzazione delle misure del sensore

<Fase 3: Il Datalogger shield (interfaccia dati) per Arduino>

Durata: 2 ore

Gli studenti saldano lo shield di datalogging sulla scheda Arduino^[4] dotata di scheda SD necessaria per la registrazione dei dati (©7). Gli studenti programmano il datalogger, che ha il proprio orologio in tempo reale (RTC). Quindi iniziano l'esperimento utilizzando acqua demineralizzata a 25°C e ne misureranno pH e temperatura immergendo i sensori nel liquido per 10 secondi.



© 6: Un sensore di pH

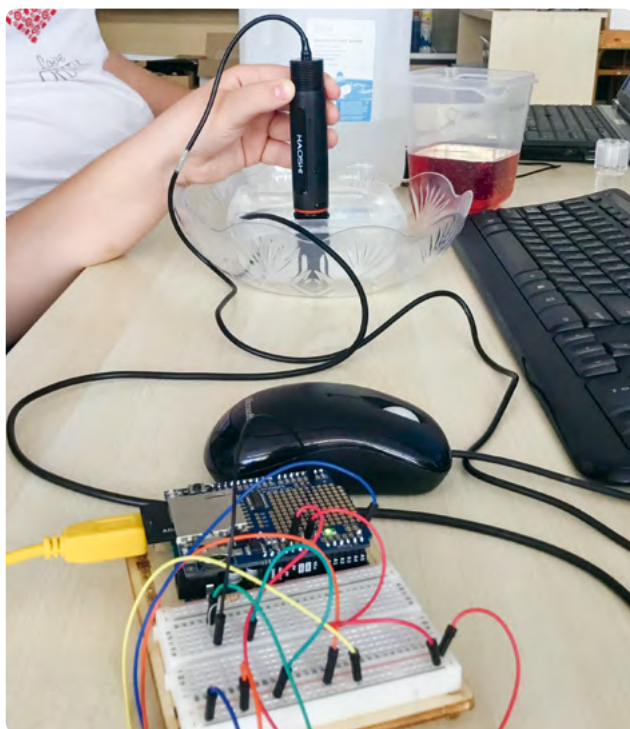


© 7: Data Logger Shield saldato su Arduino^[4]

<Fase 4: L'esperimento>

Durata: 1 ora

Gli studenti svolgono i test con campioni di acqua demineralizzata, che hanno tutti una temperatura diversa. Si inizia con acqua demineralizzata (a temperatura ambiente) in una ciotola, o beaker, immersa in un bagno di acqua e cubetti di ghiaccio attorno ad essa [8 & 9]. In questo modo, il bagno d'acqua raffredda il liquido di misura in modo delicato e graduale. Ogni minuto, i sensori vengono immersi nel liquido per 10 secondi e si raccolgono i dati. La procedura viene ripetuta almeno 6 volte per raccogliere una grande quantità di dati necessari per la fase di plottaggio dei valori.



8: Misura del pH e della temperatura



9: Bagno d'acqua con cubetti di ghiaccio

<Fase 5: Risultati>

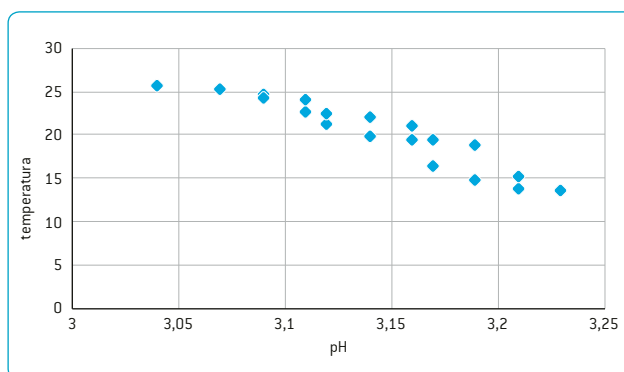
Durata: 1 ora

Gli studenti scollegano la scheda SD dal data logger, la inseriscono nel computer portatile per leggere il file di dati, li importano in un foglio di calcolo (es. MS-Excel) e plottano i dati su un grafico, che rappresenta la possibile relazione tra temperatura e pH. I dati sono salvati sulla scheda SD stessa in formato .csv. L'insegnante e gli studenti discutono leggendo e interpretando i dati e formulano una possibile risposta alla domanda della ricerca. Gli studenti presentano i loro risultati alla classe e li discutono con i loro coetanei.

<Conclusione>

Al termine dell'attività gli studenti hanno fatto pratica di interdisciplinarietà STEM, mediante programmazione e informatica.

L'esperimento potrebbe essere esteso utilizzando una varietà più ampia di liquidi, come ad esempio l'aceto a differenti temperature (vedi 10).



10: Dati dell'aceto

Per lo svolgimento di questa attività è necessario che la saldatura tra lo shield di datalogging e la scheda Arduino venga realizzata con grande precisione; alcuni studenti potrebbero avere difficoltà a eseguirla correttamente, in tal caso si consiglia all'insegnante di svolgerla personalmente.

<Riferimenti>

- [1] www.arduino.cc
- [2] www.arduino.cc/en/Main/Software
- [3] Immagine: oomlout (https://commons.wikimedia.org/wiki/File:ARSH-09-DL_03.jpg), "ARSH-09-DL 03", CC BY-SA 2.0, <https://creativecommons.org/licenses/by-sa/2.0/legalcode>
- [4] Immagine: oomlout ([https://commons.wikimedia.org/wiki/File:ARSH-09-DL_\[5703636953\].jpg](https://commons.wikimedia.org/wiki/File:ARSH-09-DL_[5703636953].jpg)), "ARSH-09-DL [5703636953]", CC BY-SA 2.0, <https://creativecommons.org/licenses/by-sa/2.0/legalcode>

Capitan telecomando

<Autore> Immaculada Abad Nebot

<Autore> Pere Compte Jové



<Info>

<Parole chiave> telecontrollo, progettazione 2D e 3D, modellazione, saldatura schede elettroniche, programmazione chip, programmazione app, stampante 3D

<Discipline> tecnologia, ingegneria

<Età degli studenti> 14–16

<Hardware> Arduino^[1], modulo Bluetooth, materiali per la costruzione del modello di barca

<Linguaggi> Arduino, ArduinoBlocks^[2], AppInventor^[3]

<Livello di difficoltà di programmazione> medio

<Sommario>

Gli studenti progettano e costruiscono la propria barca che navigherà in una piscina. Una volta completata questa sfida iniziale, utilizzeranno una scheda Arduino per controllare l'imbarcazione a distanza con un tablet o uno smartphone.

<Introduzione concettuale>

Gli studenti progettano il proprio modello e imparano a considerare questo compito da un punto di vista ingegneristico. Il progetto inizia visionando e analizzando su internet diversi tipi di imbarcazioni. Successivamente, un piccolo gruppo di studenti costruisce un modello di imbarcazione stabile sull'acqua.^[4]

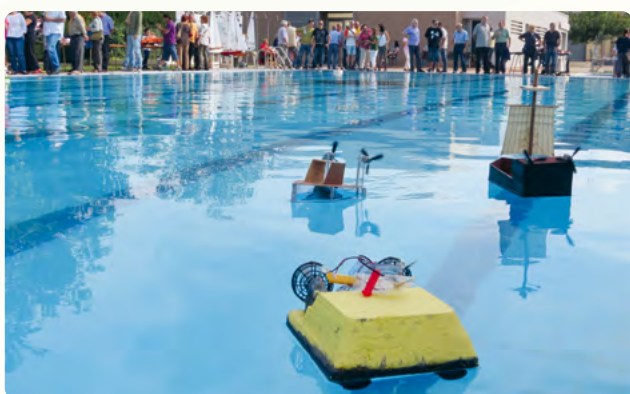
Il funzionamento è basato su un interruttore fatto dagli studenti e che permette all'utente di controllare due motori (ognuno dei quali può girare avanti e indietro).

Gli studenti installano anche un modulo Bluetooth in modo da poter controllare l'imbarcazione a distanza tramite uno smartphone. Utilizzando AppInventor^[3], realizzano un'app con diversi sistemi di controllo, ad esempio con pulsanti, con controllo vocale o con un accelerometro (l'imbarcazione potrà cambiare direzione a seconda della posizione della mano della persona che la controlla).

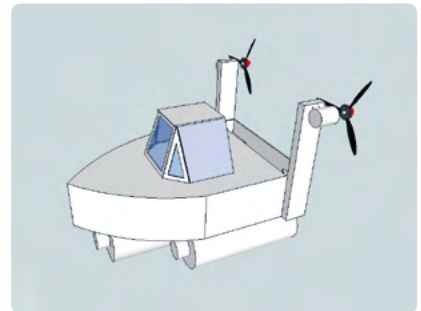
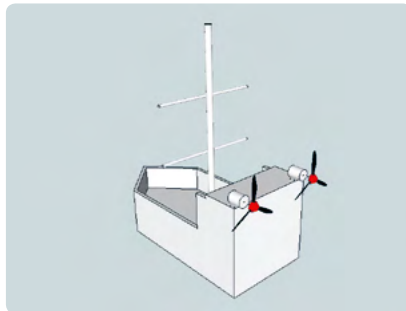
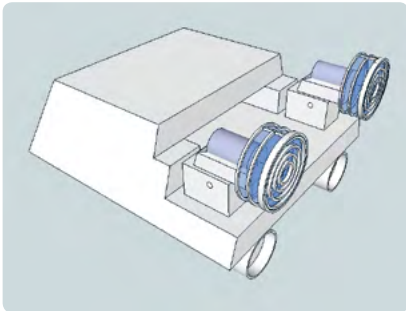
Infine, gli studenti possono presentare i loro progetti ad una mostra di modellismo navale (📍 1) davanti agli esperti, e discutere con loro di eventuali difetti e implementazione dei loro modelli. Questo permette agli studenti di analizzare criticamente il lavoro e migliorare i loro modelli con la realizzazione delle versioni successive.

<Cosa fanno gli studenti/insegnanti>

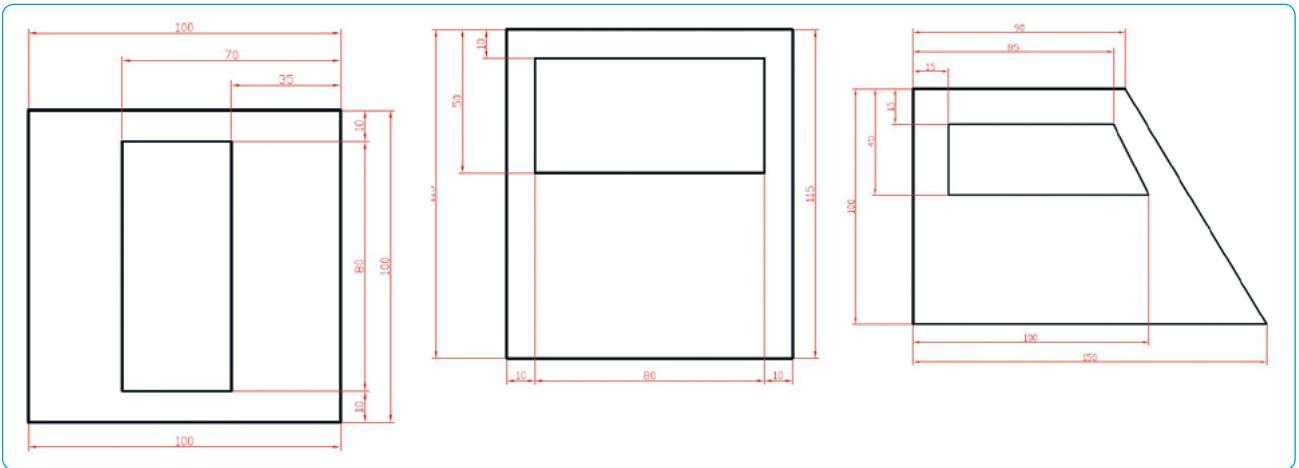
Gli studenti progettano il design preferito della loro imbarcazione dopo aver visionato diversi modelli su Internet. Possono disegnare l'imbarcazione con un software di progettazione 3D come sketchUp^[5] o Tinkercad^[6]. Tuttavia, devono tenere presente che le barche devono essere molto stabili sull'acqua per evitare che si ribaltino. (📍 2)



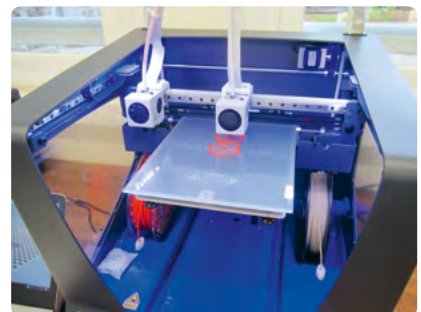
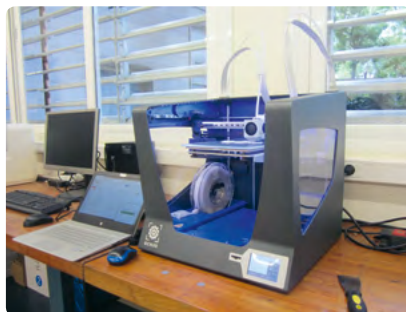
📍 1: Presentazione dei modelli di imbarcazioni in occasione di una mostra di modellismo navale in Spagna



© 2: Vari modelli 3D di imbarcazioni



© 3: Porta, parte anteriore e laterale della cabina dell'imbarcazione



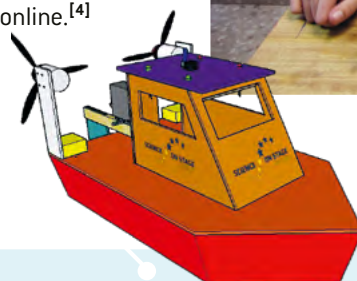
© 4a-c: Stampanti presso il centro stampa 3D Cesire Aulatec, Barcellona

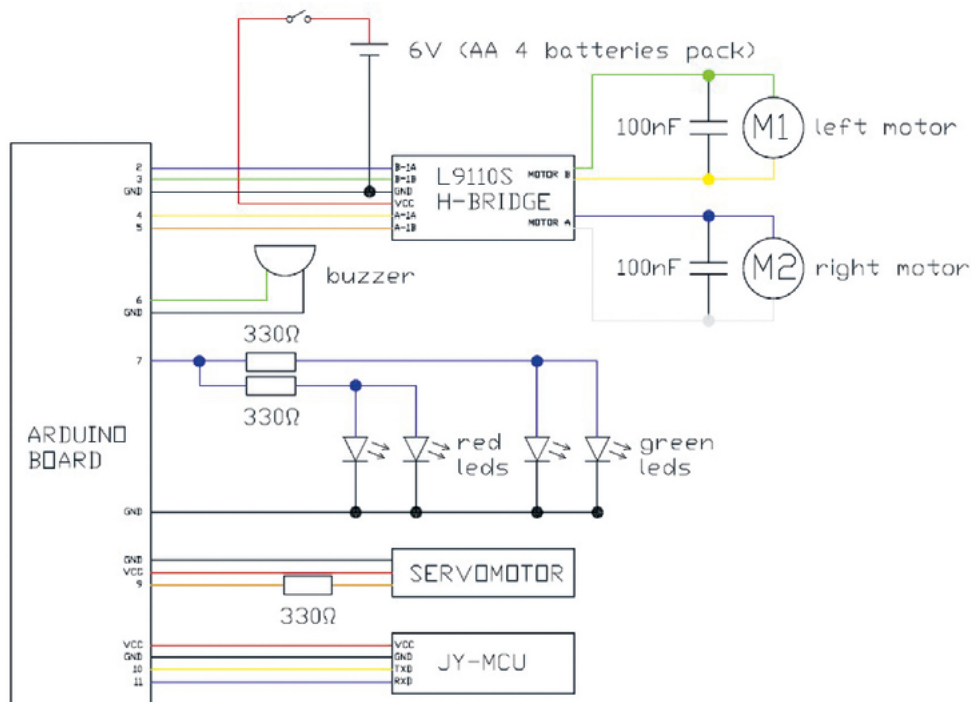
Sono disponibili online le planimetrie per un modello di barca (©3), complete di tutte le misure. Se volete utilizzare questo disegno o stamparlo, potete scaricarlo in vari formati (.skp, .stl e .gcode) al riferimento fornito nella sezione "Riferimenti".^[4]

Se la vostra scuola non dispone di una stampante 3D, spesso è possibile collaborare con altre istituzioni come università, servizi forniti da laboratori di maker o altri soggetti. Nel nostro caso, gli studenti hanno eseguito la stampa 3D presso un centro stampa 3D (©4a-c). Le istruzioni passo-passo per la costruzione dell'imbarcazione sono disponibili online.^[4]



© 5, 6: Modello finale di barca



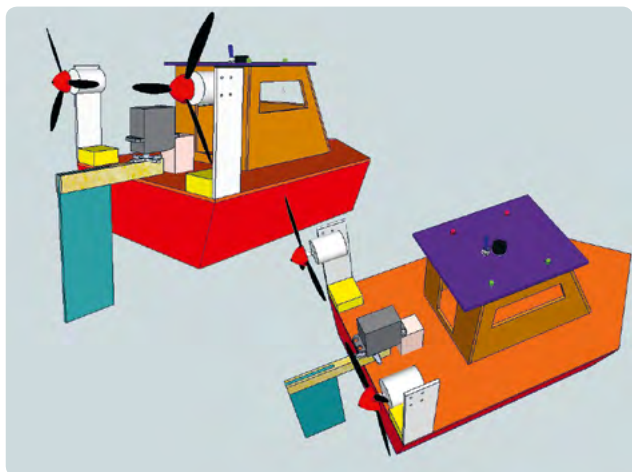


Ⓒ 7: Schema del circuito elettronico

<Circuito elettronico>

Per collegare gli accessori alla scheda Arduino^[7], utilizzare lo schema di Ⓒ7 e seguire le relative istruzioni:

1. È possibile installare un cicalino (buzzer) (Arduino pin 6) e luci (Arduino pin 7) sul tetto. (Ⓒ8)
2. Nella parte posteriore della barca è possibile collegare un servomotore per controllare un timone (Arduino pin 9). (Ⓒ8)
3. Collegare il modulo Bluetooth secondo lo schema TXD (Arduino pin 10) e RXD (Arduino pin 11).
4. Collegare una scheda di controllo del driver motore L9110S per Arduino con batterie esterne. (Ⓒ7)



Ⓒ 8: Barca con dispositivi accessori



Ⓒ 9: Costruzione delle imbarcazioni

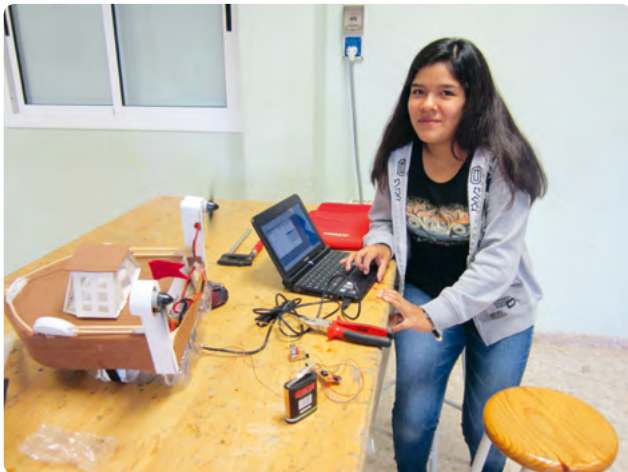
<Come controllare i motori e le altre funzionalità dell'imbarcazione>

Si consiglia di programmare Arduino con Arduino IDE^[1], ArduinoBlocks^[2] o un altro programma simile.

Gli studenti possono programmare le seguenti funzionalità:

1. Accendere e spegnere le luci sul tetto.
2. Emettere un suono con il buzzer.
3. Controllare la posizione del timone tramite il servo-motore (40° a destra, 20° a destra, al centro, 20° a sinistra e 40° a sinistra).
4. Entrambi i motori devono girare e la barca si muove in avanti.

5. Entrambi i motori devono girare e la barca si muove all'indietro.
6. L'imbarcazione deve cambiare direzione verso destra (il motore sinistro gira in avanti e quello destro all'indietro).
7. L'imbarcazione deve cambiare direzione verso sinistra (il motore destro gira in avanti e quello sinistro all'indietro).
8. Controllare tutti i programmi via Bluetooth.



© 10: Programmare l'imbarcazione



© 11: L'interfaccia utente dell'app

<Programmare l'applicazione per controllare l'imbarcazione con uno smartphone utilizzando AppInventor^[3]>

Si possono progettare le seguenti funzionalità:

1. Programmare l'applicazione in modo che utilizzi il Bluetooth per connettersi con l'imbarcazione.
2. Controllare i diversi elementi della barca tramite i pulsanti.
3. Controllare la pala del timone con una barra di scorrimento.
4. Controllare l'imbarcazione usando l'accelerometro del tablet o dello smartphone; inclinando lo smartphone in avanti, indietro, a destra o a sinistra, l'imbarcazione si muoverà nella direzione corrispondente.
5. Utilizzare l'opzione di riconoscimento vocale per controllare l'imbarcazione con la voce.
6. Combinazioni varie di tutti questi programmi.

<Elenco dei materiali e delle attrezzature necessarie>

È possibile trovare online, mediante i riferimenti forniti, una lista contenente tutti i materiali necessari^[4]. L'elenco include dettagli quali la quantità necessaria, la fascia di prezzo e dove si possono reperire.

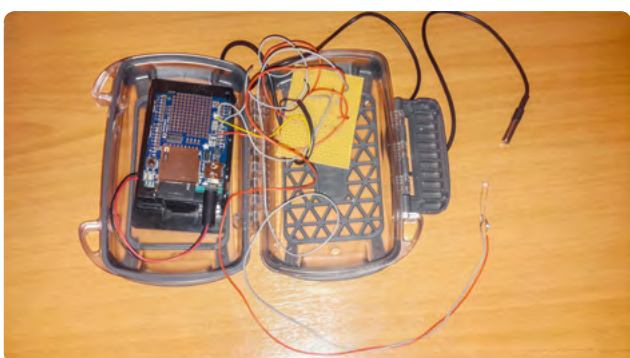
Il materiale per la costruzione della barca costa circa 21 €, per i componenti elettronici circa 15 €, e 6 € per il resto del materiale necessario.

<Attività cooperativa>

Durante la realizzazione di questa attività, abbiamo collaborato con Eleftheria Karagiorgou e Sevasti Tsiliki del 7° Liceo Superiore di Trikala, Grecia, per implementare un circuito Arduino in un modello di hydrobot sottomarino che ci permette di navigare sott'acqua. In questo caso, è molto importante proteggere i motori con cera e utilizzare una custodia protettiva e stagna per la piastra Arduino per evitare che l'acqua entri nei motori. Abbiamo dotato il nostro hydrobot, denominato 'Argolith'^[8], del microcontrollore Arduino UNO per fornirgli di un 'cervello' elettronico e per registrare le misure di luminosità e temperatura sott'acqua. Il "cervello" comprendeva anche un modulo per la registrazione dei dati dotato di orologio e di un circuito di registrazione dati per scheda SD, su cui abbiamo salvato i dati misurati.



© 12: Costruzione dell'hydrobot sottomarino



© 13: Arduino e il contenitore a tenuta stagna

Il materiale online contiene un video subacqueo dell'hydrobot "Argolith" durante i test svoltisi in un fiume a Trikala, in Grecia.^[4]

<Riferimenti>

- [1] www.arduino.cc
- [2] www.arduinoblocks.com
- [3] <http://appinventor.mit.edu>
- [4] Tutte le fasi di questo progetto e ulteriori informazioni:
www.science-on-stage.de/coding-materials.
- [5] www.sketchup.com
- [6] www.tinkercad.com
- [7] www.arduino.cc/en/Reference/Board
- [8] Un manuale di costruzione è disponibile all'indirizzo
<http://seaperch.mit.edu/build.php>.

Come programmare

<Autore> Bernard Schriek

Questo articolo tratta l'uso della programmazione (coding). Potreste aver bisogno di hardware e software particolari per realizzare le unità didattiche di questo opuscolo; è inoltre utile avere una conoscenza di base dei fondamenti di programmazione. Questo capitolo ha lo scopo di fornire una chiara visione d'insieme di questo tipo di informazioni.

<Hardware>

La programmazione in ambito scientifico riguarda principalmente la misurazione di grandezze fisiche e chimiche con sensori e il controllo di determinate uscite. I sensori, tipicamente, misurano parametri fisici come la temperatura, il rumore, la luce, la distanza, il pH, la pressione esercitata su pulsanti, il tatto, ecc. Le uscite generalmente possono essere un LED, un cicalino (buzzer), un altoparlante, un motore, ecc.

<Arduino>

Questi sensori e uscite possono essere collegati o sono già integrati in una scheda microcontrollore o in un microcomputer a scheda singola. Questi possono essere di tipologie più o meno sofisticate. La scheda più semplice disponibile è un Arduino^[1]. E' disponibile in diverse versioni, quella denominata Arduino-UNO è ampiamente utilizzato nelle scuole (e anche in questa pubblicazione). La scheda contiene un microcontrollore a 8 bit che funziona abbastanza lentamente. Tuttavia, la velocità del processore non è importante per la maggior parte delle applicazioni in quanto la scheda comunica esclusivamente tramite un pulsante di reset e un LED.

La scheda può essere cablata a vari altri sensori e uscite. I programmi per un Arduino sono scritti su un computer e inviati ad un Arduino tramite una connessione USB. Quando il programma viene caricato su Arduino, può essere avviato (e successivamente interrotto e riavviato) premendo il pulsante di reset. Quando l'Arduino è collegato via USB, è alimentato attraverso la connessione USB, tuttavia richiede una fonte di alimentazione separata quando è scollegato dal computer.

Un modo molto pratico per espandere una scheda Arduino è attraverso l'uso di shields, che sono schede di espansione del

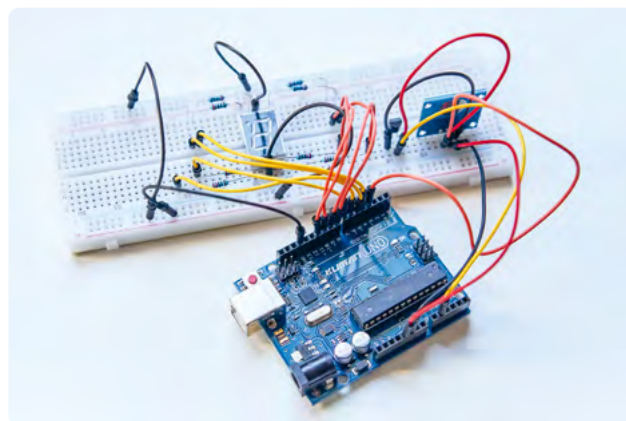
circuito che si collegano direttamente ai pin di Arduino. Ad esempio, spesso vengono utilizzati moduli di interfacciamento di data logger muniti di un orologio in tempo reale. I dati vengono scritti su una scheda SD che viene inserita nello slot per schede SD del microcontrollore Arduino.

Su Internet si possono trovare esempi di programmazione ben documentati per quasi tutte le espansioni disponibili (interfacce, sensori e uscite).

I computer Windows, MacOS o Linux possono essere utilizzati per programmare Arduino. Il software può essere scaricato online.^[2] In rete troverete anche numerosi esempi approfonditi che spiegano in dettaglio come programmare l'Arduino. I programmi per Arduino sono chiamati "sketches".

<Calliope mini e BBC micro:bit>

Un altro tipo di computer a scheda singola è il Calliope-mini^[3], che è a sua volta compatibile con un altro tipo di microcontrollore, il BBC micro:bit^[4]. La differenza tra queste due schede è che Calliope mini dispone di molti sensori e uscite già integrate, quindi per molti progetti non è necessario aggiungerli. Il Calliope-mini dispone già anche di Bluetooth per comunicare con altre schede oppure con smartphone. Il processore è più forte e più veloce del processore dell'Arduino, ed ha anche una memoria notevolmente superiore. I programmi scritti su computer possono essere trasferiti via USB su Calliope mini. Il pro-

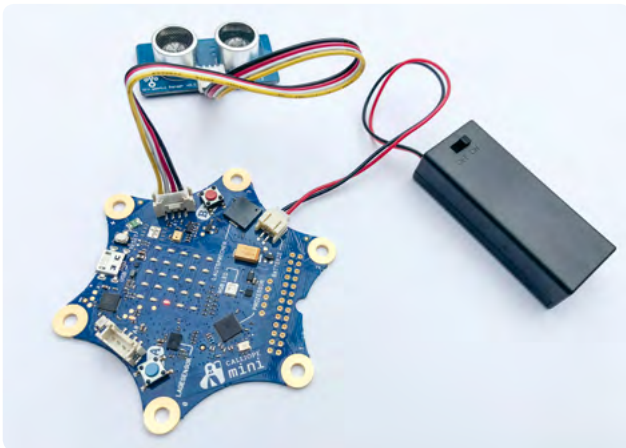


© 1: Un Arduino

gramma si avvia automaticamente al momento del trasferimento, ma può anche essere riavviato premendo il pulsante di reset. Calliope-mini oltre a disporre già di molti sensori, può essere collegato ad altri tramite connettori grove standardizzati^[5].

Dispone di una matrice a LED 5×5 integrata che può essere utilizzata per lo scorrimento del testo. Il Calliope-mini è leggermente più costoso dell'Arduino, ma ha l'innegabile vantaggio di avere molti sensori ed uscite, per tale ragione è molto adatto per lo svolgimento di attività didattiche scolastiche.

Calliope-mini utilizza JavaScript per la programmazione, ma è anche possibile utilizzare ambienti di programmazione a blocchi più adatti per i bambini più piccoli e che verranno descritti in dettaglio più avanti.



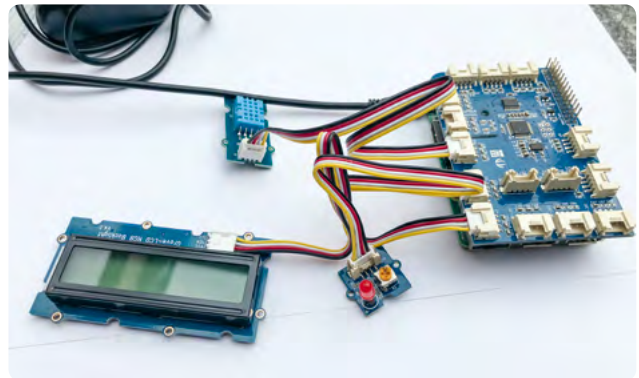
© 2: Calliope mini con un sensore di distanza ad ultrasuoni e le batterie

<Raspberry Pi>

Il Raspberry Pi^[6] è un computer a scheda singola, molto comune e che funziona sia in ambiente Linux che Windows. Può essere collegato sia ad uno schermo tramite un cavo HDMI, che a mouse e tastiera via USB. Il suo prezzo è molto ragionevole (tra i 50 e i 60 euro), ma comunque doppio rispetto al Calliope-mini. Sul Raspberry Pi è possibile utilizzare quasi tutti i linguaggi di programmazione. Utilizza una scheda SD per salvare programmi e dati. Può essere espanso con diverse schede di espansione idonee per gli scopi più diversi. Dispone di un modulo Wi-Fi integrato e può essere collegato ad una rete Wi-Fi locale. Come tutte le altre schede, necessita di un'alimentazione esterna tramite batterie o adattatore-alimentatore. Il Raspberry Pi è, di fatto, un computer vero e proprio, anche se, ovviamente, meno veloce di un desktop o di un notebook e dispone di una sola scheda SD come memoria esterna. In rete è possibile trovare migliaia di progetti didattici realizzati con questo strumento che, comunque, richiede un po' di esperienza di programmazione con i computer per poter essere usato proficuamente.



© 3: Un Raspberry Pi



© 4: Raspberry Pi con scheda di espansione

<LEGO Mindstorms>

LEGO Mindstorms^[7] è un sistema microcomputer molto più costoso dei precedenti. La maggior parte degli studenti ha già una certa esperienza con LEGO, quindi con loro si possono costruire facilmente strumenti e macchine a controllo numerico. LEGO dispone di un proprio sistema di programmazione basato su icone chiamato EV3 software, basato su LabView^[8] (un software professionale per la misurazione e il controllo). Qui i blocchi di programmazione vengono spostati per realizzare un programma di lavoro. L'uso e la programmazione dei motori è molto importante quando si programma un EV3-Robot. Oltre al software originale LEGO, è possibile utilizzare anche il leJOS^[9], un'implementazione speciale della Java virtual machine. Inoltre molte persone usano Eclipse, un plug-in LEGO, come strumento di sviluppo Java su computer. Il processore principale può essere collegato ad un gran numero di sensori e di unità di uscita aggiuntive, che però, come tutti i materiali LEGO, sono più costosi di quelli precedentemente indicati.

Ci sono centinaia di altri computer a scheda singola, molti dei quali progettati per gestire robot per ogni scopo. Se siete alla ricerca di altre idee per i vostri progetti, vi consigliamo di visitare il sito hackster.io^[10]; questo sito richiede la registrazione [gratuita] per accedere alle descrizioni dettagliate dei progetti.

<Software>

Nel XXI secolo le competenze e le abilità di programmazione diventeranno sempre più importanti in tutti i settori della vita. Uno degli obiettivi di questo opuscolo è quello di aiutare gli insegnanti a stimolare interesse e motivazione dei loro studenti per la programmazione. Gli studenti sono, solitamente, già molto interessati a questi argomenti ed hanno, spesso, solo bisogno degli strumenti e dei linguaggi di programmazione adeguati per realizzare interi progetti e, di conseguenza, essere più motivati. La scelta del linguaggio di programmazione dipende dall'età degli studenti; ad esempio, gli studenti più giovani avranno bisogno di maggiore aiuto visivo durante il processo di programmazione e di debug. Pertanto, i prossimi paragrafi illustreranno come implementare diversi importanti concetti della programmazione, sia "a blocchi" che "a testo".

<Variabili>

Le variabili sono usate per memorizzare i valori per un uso successivo. Un buon modo di guardare alle variabili è quello di immaginarle come delle scatole. Un determinato box contiene un certo valore che ha una forma che dipende dal tipo di valore (intero, float, stringa, booleano, cioè vero o falso); inoltre è applicata un'etichetta con il nome della variabile. Vi suggeriamo di usare nomi esplicativi per le variabili, come ad esempio "temperatura" invece della lettera "t" in modo che altre persone possano comprendere meglio i vostri programmi. Nei linguaggi a blocchi (Scratch^[11], Snap!^[12], MakeCode^[13]), una variabile ha l'aspetto di un'etichetta e possiamo vedere il nome e il valore anche mentre un programma è in esecuzione (📍5). Questa caratteristica è molto utile per il debug. Esistono anche blocchi per impostare o modificare il valore di una variabile che sono autoesplicativi.

temperature 23

📍 5

<Assegnazioni>

In alcuni linguaggi di programmazione testuale, le variabili devono essere dichiarate. La dichiarazione contiene informazioni sul tipo di variabile (intero, stringa, stringa,). Altri linguaggi di impostazione testuale attendono la prima assegnazione per decidere implicitamente il tipo di variabile.

L'assegnazione ha un ostacolo: è scritta come un'equazione matematica. "temperatura = 23", ad esempio, è un'assegnazione e significa che alla variabile temperatura è dato il valore 23 (📍6). Se si vuole confrontare la temperatura con il valore 25, si scrive "temperatura == 25", usando due segni uguali. Questa differenza è la causa di molti errori nello sviluppo del software.

```
show variable temperature
set temperature to 23
```

📍 6

<Flusso di programma>

La maggior parte dei programmi per computer è formata da dichiarazioni che vengono elaborate consecutivamente, cioè una dopo l'altra. Certo, molti linguaggi includono l'elaborazione parallela, dove due o più programmi vengono eseguiti contemporaneamente in sequenza – anche Scratch^[11] o Snap!^[12] offrono questa elaborazione parallela.

Ma ora esaminiamo un singolo processo. Le istruzioni vengono eseguite dal primo blocco, o linea, fino all'ultimo blocco, o



linea. Questa si chiama sequenza. Di solito però le cose non sono così semplici e si desidera che il programma esegua comandi diversi in base alle decisioni prese in determinati punti del programma. Questo si chiama branching (ramificazione).

<Branching (diramazione)>

Ci sono tre tipi di “branching” o “selezione”: a una uscita (one-sided), a due uscite (two-sided) e a più uscite (multi-sided), e tutti hanno bisogno di una condizione per indicare ciò che si vuole che il programma faccia. Questa condizione è di solito un confronto tra dei valori. Il risultato è di tipo booleano, vero / falso. La selezione ad una uscita aggiunge solo diverse istruzioni aggiuntive al flusso del programma, che vengono eseguite quando la condizione risulta vera. La selezione a due uscite aggiunge due ulteriori serie di istruzioni di cui però viene eseguito un solo set, a seconda del risultato della condizione. La selezione a più uscite, infine, rende eseguibili diversi set di istruzione a seconda del valore di una variabile. La programmazione con linguaggi a blocchi fornisce una buona visualizzazione sia delle condizioni che del set di istruzioni. Mentre la programmazione con linguaggio testuale utilizza istruzioni “if” (se) oppure “if-else” (se-allora) per i primi due tipi di selezione, usa l’istruzione a switch o multipartizione “case” per la selezione a più uscite.



📷 7



📷 8

<Cicli>

I loop (cicli), che vengono utilizzati per set di istruzioni eseguite ripetutamente, sono un altro strumento importante usato per controllare il flusso delle istruzioni. I loop possono essere distinti dalla condizione che controlla il loop. La condizione – di solito un confronto – può essere all’inizio o alla fine del ciclo. Se la condizione è all’inizio e risulta falsa, le istruzioni del ciclo non saranno mai eseguite. Se la condizione è alla fine del ciclo, le istruzioni del ciclo saranno eseguite almeno una volta. I cicli di conteggio, a volte chiamati cicli for (for-loops), sono usati quando è noto quante volte le istruzioni del ciclo devono essere eseguite.



📷 9



📷 10

Le sequenze di istruzioni, i loop e le diramazioni sono normalmente annidati. Ciò significa che un ciclo può essere all’interno di una selezione, e una selezione “branching” può essere all’interno di un ciclo. Ci sono regole su come documentare il flusso del programma in un diagramma. In questo opuscolo, usiamo i diagrammi di Nassi-Shneiderman^[14] per spiegare il flusso di istruzioni nei nostri programmi.

<Riferimenti>

- [1] www.arduino.cc
- [2] www.arduino.cc/en/Main/Software
- [3] <https://calliope.cc/en>
- [4] www.microbit.co.uk/home
- [5] http://wiki.seeedstudio.com/Grove_System/
- [6] www.raspberrypi.org
- [7] www.lego.com/en-us/mindstorms
- [8] <https://en.wikipedia.org/wiki/LabVIEW>
- [9] www.lejos.org
- [10] www.hackster.io
- [11] <https://scratch.mit.edu>
- [12] <https://snap.berkeley.edu>
- [13] <https://makecode.calliope.cc/?lang=en>
- [14] https://en.wikipedia.org/wiki/Nassi-Shneiderman_diagram

<Imparate la programmazione con Snap!>

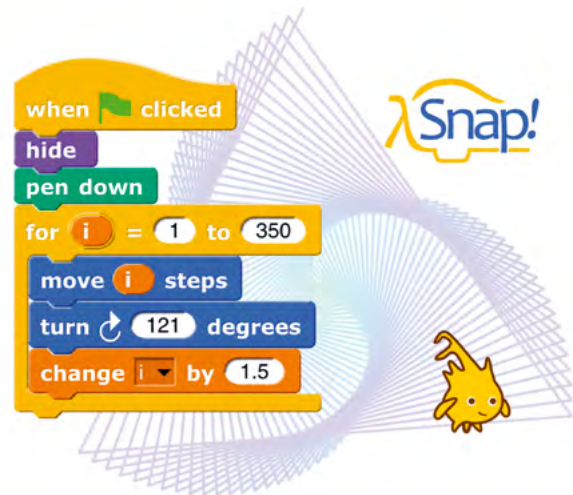
snap.berkeley.edu/run

La creatività e l'alfabetizzazione informatica e multimediale sono considerate competenze importanti nella rivoluzione digitale in corso. Snap! è uno strumento che supporta le persone di qualsiasi età e background per aiutarle ad entrare in contatto con l'informatica (Computer Science, CS) e la programmazione e partecipare agli sviluppi di questa disciplina.

<Cos'è Snap!?>

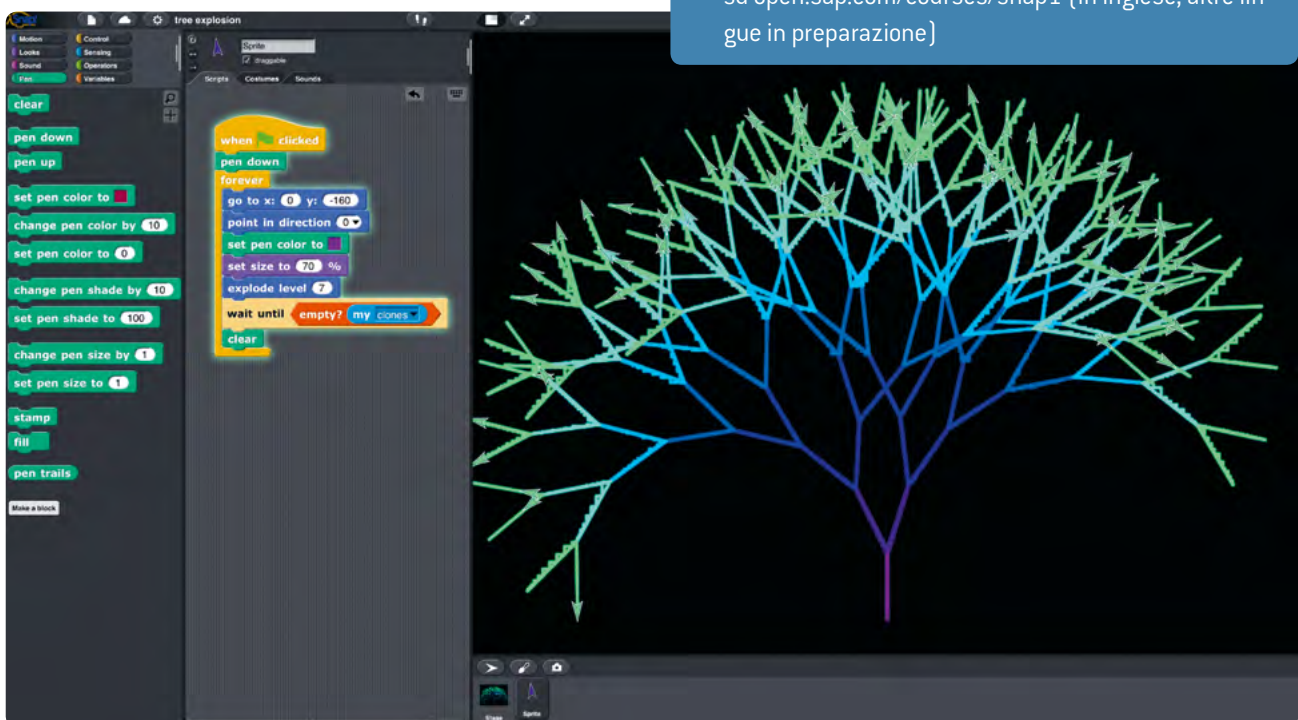
Snap!—Build your own Blocks è un linguaggio di programmazione visuale, basato su blocchi. Invita gli studenti a dare vita alle loro idee e nel contempo a conoscere l'informatica in un modo divertente e sperimentale. L'elemento affascinante di *Snap!* è la sua forte aspirazione a consentire un basso livello di competenze in ingresso senza ridurre l'espressività dello strumento. Consente a programmatori principianti ed esperti di immergersi in concetti avanzati di CS come strutture dati arbitrarie, funzioni di ordine superiore e persino strutture di controllo personalizzate in un ambiente visivamente attraente e comprensibile.

Snap! è sviluppato dalla SAP insieme ai ricercatori dell'UC Berkeley. Oggi è disponibile in oltre 40 lingue e viene utilizzato per l'educazione alla CS in altrettanti paesi. *Snap!* è open-source e funziona con tutti i moderni browser web.



<Sapevate che Snap!...>

- ↳ è uno dei 100 migliori linguaggi di programmazione nell'indice TIOBE
- ↳ è usato per l'intelligenza artificiale dai ricercatori dell'Università di Oxford
- ↳ viene utilizzato dal produttore di scena per la stampa 3D, il ricamo e la robotica
- ↳ è facile da insegnare, ad esempio con il corso gratuito su open.sap.com/courses/snap1 (in inglese, altre lingue in preparazione)



<Partecipare ai Meet and Code>

Chiedi il supporto di cui hai bisogno per organizzare e promuovere la programmazione nella tua scuola!

“Non avevo idea che programmare fosse così facile!” *Meet and Code 2018* è stata una vera e propria rivelazione per Philip. “Voglio davvero continuare”, disse Louisa dopo aver partecipato ad un hackathon. Ed è esattamente quello che queste iniziative vogliono dimostrare: la programmazione è divertente e facile da imparare.

Nel secondo anno di *Meet and Code*, più di 52.000 ragazzi e ragazze hanno partecipato a più di 1.100 eventi in 22 paesi. Come sempre, l'azione si è svolta durante la settimana europea della programmazione (*EU Code Week*) in ottobre.

Ogni evento verificato e ritenuto meritevole di finanziamento ha ricevuto un finanziamento fino a 500 euro. Tutti i tipi di



© Peter Böhmer

eventi di programmazione possono essere sostenuti e qualsiasi organizzazione senza scopo di lucro può candidarsi con un suo progetto.



© Dietrich Bechte1



© Dietrich Bechte1

L'iniziativa intende continuare anche nei prossimi anni per confermare i successi sin qui conseguiti: numerosi eventi, progetti e Laboratorio si svolgeranno durante la *EU Code Week* in 22 paesi. L'obiettivo è quello di introdurre bambini e giovani tra gli 8 e i 24 anni nel mondo della tecnologia e della programmazione. Gli eventi sono pensati per mostrare ai giovani quanto può essere divertente la programmazione e come può aiutare a dare forma alle proprie idee. Essi esplorano un'ampia gamma di argomenti tecnologici, digitali e di programmazione creativa, e vengono così ispirati a sviluppare le competenze digitali di cui hanno bisogno nel mondo di oggi.

Promotori dell'iniziativa l'organizzazione di Monaco di Baviera Haus des Stiftens gGmbH con il suo portale informatico Stifter-helfen e i rispettivi partner nazionali della rete TechSoup Europe. *Meet and Code* è reso possibile da SAP.

Anche nel 2019 verranno assegnati dei premi per le idee più creative e le realizzazioni più originali. Il *Meet and Code Award* sarà presentato in almeno tre categorie, tra cui innovazione e diversità. Tutte le attività, le informazioni e la registrazione sono disponibili all'indirizzo

www.meet-and-code.org

Seguici e unisciti alla discussione:
 @stifter_helfen @TechSoupEurope
 #meetandcode #codeEU #SAP4Good

<Altro materiale>



Gli autori hanno creato risorse e materiale aggiuntivo per le unità didattiche. Potete trovarli online come download gratuito all'indirizzo www.science-on-stage.de/coding-materials

<Eventi nell'ambito del progetto Coding in STEM Education>

<2 giugno 2016>

Raccolta di idee nell'ambito dell'incontro finale del progetto *Football in Science Teaching* Bruxelles, Belgio

<20-22 ottobre 2017>

Primo Laboratorio nell'ambito della finale della *European STEM League* presso il Museo Tedesco del Calcio, Dortmund, Germania.

<1 marzo 2019>

Presentazione della pubblicazione a Vienna, Austria

<13-15 aprile 2018>

Secondo Laboratorio a Berlino, Germania

<Continuazione negli anni 2019 e 2020>

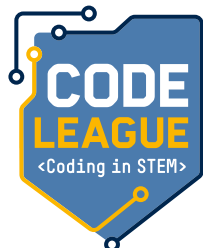
Corsi di formazione per insegnanti in vari paesi europei

<March 2019 - March 2020>

Competizione di *Code League*

<Autunno 2020>

Finale *Code League*



SCIENCE ON STAGE EUROPE

Science on Stage -

La rete europea per gli insegnanti di scienze

... è una rete di insegnanti di scienze, tecnologia, ingegneria e matematica (STEM) di tutti i livelli scolastici.

... offre una piattaforma europea per lo scambio di idee didattiche.

... sottolinea l'importanza della scienza e della tecnologia nelle scuole e tra il pubblico.

Il principale sostenitore di Science on Stage è la Federazione delle associazioni tedesche delle industrie Elettriche e Metallurgiche (GESAMTMETALL) con la sua iniziativa think ING.

Iscriviti alla sezione italiana di Science on Stage

www.science-on-stage.it

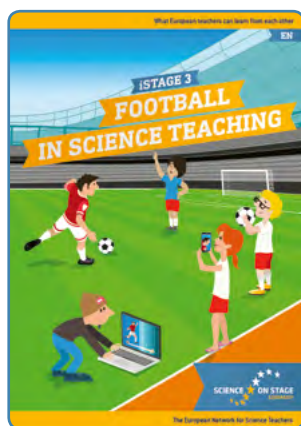
www.facebook.com/scienceonstageeurope

www.twitter.com/ScienceOnStage

Iscriviti alla nostra newsletter

www.science-on-stage.eu/newsletter

<Altro materiale>



Football in Science Teaching

- ↳ Unità didattiche sui vari aspetti STEM nel calcio
- ↳ Capitoli: Biosfera, Corpo, Palla, Palla, Grandi Dati



Smartphones in Science Teaching

- ↳ Linee guida ed esperimenti per l'apprendimento basato sull'indagine con gli smartphone



Lilu's House - Language Skills through Experiments

- ↳ Gli studenti delle scuole elementari scoprono fenomeni scientifici naturali in bagno, soggiorno e cucina mentre praticano la conversazione, la scrittura e la lettura.



Da scaricare gratuitamente al seguente indirizzo
www.science-on-stage.eu/teachingmaterials

Un progetto di



Principale sostenitore di
Science on Stage Germany



Proudly supported by



www.science-on-stage.eu