STEAM - un nuovo framework didattico per l'Alternanza Scuola Lavoro: Coding, Robotica e Design nel Milano Luiss Hub

Roberto Costantini[1], Luigi Laura[1,3], Luigi Mazza[2], and Riccardo Santilli[2]

[1] LUISS
E-mail: {rcostantini,llaura}@luiss.it
[2] ItaliaCamp S.r.l.
E-mail: {l.mazza,r.santilli}@italiacamp.com.
[3] Italian Association for Informatics and Automatic Calculus (AICA)

Sommario In questo articolo descriviamo, in maniera che potremmo definire *open source*, un progetto di Alternanza Scuola Lavoro basato sul framework STEAM (Science, Technology, Engineering, Arts e Mathematics) nel Milano Luiss Hub. Lo STEAM è un framework educativo che si propone di sviluppare competenze attraverso l'integrazione di cinque aree disciplinari, superando la separazione tra discipline tecnico-scientifiche e discipline umanistiche.
Il progetto prevede una attività formativa, fortemente basata su esperienze pratiche, che coinvolge gli studenti in un percorso didattico che integra Coding, mediante il linguaggio visuale Scratch del MIT, Robotica, usando il kit LEGO Mindstorms, e Design, con Tinkercad e Stampa 3D. Le attività del percorso didattico sono descritte in maniera dettagliata, per consentire ad enti e strutture interessate di poterlo replicare in maniera integrale o in parte.

1 Introduzione

La legge 107 del 2015 [7] ha introdotto l'Alternanza Scuola Lavoro (ASL), rendendo obbligatorie anche negli indirizzi non professionalizzanti attività didattiche finalizzati al fornire agli studenti competenze spendibili nel mondo del lavoro. Tra le finalità dell'ASL citiamo, dalla guida pubblicata dal MIUR [4], le seguenti:

1. *attuare modalità di apprendimento flessibili e equivalenti sotto il profilo culturale ed educativo, rispetto agli esiti dei percorsi del secondo ciclo, che colleghino sistematicamente la formazione in aula con l'esperienza pratica;*
2. *arricchire la formazione acquisita nei percorsi scolastici e formativi con l'acquisizione di competenze spendibili anche nel mercato del lavoro;*
3. *favorire l'orientamento dei giovani per valorizzarne le vocazioni personali, gli interessi e gli stili di apprendimento individuali;*

In questo articolo presentiamo un percorso didattico che integra, in maniera fortemente interconnessa, Coding, Robotica e Design: in particolare, al termine

di questo percorso gli studenti dovranno completare un progetto nel quale saranno necessarie le competenze apprese di tutte e tre le discipline; questo percorso didattico sarà erogato a partire dal prossimo anno scolastico (2017-2018) nel Milano Luiss Hub.

Lo STEAM [16,17] è un framework educativo che si propone di sviluppare competenze attraverso l'integrazione di cinque aree disciplinari, superando la separazione tra discipline tecnico-scientifiche e discipline umanistiche, così come quella tra sapere teorico e capacità pratiche. L'integrazione delle cinque aree disciplinari è basata su percorsi didattico-laboratoriali, veri e propri challenge interdisciplinari in cui gli studenti sono chiamati a:

- risolvere problemi complessi attraverso molteplici percorsi;
- comunicare efficacemente piani e idee;
- collaborare utilizzando tutti gli strumenti disponibili.

Lo STEAM[4] è un framework didattico, non un differente percorso curriculare. Non si tratta di apprendere diverse discipline, ma di sviluppare negli studenti il pensiero critico ed un approccio multidisciplinare.

In particolare, all'interno di questo approccio, la parte di Coding sarà basata sul linguaggio visuale Scratch, sviluppato dal MIT [13], che ha dimostrato di poter essere un valido strumento per favorire l'interesse degli studenti verso la programmazione [10].

In questo documento, in maniera *open source*, descriviamo in dettaglio questo percorso didattico, includendo riferimenti bibliografici, software utilizzato e attrezzature necessarie, per consentire a strutture interessate di poterlo replicare integralmente o, se necessario, adattarlo alle proprie esigenze: a tale scopo discutiamo anche alcune possibili varianti.

Questo documento è organizzato nel modo seguente: la prossima sezione descrive la struttura che ospita il nostro percorso didattico, il Milano Luiss Hub. Nella sezione successiva presentiamo in dettaglio il programma e l'organizzazione delle attività all'interno del percorso, mentre la Sezione 4 riepiloga la dotazione hardware e software. Infine, nella Sezione 5 concludiamo illustrando alcuni spunti di riflessione.

2 La struttura: il Milano Luiss Hub

Il Milano Luiss Hub nasce grazie a una collaborazione tra il Comune di Milano, che fornisce la struttura, e Fondazione Giacomo Brodolini, Italia Camp S.r.l e l'università LUISS Guido Carli che la gestiranno, allestendo lo spazio di circa mille metri quadri per realizzare un centro di innovazione in una zona centrale di Milano.

In particolare, la struttura (di prossima inaugurazione) sarà divisa nelle seguenti aree:

[4] Per una panoramica su iniziative internazionali legate al framework STEAM si consiglia il sito `https://steamedu.com/`.

Figura 1. Un rendering della struttura Milano Luiss Hub (sinistra) e l'allocazione degli spazi per le attività di Alternanza Scuola Lavoro (destra)

- **Innovation Gallery**: uno spazio per esposizioni e per sperimentazione di prototipi;
- **Learning Center**: uno spazio per la formazione, in cui organizzare workshop, corsi e seminari;
- **FabLab**: un spazio dedicato ai makers, in cui sono disponibili diverse tecnologie moderne di fabbricazione digitale come: Tagliatrici Laser, Plotter da Stampa e Taglio, Banco di Saldatura, Frese CNC e Stazioni di Prototipazione Elettronica;
- **Startup Accelerator**, che offre programmi dedicati a supportare lo sviluppo di nuove imprese innovative;
- **Alternanza scuola - lavoro**, divisa in due aree distinte: SteamLab e FabLab didattico.

Nella Figura 1 mostriamo un rendering della struttura e la planimetria dell'area dedicata all'alternanza scuola lavoro.

3 Programma e organizzazione delle attività

Coding, Robotica e Design 3D sono i tre ambiti, fortemente collegati, in cui si sviluppa il nostro progetto didattico dedicato a ragazzi delle scuole superiori.

I ragazzi, nelle 80 ore a disposizione, divisi in due settimane, otto ore al giorno dal lunedì al venerdì, devono realizzare un progetto complesso, che consiste nella realizzazione di un modello di macchina a guida autonoma in grado di parcheggiarsi da sola seguendo le linee disegnate sul pavimento e che identificano lo spazio di parcheggio, evitando ostacoli[5].

La macchina sarà realizzata usando i pezzi della Lego serie MindStorm EV3, che consente agli utenti di realizzare dei veri e propri robot autonomi, fornendo un ambiente di programmazione che ricorda molto quello di Scratch, non a caso utilizzato per il Coding. Gli studenti dovranno utilizzare le stampanti 3D, e

[5] Ci sono diverse varianti di questo progetto e altri progetti simili come difficoltà realizzativa, ma per semplicità di esposizione in questo documento ci limitiamo solo a questo progetto a scopo illustrativo, rinviando il lettore al sito del progetto dove troverà maggiori dettagli.

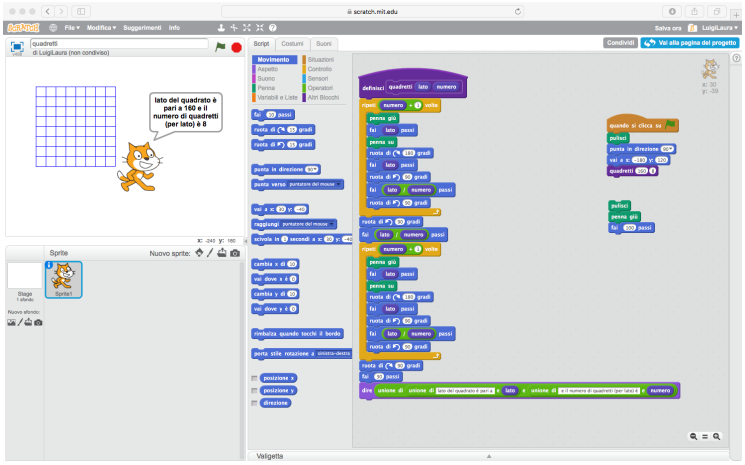

Figura 2. L'ambiente principale di Scratch, a sinistra lo *stage* (in cui si vede il gatto mascotte di Scratch) e a destra l'area per i comandi, rappresentati con blocchi geometrici che si incastrano tra di loro.

quindi i CAD 3D, per realizzare la carrozzeria della macchina, aiutandosi con forme 3D di pezzi Lego già disponibili nell'ambiente CAD (Tinkercad) prescelto.

La prima settimana sarà dedicata al Coding, attraverso il linguaggio Scratch e il suo ambiente visuale di sviluppo; i ragazzi impareranno Scratch lavorando a coppie con un solo computer, usando quindi la tecnica di sviluppo del software *Pair Programming*.

Nella seconda settimana i ragazzi impareranno dapprima i principi del Design di oggetti 3D, utilizzando il CAD Tinkercad, e le basi della stampa 3D, e poi si dedicheranno allo sviluppo del progetto usando Lego Mindstorms e il suo ambiente di programmazione visuale.

3.1 Coding

Il primo tassello del nostro percorso didattico è incentrato sull'acquisizione di competenze di Coding, utilizzando Scratch[6] come linguaggio/ambiente di programmazione.

Scratch è un linguaggio di programmazione visuale atipico, sviluppato dal team Lifelong Kindergarten ("Asilo a vita") del MIT. Dal 2003, anno in cui è stato creato, continua a riscuotere consensi per supportare l'insegnamento dei concetti di informatica a ragazzi di tutte le età, suscitando in loro un grande interesse [1,5,10,12,13] e contribuendo anche allo sviluppo di abilità di pensiero matematico [2].

A differenza dei linguaggi di programmazione "tradizionali", in cui l'utente utilizza un editor di testo per scrivere il codice che viene poi compilato in un

[6] `https://scratch.mit.edu/`

file eseguibile utilizzando un compilatore (e un linker per l'inclusione di librerie software), Scratch offre agli utenti un ambiente fortemente interattivo in cui c'è uno *stage*, al cui interno si muovono degli *sprite*. Gli sprite e lo stage possono essere controllati mediante istruzioni, rappresentate da blocchi geometrici, che vengono collegate tra di loro incastrando opportunamente i relativi blocchi geometrici. Ad esempio le istruzioni di ciclo (condizionato e non) sono rappresentate da blocchi a forma di tenaglia, al cui interno è possibile incastrare le istruzioni del ciclo. In Figura 2 è mostrato l'ambiente principale di Scratch.

Questo ambiente di sviluppo visuale è risultato molto intuitivo anche per utenti alle prime armi, che dopo pochi minuti sono già in grado di provare i loro primi programmi: questo costituisce un vantaggio probabilmente incolmabile con i linguaggi di programmazione "tradizionali", in cui l'utente riesce a far scrivere *Hello World!* al computer dopo una serie di attività di cui tipicamente comprende poco; inoltre, l'editor visuale evita che l'utente debba digitare il testo, risparmiandogli quindi, almeno in questa prima fase, gli errori sintattici che contribuiscono ad aumentare le difficoltà di utenti alle prime armi.

Scratch ha un ulteriore grande vantaggio dal punto di vista didattico: una comunità di utenti in continua espansione, molto attiva, che conta attualmente circa 17 milioni di persone e circa 21 milioni di progetti condivisi[7]. Un programma Scratch, chiamato progetto, può essere condiviso pubblicamente: a quel punto ogni utente può interagire con il progetto (premendo la bandiera verde), studiarne il funzionamento (mediante il pulsante *Guarda dentro*) o addirittura modificarlo (pulsante *Remix*). In questa maniera è molto facile, ad esempio, assegnare esercizi agli studenti creando progetti pubblici incompleti, che poi saranno completati dagli studenti.

Scratch ha anche una comunità di insegnanti, ScratchEd[8], dove scambiare idee, esperienze e materiali didattici.

Chiudiamo questa sezione dedicata al Coding parlando brevemente del *pair programming*, che è una pratica di sviluppo del software, nata nei primi anni novanta, in cui due programmatori lavorano insieme usando un solo computer [15]. A turno, uno dei due utilizza il computer mentre l'altro controlla continuamente quello che viene fatto; in questa maniera vengono ridotti sensibilmente piccoli errori sintattici e sviste ed è stato osservato che, nel complesso, lo sviluppo del software procede più rapidamente che non facendoli lavorare in parallelo.

Recentemente sono stati studiati gli effetti del pair programming anche nei primi corsi di programmazione, e i risultati in letteratura concordano nel sostenere che abbia effetti positivi anche sulla fase di apprendimento: in esperimenti in cui gli studenti sono stati divisi in due gruppi, uno in cui gli studenti lavoravano a coppie (pair programming) e l'altro in cui lavoravano singolarmente (gruppo di controllo), gli studenti che hanno lavorato a coppie sono andati consistentemente meglio degli studenti del gruppo di controllo, in maniera statisticamente più che significativa [8,11]. Inoltre, sorprendentemente, i risultati sembrano

[7] Statistiche aggiornate sono disponibili alla pagina `https://scratch.mit.edu/statistics/`

[8] `http://scratched.gse.harvard.edu`

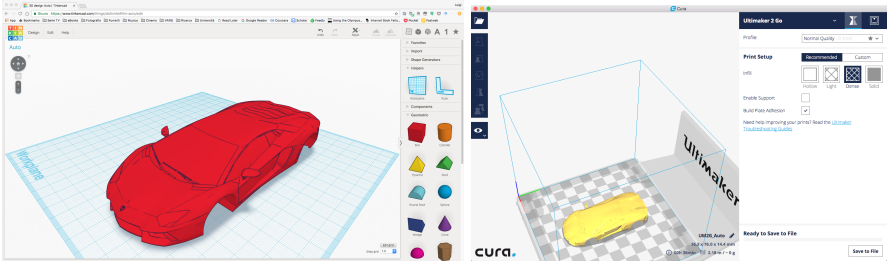

Figura 3. Progetto della carrozzeria dell'auto con Tinkercad (sinistra) e preparazione per la stampa 3D (slicing) con Cura (destra).

suggerire che il pair programming potrebbe incrementare le quote rosa tra i programmatori [11].

3.2 Design

Nel nostro percorso didattico STEAM il Design rappresenta probabilmente la componente più artistica. I ragazzi in questa fase vengono introdotti a tutte le fasi della stampa 3D: dalla progettazione dell'oggetto usando un CAD 3D alla fase di stampa vera e propria, con la scelta dei materiali, passando per la fase di slicing (preparazione del file per la stampa).

Come CAD 3D abbiamo deciso di usare Tinkercad[9], un CAD gratuito web based, che funziona in un browser. Tinkercad ha una interfaccia semplice e una serie di tutorial che guidano passo passo gli utenti alle prime armi. Inoltre, come per Scratch, c'è una community molto attiva che condivide progetti, che possono essere importati e modificati. Nella Figura 3 (lato sinistro) possiamo vedere l'interfaccia di Tinkercad. Gli studenti hanno anche a disposizione degli scanner 3D con i quali possono "importare" oggetti reali nelle loro creazioni.

Dopo che l'oggetto è stato creato con Tinkercad, prima di essere stampato deve essere preparato per la stampa utilizzando un software di *slicing*, che letteralmente fa a fette l'oggetto creando i diversi livelli della stampa additiva; ad esempio se vogliamo stampare uno sgabello con le gambe inclinate non è possibile stamparlo dal basso, in quanto le gambe cadrebbero durante la fase di stampa. Come software di slicing abbiamo scelto Cura[10], che è un software gratuito fatto dalla Ultimaker, la società che produce le stampanti 3D a disposizione degli studenti. Nella Figura 3 (lato sinistro) possiamo vedere l'interfaccia di Cura. Esistono anche software open source gratuiti, tra i quali probabilmente il più diffuso è Slic3r[11], ma abbiamo ritenuto di usare quello fornito con la stampante.

Dal momento che la fase stampa vera e propria dura diverse ore abbiamo raccomandato ai tutor di far partire le stampe nelle prime ore della giornata,

[9] `https://www.tinkercad.com/`
[10] `https://ultimaker.com/en/products/cura-software`
[11] `http://www.slic3r.org/`

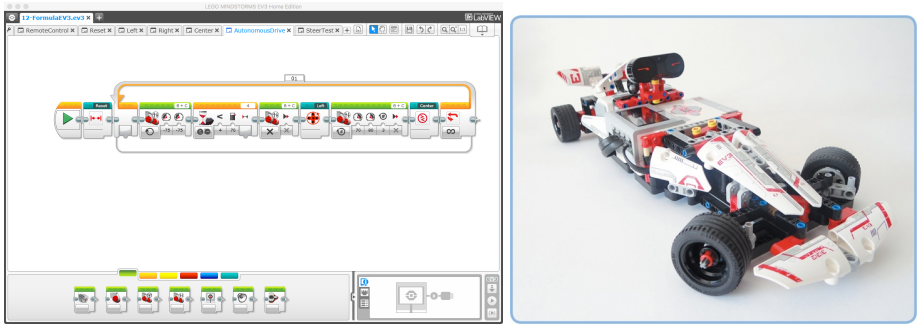

Figura 4. Ambiente di programmazione Lego Mindstorms EV3, con il codice per la guida autonoma in evidenza (sinistra) e l'automobile realizzata (destra).

in maniera che i ragazzi possano vedere completato il processo nella giornata stessa.

3.3 Robotica

L'ultimo tassello del nostro percorso didattico è costituito dalla Robotica, realizzata utilizzando la linea Lego Mindstorms EV3. Rispetto ai mattoncini Lego tradizionali possiamo dire che la Lego Mindstorms aggiunge un mattone programmabile, chiamato EV3 brick, a cui possono essere collegati sensori e attuatori mediante dei cavetti con connettori simili a quello telefonico (RJ-11). Il brick può essere programmato mediante un ambiente visuale simile a quello di Scratch, liberamente scaricabile dal sito della Lego. In Figura 4 possiamo vedere l'interfaccia dell'ambiente di programmazione EV3 (sinistra) e un veicolo realizzato seguendo le istruzioni prese dal testo [14]: possiamo vedere il brick nella parte centrale del veicolo, da cui partono i cavi diretti ai motori (due motori grandi per la trazione e un motore piccolo per lo sterzo) e ai sensori (di prossimità e IR, per poter anche controllare il veicolo a distanza mediante un telecomando).

La linea Mindstorms è stata introdotta dalla Lego nel 1996, e da allora ne sono state sviluppate diverse versioni, di cui la EV3 è la più recente, introdotta nel 2013. Il brick EV3 monta un processore ARM9 ed è dotato di un piccolo display (risoluzione 178x128), una porta USB, uno slot SD e connessione bluetooth per poter essere controllato usando uno smartphone (Robot Commander App[12]) o programmato mediante un tablet (EV3 Programmer App[13]).

In letteratura ci sono tantissimi studi relativi all'utilizzo della linea Mindstorms a fini educativi, per ragazzi di diverse età e tipologie [6,3].Non mancano anche i detrattori [9], con obiezioni basate principalmente sul costo che, seppur relativamente basso per un singolo kit, risulta assolutamente non trascurabile se si vuole dotare ogni ragazzo della classe di un proprio kit.

[12] `https://www.lego.com/en-us/mindstorms/downloads/robot-commander-app`
[13] `https://www.lego.com/en-us/mindstorms/apps/ev3-programmer-app`

4 Hardware e Software

In questa sezione riepiloghiamo la dotazione hardware e software necessaria per il percorso didattico proposto, distinguendo tra i requisiti minimi e la dotazione reale della struttura Milano Luiss Hub.

Per quello che riguarda l'hardware serve:

- un computer ogni due studenti (nella struttura ci sono 15 computer);
- (almeno) una stampante 3D (nella struttura ci sono 12 stampanti Ultramaker 2 go);
- (opzionale) scanner 3D (nella struttura ci sono 5 Scanner 3DSystem Sense2);
- Lego Mindstorm EV3 (nella struttura c'è il kit da 24 studenti che comprende 12 kit completi, quindi un kit ogni 2-3 ragazzi).

Per il software:

- un browser (e una connessione a Internet!), necessario per Scratch e Tinkercad;
- un software per lo *slicing*, ad esempio Slic3r (nella struttura usiamo Ultimaker Cura)
- il software per programmare i Lego Mindstorms.

5 Conclusioni

In questo articolo abbiamo presentato un progetto didattico, basato su Coding, Robotica e Design, che sarà erogato a partire dal prossimo anno scolastico (2017-2018) nel Milano Luiss Hub.

Nella Tabella 1 è mostrato il programma dettagliato delle due settimane delle attività. Il programma è diviso in blocchi didattici di due ore, in cui indicativamente i tutor fanno lezione e/o presentano gli esercizi per 15-20 minuti al massimo, mentre nel resto del tempo gli studenti sono impegnati in attività pratiche.

Abbiamo sviluppato anche dei moduli alternativi, qualora gli insegnanti o gli studenti abbiano richieste mirate, in cui gli studenti possono:

- creare app usando MIT AppInventor 2;
- creare dei siti web usando Wordpress;
- utilizzare Arduino o il Raspberry Pi per realizzare piccoli progetti.

Al termine di ogni percorso didattico svolto chiederemo agli studenti di compilare un questionario, che sarà usato, insieme al feedback degli insegnanti e dei tutor, per valutare l'esperienza fatta dai ragazzi e, possibilmente, migliorarla.

PRIMA SETTIMANA (CODING CON SCRATCH)					
	Lunedi	Martedì	Mercoledì	Giovedì	Venerdì
	Introduzione a Scratch	**Animazioni e Suoni**	**Variabili**	**Strutture di controllo e ricorsione**	**Mini-progetto finale in Scratch**
09-11	Introduzione a Scratch: concetti di base: lo stage, gli sprite, I costumi, le azioni	Gestione dei costumi, animazioni	Introduzione alle variabili, tipi di dato	Teorema di Bohm-Jacopini	Mini-progetto finale in Scratch: pianificazione attività
11-13	Primo videogioco in Scratch: Pong	Gestione dei suoni	Ricevere input dagli utenti	Ricorsione	Mini-progetto finale in Scratch: realizzazione
	Spostamenti e disegni	**Procedure**	**Strutture di controllo e Variabili booleane**	**Strutture di dati**	**Mini-progetto finale in Scratch**
14-16	Comandare gli spostamenti in Scratch	Build your own block (senza parametri)	Strutture di controllo: if, if/else, if annidati	Liste statiche	Mini-progetto finale in Scratch: realizzazione
16-18	Disegnare con la penna figure geometriche	Build your own block (con parametri)	Introduzione agli operatori dell'algebra booleana (AND, OR e NOT)	Liste dinamiche	Mini-progetto finale in Scratch: presentazione

SECONDA SETTIMANA (DESIGN 3D e ROBOTICA CON LEGO MINDSTORMS)					
	Lunedi	Martedì	Mercoledì	Giovedì	Venerdì
	Introduzione alla stampa 3D	**Prima stampa in 3D**	**Robotica: LEGO Mindstorm EV3**	**Progetto FINALE**	**Progetto FINALE**
09-11	Concetti di base della stampa 3D: scanner 3D, cad 3D, slicer e print	Stampa del primo modello 3D: preparare la stampa in Tinkercad	Introduzione a LEGO Mindstorm EV3: controllo del brick mediante telecomando	Introduzione al progetto finale	Progetto finale: realizzazione
11-13	Scanner 3D e repository di oggetti (es. thingiverse)	cenni su slicer open source, Gcode; introduzione e stampa con Cura (slicer Ultramaker)	Programmazione di Lego Mindstorm	Pianificazione del progetto e divisione in compiti	Progetto finale: realizzazione
	Tinkercad	**Esercitazione Stampa 3D**	**Robotica: sensori e attuatori**	**Progetto FINALE**	**Progetto FINALE: PRESENTAZIONE**
14-16	Introduzione a Tinkercad: interfaccia grafica e principali pannelli; primitive (solidi) e loro combinazioni	Mini-progetto stampa 3D	Introduzione ai sensori	Progetto finale: realizzazione	Presentazione dei progetti
16-18	Librerie di modelli in Tinkercad	Mini-progetto stampa 3D	Introduzione ai motori e ingranaggi	Progetto finale: realizzazione	Presentazione dei progetti

Tabella 1. Il programma dettagliato delle due settimane del percorso didattico.

Riferimenti bibliografici

1. Karen Brennan and Mitchel Resnick. Stories from the scratch community: Connecting with ideas, interests, and people. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, SIGCSE '13, pages 463–464, New York, NY, USA, 2013. ACM.
2. Luis Alberto Calao, Jesús Moreno-León, Heidy Ester Correa, and Gregorio Robles. Developing Mathematical Thinking with Scratch. In *Design for Teaching and Learning in a Networked World*, pages 17–27. Springer, 2015.
3. Ethan Danahy, Eric Wang, Jay Brockman, Adam Carberry, Ben Shapiro, and Chris B. Rogers. Lego-based robotics in higher education: 15 years of student creativity. *International Journal of Advanced Robotic Systems*, 11(2):27, 2014.
4. Direzione generale per gli ordinamenti scolastici e la valutazione del sistema nazionale di istruzione del MIUR, Attività di Alternanza Scuola Lavoro. Guida operativa per la scuola, 2015.
5. Filiz Kalelioglu and Yasemin Gülbahar. The effects of teaching programming via scratch on problem solving skills: A discussion from learners' perspective. *Informatics in Education*, 13(1):33, 2014.
6. Özgen Korkmaz. The effect of scratch-and lego mindstorms ev3-based programming activities on academic achievement, problem-solving skills and logical-mathematical thinking skills of students. *Malaysian Online Journal of Educational Sciences*, 4(3):73–88, 2016.
7. Legge n. 107 del 13 luglio 2015: Riforma del sistema nazionale di istruzione e formazione e delega per il riordino delle disposizioni legislative vigenti.
8. Charlie McDowell, Linda Werner, Heather Bullock, and Julian Fernald. The effects of pair-programming on performance in an introductory programming course. *ACM SIGCSE Bulletin*, 34(1):38–42, 2002.
9. Myles McNally, Michael Goldweber, Barry Fagin, and Frank Klassner. Do lego mindstorms robots have a future in cs education? In *SIGCSE*, volume 6, pages 61–62, 2006.
10. Orni Meerbaum-Salant, Michal Armoni, and Mordechai Ben-Ari. Learning computer science concepts with scratch. *Computer Science Education*, 23(3):239–264, 2013.
11. Nachiappan Nagappan, Laurie Williams, Miriam Ferzli, Eric Wiebe, Kai Yang, Carol Miller, and Suzanne Balik. Improving the cs1 experience with pair programming. *ACM SIGCSE Bulletin*, 35(1):359–362, 2003.
12. D Ozoran, N Cagiltay, and D Topalli. Using scratch in introduction to programming course for engineering students. In *2nd International Engineering Education Conference (IEEC2012)*, pages 125–132, 2012.
13. Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay Silver, Brian Silverman, and Yasmin Kafai. Scratch: Programming for all. *Commun. ACM*, 52(11):60–67, November 2009.
14. Laurens Valk. *Lego Mindstorms Ev3 Discovery Book: A Beginner's Guide to Building and Programming Robots*. No Starch Press, 2014.
15. Laurie Williams, Robert R Kessler, Ward Cunningham, and Ron Jeffries. Strengthening the case for pair programming. *IEEE software*, 17(4):19–25, 2000.
16. G Yackman. Recognizing the A in STEM education, 2007.
17. Georgette Yakman. What is the point of STE@M?–A Brief Overview. *Steam: A Framework for Teaching Across the Disciplines. STEAM Education*, 7, 2010.