SIRD Società Italiana di Ricerca Didattica
Giornale Italiano della Ricerca Educativa
Italian Journal of Educational Research

Project-based learning in computer science laboratory for education. A longitudinal study

La didattica per progetti nelle attività di laboratorio di informatica per l'educazione. Uno studio longitudinale

Sergio Miranda
University of Salerno, Dept. of Human Sciences, Philosophy and Education, Salerno (Italy)

Double blind peer review

Citation: Miranda, S. (2021). Project-based learning in computer science laboratory for education. A longitudinal study. *Italian Journal of Educational Research*, 27, 131-139.

Corresponding Author:Sergio Miranda
Email: semiranda@unisa.it

Received: October 5, 2021
Accepted: December 8, 2021
Published: December 23, 2021

Pensa MultiMedia / ISSN 2038-9744
https://doi10.7346/sird-022021-p131

Abstract
Teaching computer science in a humanities degree course is not a simple action, not because of the complexity of the subject itself, but because of the natural reluctance of students towards an apparently difficult discipline and in any case different from the others in their study plan. This paper describes an experience conducted at the University of Salerno with the aim of stimulating the students of the degree course in "Educational Sciences" to participate in the activities of the "Computer science laboratory for education" organized according to a project-based approach which leaves participants a wider margin of autonomy and which gives more space to their creativity. The results obtained are particularly encouraging and offer suggestions on methodologies to be used to increase involvement of students and stimulate them to develop their computational thinking.

Keywords: learning; computational thinking; creative programming; project-based learning.

Riassunto
Insegnare informatica in un corso di laurea umanistico non è un'azione semplice, non per la complessità della materia in sé, quanto per la naturale riluttanza degli studenti nei confronti di una disciplina apparentemente ostica e comunque diversa dalle altre presenti nel loro piano di studi. Questo lavoro descrive un'esperienza condotta all'Università di Salerno con l'obiettivo di stimolare gli studenti del corso di laurea in "Scienze dell'educazione" a partecipare alle attività di "Laboratorio di informatica per l'educazione" organizzate secondo un approccio per progetti che lascia ai partecipanti un più ampio margine di autonomia e che dà più spazio alla loro creatività. I risultati ottenuti sono particolarmente incoraggianti e offrono suggerimenti sulle metodologie da utilizzare per aumentare il coinvolgimento e stimolare gli studenti allo sviluppo del pensiero computazionale.

Parole chiave: learning; pensiero computazionale; programmazione creativa; apprendimento basato sui progetti.

1. Introduzione

Il pensiero computazionale è un processo logico-creativo che si può acquisire soprattutto con la pratica (Pellerey, 2018), una pratica che viene stimolata attraverso l'insegnamento dell'informatica e, più nello specifico, durante l'apprendimento del *coding* ovvero di quell'insieme di conoscenze e procedure che riguardano la programmazione del computer (Knuth, 1974; Papert, 1980; Wing, 2006). Quando si inizia da bambini, quest'attività contribuisce allo sviluppo di una forma mentis (Gardner, 1983) che consentirà di affrontare durante la crescita, problemi via via più complessi ma sempre seguendo un approccio sistematico (Wing, 2006).

Nel mondo della scuola e dell'università c'è la consapevolezza che il pensiero computazionale sia una delle cosiddette "competenze trasversali"[1] indispensabili in tutti gli ambiti disciplinari, ma, ad oggi, c'è una sostanziale difficoltà ad individuare un approccio valido ed efficace a stimolarne e supportarne lo sviluppo. Piuttosto di frequente capita che si insegni l'informatica ricadendo nello sterile approccio orientato alla trasmissione di concetti o, ancor peggio, all'addestramento all'uso di applicazioni software non ponendo l'accento sulla centralità del discente quale soggetto attivo del processo di apprendimento (Bellettini et al., 2018). In tal senso, un approccio metodologico ritenuto efficace (Quartapelle, 1999) è la didattica per progetti o apprendimento basato su progetti (*project based learning*).

2. Quadro teorico di riferimento ed obiettivi

Il pensiero computazionale è un ragionamento algoritmico attraverso il quale affrontare e risolvere i problemi ed è anche inteso come la capacità di scomporre un problema iniziale in sotto-problemi più semplici per affrontarli identificando una sequenza precisa di passi che siano in grado di portare ad una soluzione (Wing, 2006). Questa definizione è stata successivamente integrata da Cuny, Snyder, e Wing (2010, p. 1) secondo i quali il pensiero computazionale racchiude: "*i processi di pensiero coinvolti nella formulazione dei problemi e delle loro soluzioni in modo che le soluzioni siano rappresentate in una forma che può essere efficacemente eseguita da un agente di elaborazione delle informazioni*".

Il legame tra pensiero computazionale e *coding* ricorda quello analogo tra pensiero e linguaggio (Vygotskij, 1962); per Vygotskij, infatti, pensiero e linguaggio seguono linee di sviluppo differenti e solo quando il linguaggio viene interiorizzato, promuove lo sviluppo del pensiero.

Secondo Romero, Lepage e Lille (2017), quando si insegnano materie tecnico-scientifiche come il *coding*, in corsi di laurea dell'area umanistica capita spesso che sia la presenza in aula, sia la partecipazione ed il coinvolgimento degli studenti siano ridotti. Per questi autori, le cause sono imputabili non tanto alla complessità della materia, nonostante venga spesso resa nella forma più elementare possibile, e nemmeno al frequente rifiuto da parte degli studenti di avvicinarsi a concetti di natura tecnico-scientifica che quindi ricordano, seppur, vagamente, la matematica o le scienze, quanto all'incapacità o allo scarso utilizzo, spesse volte, di approcci adeguati all'insegnamento di questa disciplina.

Conoscere una disciplina non vuol dire essere capaci di insegnarla (Nigris, 2016) e spesso si incorre in quello che Bruner (1978) definiva *problema della conversione* e che Chevallard (1985) riprende successivamente come problema di *trasposizione didattica dei saperi*.

Adottare metodologie *learner centred* consente di rivolgere l'attenzione alle esigenze dei partecipanti intervenendo sulla loro motivazione (Mezirow, 2003). La motivazione è in relazione reciproca con l'apprendimento significativo, nel senso che l'una implica l'altra e viceversa (Ausubel, 1963). Il docente, pur assumendo un ruolo apparentemente marginale, resta comunque il riferimento capace di favorire l'apprendimento motivando i partecipanti ed incrementandone il coinvolgimento (Margiotta, 2014) rafforzando il legame tra i bisogni dei partecipanti e le conoscenze, tra la teoria e l'esperienza pratica (Bruner, 1966). In questa ottica, il *project based learning*, partendo da un problema reale, con l'obiettivo di trovarne possibili soluzioni, consente di favorire lo sviluppo di apprendimenti capaci di risvegliare gli interessi legati

1 Iniziativa DIGCOMP della Direzione Generale Europea per la Cultura e l'Educazione del 2010; Piano nazionale per la scuola digitale nella Legge 107/2015, comma 56.

alle identità dei partecipanti (Wenger, 2006) facendone emergere, nella ricerca della soluzione, oltre che le capacità gestionali di organizzazione delle attività, l'interazione sociale tra i partecipanti (Kilpatrick, 1918) e la creatività (De Bartolomeis, 1978; David, 2008).

La creatività è sempre stata oggetto di ricerche e studi in ambito sia psicologico che pedagogico: ad esempio Bruner (2005) ne individua le caratteristiche "*nell'abilità e nell'attitudine ad intuire in modo immediato possibili relazioni formali, prima ancora di saperle dimostrare in un orizzonte logico*" (p. 55). Da un'azione creativa tramite la quale un individuo crea elementi originali, nasce una "*sorpresa produttiva*" e, nel contempo, si attivano processi metacognitivi. La creatività è un processo che porta alla creazione di nuove idee, di nuovi concetti, di nuovi legami logici tra concetti già esistenti o di trasformazioni in qualcosa di nuovo (Guilford, 1950). Proprio una componente del pensiero creativo permette di analizzare e valutare possibili alternative nella risoluzione di un dato problema, modellando e adattando le conoscenze acquisite. Questo accade, in particolare, nella fase di elaborazione, ovvero quando il pensiero genera creativamente differenti risposte al problema dato e le ricombina in modo originale per giungere alla soluzione (Guilford, 1970).

In tale prospettiva, l'insegnamento, inteso non solo come un'azione limitata agli anni di scuola, deve promuovere questo tipo di approccio durante tutto l'arco della vita, valorizzando il potenziale creativo dei discenti anche in età adulta. Questo discorso trova riscontri più recenti in quanto la creatività è vista come una competenza chiave all'interno di diversi quadri per l'istruzione del ventunesimo secolo (Voogt & Roblin, 2012) poiché capace di innescare processi metacognitivi che portano ad un apprendimento significativo anche in relazione all'uso delle tecnologie (McGuinness & O'Hare, 2012). In questo senso, gli insegnanti dovrebbero sviluppare le loro capacità di stimolare l'uso creativo delle tecnologie e, in particolare, della programmazione, poiché essa non riguarda solo la scrittura di codice, ma anche la capacità di analizzare una situazione, identificare i suoi componenti chiave, modellare dati e processi e creare o perfezionare un programma (Knuth, 2014). La programmazione dei computer ha dunque una intrinseca natura creativa. È fondamentale farla venire fuori. Occorre individuare una metodologia di insegnamento che la faccia percepire come uno strumento di modellazione e costruzione della conoscenza che consenta attività creative di risoluzione dei problemi e che, quindi, spinga al coinvolgimento dei partecipanti. Quando gli studenti vengono coinvolti in un'attività di programmazione creativa, essi riescono a utilizzare "*ambienti basati sulla tecnologia per costruire modelli rappresentazionali dei fenomeni che vengono studiati*" (Jonassen & Strobel, 2006, p. 2). La natura fortemente interattiva degli ambienti all'interno dei quali si sviluppano i programmi fa il resto. Gli studenti possono realizzare presto dei prototipi per testare i modelli che concepiscono e questo ha un enorme potenziale formativo (Ke, 2014).

Nonostante ciò, le attività di programmazione necessitano di una integrazione. La programmazione dovrebbe essere considerata come una strategia pedagogica e non solo uno strumento tecnico o un insieme di tecniche di codifica da apprendere (Pierce, 2013). Mentre alcuni usi delle tecnologie coinvolgono lo studente in modo passivo o vagamente interattivo, altri usi lo coinvolgono in un processo creativo di costruzione della conoscenza in cui la tecnologia stessa mira a migliorare il processo di apprendimento. L'impegno nell'insegnamento alla programmazione può essere distinto in vari livelli. Tra questi, i livelli più bassi sono rappresentati da lezioni frontali e trasferimenti di concetti, mentre il livello più alto è dato proprio dalla programmazione creativa, inteso come un'attività che riesce a coinvolgere ogni studente nel processo di progettazione e sviluppo di un'opera originale (Romero, Lepage & Lille, 2017). In questo approccio, gli studenti sono incoraggiati a utilizzare la programmazione come mezzo per la co-costruzione della conoscenza. Grazie all'uso di particolari ambienti[2], possono inventare una storia, concependone l'ambientazione, la trama, i personaggi. Ciò non accade in altre attività dove il percorso di apprendimento e gli obiettivi sono definiti in modo da garantire che tutti gli studenti siano in grado di raggiungere gli stessi risultati. Queste attività di apprendimento, per quanto possano essere graduali ed efficaci, non sollecitano il livello di pensiero e le strategie cognitive e metacognitive che invece attività di programmazione co-creativa possono stimolare (Papert, 1992). In particolare, quando le richieste non sono particolarmente dettagliate e presentano, cioè, un certo livello di incertezza e di complessità, lo studente è ancor di più

2 Esistono numerosi ambienti di programmazione completamente visuale. Scratch (https://scratch.mit.edu/) è uno dei più diffusi ed è quello utilizzato nelle attività descritte in questo articolo.

stimolato ad impegnarsi innanzitutto per comprendere il problema da trattare e poi per modellare, strutturare, sviluppare e perfezionare un programma che sia originale e capace di risolvere il problema posto.

Il *project based learning*, dalla pianificazione del problema (problem posing) alla individuazione della soluzione (problem solving), riesce a far intrecciare la conoscenza teorica e la conoscenza pratica (Kilpatrick, 1936) e ad attivare un sano "apprendere operando" (Paparella, 2006) capace di focalizzare l'attenzione sulle conoscenze teoriche realmente rilevanti per la pratica (Zecca, 2014). Quindi, siccome per stimolare lo sviluppo del pensiero computazionale occorre individuare e formulare dei quesiti per i quali il processo da seguire, o il risultato stesso potrebbero non essere univocamente determinati, il *project based learning* assume una particolare rilevanza nell'ambito della tematica qui descritta e della ricerca dell'approccio più efficace. Studi recenti (García-Peñalvo et al., 2018) dimostrano quanto l'apprendimento basato sui progetti abbia portato a risultati decisamente positivi in contesti simili. Occorre proporre problematiche in cui ci possa essere più di una soluzione implementabile, in cui il processo di modellazione del problema consenta poi l'applicazione di tante possibili soluzioni tutte altrettanto valide ma concepite secondo un approccio costruttivista, in cui ai discenti che affrontano il problema venga dato un margine di libertà e uno spazio per la creatività (Miller & Krajcik, 2019; Shin et al., 2021).

L'apprendimento basato su progetti presuppone che siano rispettati alcuni principi fondamentali (Thomas, 1998). Innanzitutto, il progetto deve vertere su tematiche centrali rispetto al curriculum di studi e, in particolare, rispetto alla disciplina trattata. Deve stimolare l'investigazione in chiave costruttivista, coinvolgendo i partecipanti nella ricerca della soluzione. Il progetto deve essere realistico e basato su problemi autentici per i quali non basta la mera applicazione di una soluzione "scolastica" (Thomas, 2000; Markham, 2011). In questo modo, i partecipanti vengono stimolati nella ricerca della soluzione e si riduce la distanza tra il "sapere sapiente" e il "sapere insegnato", tra "il soggetto" e "l'oggetto culturale" (Vygotskij, 1974). L'aula si trasforma così in un laboratorio all'interno del quale si possono affrontare i problemi della vita reale e la classe dei partecipanti si trasforma in una "comunità di pratica" (Wenger, 2006).

Nell'instaurare una relazione circolare dalla teoria alla pratica e viceversa, l'apprendimento basato sui progetti rappresenta un approccio capace di incoraggiare la partecipazione degli studenti e di aumentarne l'effettivo coinvolgimento (Markham, Larmer & Ravitz, 2003; Marzano et al., 2017). L'obiettivo di questo lavoro è stato di adottare un approccio di didattica per progetti con gli studenti iscritti al secondo anno del corso di laurea triennale in "Scienze dell'educazione" presso l'Università degli Studi di Salerno frequentanti il "Laboratorio di informatica per l'educazione" per verificare in qual misura ciò abbia influenzato la partecipazione attiva degli studenti, il loro coinvolgimento e, infine, se abbia stimolato lo sviluppo del pensiero computazionale.

3. L'approccio proposto

La progettazione didattica, la ricchezza degli scambi comunicativi, la costruzione collettiva del sapere, il rendere gli studenti protagonisti, sono tutte condizioni che contribuiscono al raggiungimento dei risultati di apprendimento (Johnson et al., 1996). Nel contesto specifico dell'insegnamento del *coding* nei corsi di laurea in materie umanistiche, i destinatari sono sostanzialmente dei novizi ovvero degli studenti che hanno ridotte conoscenze in materia o che non ne hanno affatto e ciò rende il loro apprendimento molto simile a ciò che accade nei primi cicli di istruzione (Brennan & Resnick, 2013). Ciò lascia pensare che per tale insegnamento, serva una metodologia diversa dalle lezioni frontali (Pierce, 2013). Durante le lezioni di "*Laboratorio di informatica per l'educazione*", un insegnamento del secondo anno del corso di laurea triennale in "*Scienze dell'educazione*" presso l'Università degli Studi di Salerno, è stato sperimentato un approccio di *project based learning*.

Le lezioni si articolano in un percorso di 36 ore (3 CFU) delle quali una prima parte è relativa ad argomenti teorici (la descrizione del *pensiero computazionale* e delle attività che ne favoriscono lo sviluppo) e una seconda dedicata alla pratica della programmazione.

Sono stati presi in considerazione 3 anni accademici dal 2017-18 al 2019-20 Nella seconda parte delle lezioni, l'impostazione adottata nei primi due anni è stata legata alla presentazione di esercizi per i quali il procedimento risolutivo era sostanzialmente unico (problemi "ben definiti"). Durante l'anno accademico 2019/20 è stato adottato il *project based learning* puntando allo sviluppo del pensiero computazionale at-

traverso compiti di apprendimento relativi a problemi "autentici" (Thomas, 2000) tali da lasciare margini di libertà nella scelta di una tra più soluzioni possibili (problemi "non ben definiti"). Agli studenti è stato assegnato un progetto con l'obiettivo di realizzare un gioco educativo e lasciando libertà nella ricerca della soluzione da adottare. Per ogni progetto, è stato indicato un argomento a scelta tra quelli trattati nella prima parte del corso e l'obiettivo di costruire impiegando l'ambiente Scratch un gioco educativo sull'argomento prescelto, facendo in modo che la finalità formativa e la finalità ludica fossero evidenti, ma la cui logica di funzionamento e di interazione con il giocatore sarebbe stata tutta da inventare ed implementare dagli studenti. Gli studenti sono stati divisi in gruppi di massimo 3 partecipanti. Gli argomenti trattati nelle lezioni teoriche riguardanti i principi base dell'informatica e dell'architettura dei calcolatori elettronici, sono stati riepilogati e mostrati in modo da consentire a ciascun gruppo di sceglierne uno, evitando sovrapposizioni. Completata la scelta, i gruppi hanno sviluppato i progetti in totale autonomia pianificando impegni, attività e suddivisione dei compiti nell'arco di due settimane di lavoro. Durante le ultime lezioni, tutti i gruppi hanno presentato i lavori svolti.

Per poter verificare se questo approccio didattico abbia prodotto risultati in termini di coinvolgimento dei partecipanti, a fine corso è stata condotta un'indagine utilizzando *e-Lena*, una piattaforma messa a disposizione dal *Laboratorio di Ricerca in Media Education e Didattica Attiva* (*Rimedi@*) dell'Università degli Studi di Salerno e realizzata attraverso una personalizzazione di *Moodle*.

A tutti i partecipanti è stato somministrato un questionario con l'obiettivo di raccogliere le opinioni in merito all'impostazione delle attività laboratoriali, all'approccio per progetti adottato, ai benefici ottenuti nell'apprendimento del *coding*.

4. I risultati della sperimentazione

Agli studenti che hanno preso parte alle attività didattiche nell'anno accademico 2019/20, è stato somministrato un questionario predisposto con l'obiettivo di raccogliere, su una scala di tipo Likert a 4 livelli, le percezioni su questa modalità di svolgimento delle attività didattiche. Hanno risposto al questionario 146 studenti. Gli esiti della rilevazione sono riportati in Tab.1.

	Livello	**No.**	**%**
Q1. Questa esperienza di programmazione in Scratch pensi abbia cambiato il tuo modo di affrontare i problemi?	*No*	9	6.2
	Più No che Sì	11	7.5
	Più Sì che No	19	13.0
	Sì	107	73.3
Q2. Questa è stata la tua prima esperienza con Scratch?	*No*	55	37.7
	Sì	91	62.3
Q3. La realizzazione di un progetto ti ha stimolato ad approfondire l'uso di Scratch?	*No*	4	2.7
	Più No che Sì	21	14.4
	Più Sì che No	62	42.5
	Sì	59	40.4
Q4. La realizzazione di un progetto ti ha stimolato ad approfondire gli argomenti teorici del corso?	*No*	13	8.9
	Più No che Sì	18	12.3
	Più Sì che No	27	18.5
	Sì	88	60.3
Q5. La realizzazione di un progetto in Scratch richiede molto tempo?	*No*	22	15.1
	Più No che Sì	15	10.3
	Più Sì che No	16	11.0
	Sì	93	63.7

Q6. La realizzazione di un progetto in Scratch ha migliorato la tua preparazione?	*No*	42	28.8
	Più No che Sì	16	11.0
	Più Sì che No	18	12.3
	Sì	70	47.9
Q7. La realizzazione di un progetto in Scratch aumenta i tempi di preparazione dell'esame?	*No*	12	8.2
	Più No che Sì	46	31.5
	Più Sì che No	69	47.3
	Sì	19	13.0
Q8. Programmare in Scratch è semplice?	*No*	13	8.9
	Più No che Sì	33	22.6
	Più Sì che No	78	53.4
	Sì	22	15.1
Q9. Realizzare un progetto di un gioco educativo in Scratch ti è piaciuto?	*No*	25	17.1
	Più No che Sì	31	21.2
	Più Sì che No	34	23.3
	Sì	56	38.4

Tabella 1: Risultati del questionario

Per valutare la consistenza interna del questionario nel suo complesso, è stato calcolato il coefficiente di Cronbach (1950). Valori di tale coefficiente superiori allo 0,8 esprimono una buona consistenza interna e il valore calcolato sul questionario proposto è pari a 0,977.

Analizzando le percentuali di risposte alla prima domanda (Q1) è possibile affermare che i partecipanti abbiano correttamente associato il concetto di sviluppo del pensiero computazionale alla programmazione in *Scratch* pur essendo, per circa una metà dei partecipanti (Q2), la loro prima esperienza con il *coding*. Il questionario inoltre ha raccolto dati in merito alle percezioni dei partecipanti verso questa esperienza di programmazione in Scratch (Q8 e Q9) facendo emergere che il *coding* non è stato percepito come un processo complicato e come la realizzazione del progetto è stata un'esperienza accolta con favore.

Venendo, infine, alle domande più esplicite sull'approccio metodologico adottato, le opinioni dei partecipanti risultano essere decisamente positive (Q3, Q4 e Q6) nonostante venga sottolineato come siano richiesti un maggior tempo nello studio e nella preparazione dell'esame (Q5 e Q7).

Oltre ad esaminare gli esiti di questa rilevazione, sono stati analizzati i dati relativi agli appelli di esame prendendo in considerazione le percentuali di studenti che hanno superato l'esame durante i primi appelli.

L'esame finale consiste in un test di verifica di 10 domande a scelta multipla somministrato sempre all'interno della piattaforma *e-Lena*, appositamente costruito per valutare non soltanto le conoscenze acquisite, ma anche particolari abilità di comprensione, analisi e applicazione di concetti specifici del *coding* nell'ambiente *Scratch*. Per superare l'esame e conseguire un esito positivo (*superato*), occorre rispondere correttamente ad almeno 6 domande su 10. I test, pur se formulati in maniera casuale selezionando 10 domande estratte da un archivio di oltre un centinaio di quesiti, sono costruiti in modo da riguardare tutte le parti del programma[3].

3 I quesiti usati nelle prove sono stati opportunamente calibrati e sottoposti ad *item analysis* per verificarne difficoltà, potere discriminante, selettività ed affidabilità (Trinchero, 2006). Le caratteristiche della prova adottata nei tre anni considerati sono state sempre la stesse, ovvero 10 quesiti a risposta multipla a cui rispondere in un intervallo di tempo limitato impiegando la piattaforma *e-Lena*.

I dati raccolti riguardano i numeri di studenti iscritti e i numeri di studenti che superano l'esame nei primi due appelli dei tre anni accademici indicati: 2017/18, 2018/19 e 2019/20 (Tab. 2).

	Iscritti	**Superamento esame al I appello**	**Superamento esame I appello (%)**
Studenti a.a. 2017/18	319	116	36,4
Studenti a.a. 2018/19	313	113	36,1
Studenti a.a. 2019/20	260	153	58,8

Tabella 2: Esiti relativo al superamento dell'esame di "Laboratorio di informatica per l'educazione"

La percentuale di studenti che hanno superato l'esame al primo appello è passata da circa il 36% degli anni 2017/18 e 2018/19 a circa il 59% nel 2019/20. Anche questi risultati sembrano confermare come l'approccio metodologico adottato nell'ultimo anno accademico possa aver favorito il miglioramento degli apprendimenti in relazione alle tematiche trattate.

5. Conclusioni

Nell'apprendimento basato sui progetti, attraverso il *problem posing* e il *problem solving*, le attività proposte sono in grado di promuovere l'apprendimento favorendo il nesso tra conoscenze teoriche e abilità pratiche (Kilpatrick, 1936; Paparella, 2011). Questa metodologia, anche nel nostro caso, ha consentito di impegnare gli studenti in un processo di indagine strutturato su compiti complessi, che ha posto l'accento sulla centralità del progetto e sull'approccio costruttivista che esso richiede (Marzano et al., 2017).

L'obiettivo della ricerca è stato di verificare se il *project based learning* potesse influenzare la partecipazione attiva degli studenti, favorendone il coinvolgimento per stimolare lo sviluppo del pensiero computazionale. Il *project based learning* offre agli studenti la possibilità di confrontarsi con problemi autentici e permette di sviluppare l'autonomia, il pensiero critico, la collaborazione, la capacità di comunicare, la creatività e di favorire apprendimenti significativi, dando, inoltre, ai partecipanti una forte spinta motivazionale (Panciroli et al., 2018). Il *project based learning* è identificato in letteratura come un approccio capace di attribuire agli studenti autonomia e responsabilità (Thomas, 2000), dando loro la possibilità di costruire qualcosa sulla base della loro stessa *prior knowledge* e di attivare processi di *deep understanding* (Shin et al., 2021). Questo lascia pensare che esso possa essere ritenuto una metodologia efficace nell'insegnamento del *coding* e dell'informatica. Il *coding*, infatti, aiuta a sviluppare il pensiero computazionale ma va impiegato affiancando altre metodologie e strategie (Gabbari et al., 2020). I dati rilevati, in conclusione, seppur circoscritti e relativi a una indagine di natura esplorativa, sono abbastanza incoraggianti e possono lasciare spazio ad ulteriori riflessioni sia sull'efficacia della metodologia adottata, sia su come essa possa fornire un ulteriore stimolo allo sviluppo del pensiero computazionale.

Riferimenti bibliografici

Ausubel, D. (1963). *The Psychology of Meaningful Verbal Learning,* Grune & Stratton, New York

Bellettini, C., Violetta, L., Malchiodi, D., Monga, M., & Morpurgo, A. (2018). Informatica e pensiero computazionale: una proposta costruttivista per gli insegnanti. *Proc. of Didamatica 2018*.

Brennan, K., & Resnick, M. (2013). Imagining, creating, playing, sharing, reflecting: How online community supports young people as designers of interactive media. In *Emerging technologies for the classroom* (pp. 253-268). New York: Springer.

Bruner, J. S. (1966). *Toward a Theory of Instruction* (Trad. it. *Verso una teoria dell'istruzione*, Armando, Roma, 1982).

Bruner, J. S. (1978). *Dopo Dewey. Il processo di apprendimento nelle due culture*. Roma: Armando (Trad. it. di A. Armando, *The process of education*, Harvard University Press Cambridge).

Bruner, J. S. (2005). *Il conoscere. Saggi per la mano sinistra*. Roma: Armando.

Chevallard, Y. (1985). *La transposition didactique. Du savoir savant au savoir enseigné.* Grenoble: La Pensée Sauvage.
Cronbach, L. J. (1950). Further evidence on response sets and test design. *Educational and Psychological Measurement*, 10, 3-31.
Cuny, J., Snyder, L., & Wing, J. M. (2010). *Demystifying computational thinking for non-computer scientists*. https://www.cs.cmu.edu/link/research-notebook-computational-thinkingwhat-and-why.
David, J.L. (2008). What Research Says About/Project-Based Learning. *Educational Leadership Teaching Students to Think*, 65, 5, 80-82.
De Bartolomeis, F. (1978). *Il sistema dei laboratori*. Milano: Feltrinelli.
Fabbri, L. (2007). *Comunità di pratiche e apprendimento riflessivo. Per una formazione situata.* Roma: Carocci.
Freire, Paul (1970) *La pedagogia degli oppressi*. Versione Italiana Mondadori ed. 1971.
Gabbari, M., Gagliardi, R., Gaetano, A., & Sacchi, D. (2020). Integrare "Coding e Pensiero computazionale" nella didattica. *Azioni, Tecnologie e competenze: esperienze in presenza e a distanza OPPInformazioni,* 128, 86-100.
Gardner, H. (1983). *Frames of Mind: The Theory of Multiple Intelligences (*Versione Italiana: *Formae mentis. Saggio sulla pluralità dell'intelligenza*, Feltrinelli, Milano, 1987).
García-Peñalvo, F. J., Reimann, D., & Maday, C. (2018). Introducing Coding and Computational Thinking in the Schools: The TACCLE 3 – Coding Project Experience. In M. S. Khine (Ed.), *Computational Thinking in the STEM Disciplines. Foundations and Research Highlights* (pp. 213-226). Cham, Switzerland: Springer.
Guilford, J. P. (1950). Creativity. *The American Psychologist*, 5(9), 444-454.
Guilford, J. P. (1970). Creativity: retrospect and prospect. *Journal of Creative Behaviour*, 4(2), 149-169.
Johnson, David, W., Johnson, Roger, T. Johnson, Holubec, Edythe (1996). *Apprendimento cooperativo in classe: migliorare il clima emotivo e il rendimento* (Trad., Lucio Marinelli, *Guide per l'educazione,* Erickson, Trento).
Jonassen, D., & Strobel, J. (2006). Modeling for meaningful learning. In *Engaged learning with emerging technologies* (pp. 1-27). Dordrecht: Springer.
Ke, F. (2014). An implementation of design-based learning through creating educational computer games: A case study on mathematics learning during design and computing. *Computers & Education*, 73, 26-39.
Kilpatrick, W. H. (1918). The project method. *Teachers College Record*, 19, 319-335.
Kilpatrick, W. H. (1936). *Foundations of Method. Informal Talks on Teaching*. New York: Macmillan.
Knuth, Donald, E. (1974). Computer Science and its relation to Mathematics. *The American Mathematical Monthly*, 81, 4, 323-343, April 1974.
Knuth, Donald, E. (2014). *Art of Computer Programming*, Volume 2: Seminumerical Algorithms. Addison-Wesley Professional.
Margiotta, U. (2014). Insegnare, oggi, all'Università. Un master per la didattica universitaria. *Formazione & Insegnamento*, XII, 1, 89-105.
Markham, T., Larmer, J., & Ravitz, J. (2003). *Project based learning handbook: A guide to standards-focused project based learning for middle and high school teachers*. Novato, CA: Buck Institute for Education.
Markham, T. (2011). Apprendimento basato su progetti. *Bibliotecario insegnante*, 39(2), 38-42. Enciclopedia.
Marzano, A., Vegliante, R., Miranda, S., & Formisano, M.A. (2017). La didattica per progetti nell'insegnamento di Metodologie e tecniche della ricerca educativa. *Giornale italiano della ricerca educativa*, 19, 227-239.
McGuinness, C., & O'Hare, L. (2012). Introduction to the special issue: New perspectives on developing and assessing thinking: Selected papers from the 15th international conference on thinking (ICOT2011). *Thinking Skills and Creativity*, 7(2), 75-77.
Mezirow, J. (2003). Transformative learning as discourse. *Journal of Transformative Education*, 1, 58-63.
Miller, E. C., & Krajcik, J. S. (2019). Promoting deep learning through project-based learning: A design problem. *Disciplinary and Interdisciplinary Science Education Research*, 1(1), 7.
Panciroli, C., Corazza, L., Vignola, P., Marcato E., & Leone D. (2018). Didattica innovativa. Soluzioni efficaci per contesti complessi. *Form@re - Open Journal per la formazione in rete*, 18, 2, 116-129.
Paparella, N. (2006). *Le attività di laboratorio e tirocinio nella formazione universitaria. Indagini e strumenti* (Vol. II). Roma: Armando.
Paparella, N. (2011). Insegnare per competenze in università. Modelli, procedure, metodi. In L. Galliani, C. Zaggia, & A. Serbati, *Apprendere e valutare competenze. Progettazione e sperimentazione di strumenti nelle lauree magistrali* (pp. 45-58). Lecce: Pensa MultiMedia.
Papert, S. (1980) *Mindstorms: Children, Computers, And Powerful Ideas*. (Ristampa 2020). Basic Books.
Papert, S. (1992). *The Children's machine*. New York: BasicBooks.
Pellerey, M. (2018). Educare al pensiero computazione: un'esigenza per i processi di formazione professionale. *Rassegna CNOS*, 34, 2, 37-51.
Pellerey, M. (2018). Educare al pensiero computazionale: alcuni approfondimenti e relativi apporti formativi. *Rassegna CNOS*, 34, 3, 45-58.
Pierce, M. (2013). Coding for middle schoolers: Next-generation programming languages for children are taking

up where Logo left off and teaching young students how to code to learn. *THE Journal [Technological Horizons In Education]*, 40(5).
Quartapelle, F. (Ed.).(1999). *Didattica per progetti*. Milano: Franco Angeli.
Romero, M., Lepage, A. & Lille, B. (2017) Computational thinking development through creative programming in higher education. *International Journal of Educational Technology in Higher Education,* 14, 42.
Shin, N., Bowers, J., Krajcik, J. et al. (2021) Promoting computational thinking through project-based learning. *Discip Interdscip Sci Educ Res* 3, 7.
Thomas, J. W. (1998). *Project-based learning: Overview*. Novato, CA: Buck Institute for Education.
Thomas, J. W. (2000). *A review of research on project-based learning*. San Rafael, CA: The Autodesk Foundation.
Trinchero r. (2006). *Valutare l'apprendimento nell'e learning. Dalle abilità alle competenze*. Trento: Erickson.
Voogt, J., & Roblin, N. P. (2012). A comparative analysis of international frameworks for 21st century competences: Implications for national curriculum policies. *Journal of Curriculum Studies*, 44(3), 299-321.
Vygotskij, L. S. (1962). *Thought and language*. Chicago: The MIT press (versione italiana: *Pensiero e linguaggio* del 2007 edita da Giunti).
Vygotskij, L. S. (1974). *Storia dello sviluppo delle funzioni psichiche superiori e altri scritti*. Firenze: Giunti-Barbera.
Wenger, E. (2006). *Comunità di pratica. Apprendimento, significato e identità*. Milano: Raffaele Cortina (Edizione originale pubblicata 1998).
Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35.
Zecca, L. (2014). Tra 'teorie' e 'pratiche': studio di caso sui Laboratori di Scienze della Formazione Primaria all'Università di Milano Bicocca. *Giornale Italiano della Ricerca Educativa*, VII, 13, 215-230.