[ITADINFO]

CONVEGNO ITALIANO
SULLA DIDATTICA DELL'INFORMATICA

Dal codice al prompt

Ridefinire apprendimento e valutazione della programmazione nell'era degli assistenti IA

Giovanni Pedroncelli

IISS "G. Marconi – M. Hack", Bari

Indice

1 Il problema e la domanda di ricerca

2 Fondamenti teorici

3 Il framework: quattro modalità d'uso

4 La rubrica del prompt-come-specifica

5 Esempio, validazione e conclusioni

Il problema

- Gli assistenti LLM risolvono la maggior parte degli esercizi di programmazione
- Gli studenti della secondaria li usano già, di solito senza mediazione didattica
- Il divieto non regge; l'uso non regolato espone gli inesperti a rischi metacognitivi
- La ricerca è quasi solo universitaria e descrittiva: **manca un framework per la scuola secondaria italiana**

La domanda di ricerca

Quali configurazioni didattiche consentono di mantenere e sviluppare le competenze fondamentali della programmazione quando la produzione sintattica del codice è parzialmente delegabile a un agente generativo?

La risposta proposta: quattro modalità d'uso che regolano il ruolo dell'IA e spostano il baricentro dalla produzione del codice a interpretazione, validazione e specificazione.

Il prompt come astrazione lossy

Dijkstra, 1978

- La difficoltà non viene dall'obbligo del formalismo
- Il rigore simbolico rende impossibili non senso e incoerenza
- Le lingue naturali li tollerano

Riletto oggi

- Dal codice al prompt si perde informazione
- Un buon prompt vincola: requisiti, casi limite, criteri di accettazione
- Il prompt è una **specifica dei requisiti**, contenuto disciplinare

Che cosa resta da insegnare

Complessità accidentale

- Sintassi e verbosità del linguaggio
- Limiti degli strumenti
- Gli LLM **la comprimono**, come l'alto livello rispetto all'assembly

Complessità essenziale

- Scomposizione del problema e algoritmo
- Modello dei dati, testing, debugging
- Resta allo studente: **è il nucleo invariante**

Leggere viene prima di scrivere

- Tracciare e spiegare il codice precede e predice il saperlo scrivere (Lopez et al.)
- Interpretare e produrre sono due assi distinti (tassonomia di Fuller et al.)
- Gli LLM abbassano il costo del produrre senza presidiare l'interpretare
- **Rischi:** illusione di competenza, automation bias, divario crescente tra studenti

Quattro modalità d'uso dell'IA

0

Pre-delega

Nessun supporto generativo

Focus
Modello mentale

1

IA tutor

Analizza e spiega codice esistente

Focus
Interpretazione

2

IA pair programmer

Genera codice sotto supervisione

Focus
Progettazione e validazione

3

IA esecutrice

Realizza artefatti da specifiche

Focus
Specificazione

Che cosa si verifica, che cosa si previene

Modalità	Evidenza osservabile	Verifica	Rischio mitigato
0 · Pre-delega	Traccia correttamente l'esecuzione	Tracing, Parsons problems	Delega prematura
1 · IA tutor	Individua errori nelle spiegazioni	Spiegare in modo chiaro	Automation bias
2 · Pair programmer	Costruisce test efficaci	Test suite	Illusione di competenza
3 · IA esecutrice	Produce specifiche valutabili	Rubrica del prompt	Fallacia della codifica in linguaggio naturale

Una progressione parametrica

ITT Informatica e Telecom.

- Distesa sul triennio
- Modalità 0–1 in terza, 2 in quarta, 3 in quinta
- Regressioni locali a ogni nuovo paradigma

Liceo Scienze Applicate

- Due ore settimanali di Informatica
- Microciclo ricorsivo in ogni nucleo tematico
- Modalità 3: modelli computazionali e simulazioni

La rubrica del prompt-come-specifica

Completezza dei requisiti

Funzionali e non funzionali, assunzioni

Ambiguità e casi limite

Input non validi, eccezioni, condizioni al contorno

Vincoli architetturali

Strutture dati, responsabilità, efficienza, sicurezza

Criteri di accettazione

Verificabili a priori: test, asserzioni, casi rappresentativi

Quattro livelli: Base · Intermedio · Avanzato · Esperto

Esempio: i criteri di accettazione

Base	Intermedio	Avanzato	Esperto
Non definisce come valutare la correttezza dell'output	Dice solo che il programma deve "funzionare"	Definisce condizioni di successo, esempi di input/output, casi di test	Criteri verificabili a priori, strategia di testing, asserzioni

Oltre il prompt finale

Traccia delle revisioni

- Le iterazioni sono debugging della specifica
- Scheda a tre campi: problema, modifica, risultato
- Distingue revisioni consapevoli da tentativi

Audit di sicurezza

- Codice corretto ma vulnerabile (Pearce et al.)
- Checklist: eccezioni, validazione input, credenziali in chiaro
- Raccordo con la cybersecurity

Un esempio: la biblioteca scolastica

Modalità	Che cosa fa lo studente	Che cosa si valuta
0 · Pre-delega	Rappresenta a mano libri, utenti, prestiti; tabelle di esecuzione	Prevede il comportamento del sistema
1 · IA tutor	Fa spiegare all'IA il codice dato e lo confronta col proprio modello	Spiega il codice, trova errori
2 · Pair programmer	Scrive prima la test suite, poi fa generare presta_libro()	Il codice generato supera i suoi test?
3 · IA esecutrice	Scrive la specifica completa del sistema	Qualità della specifica (rubrica)

Modalità 3: una specifica di livello esperto

*"Progetta un sistema object-oriented in Python per una biblioteca scolastica, **separando catalogo, utenti e gestione prestiti**. La ricerca di un libro per ISBN deve avere **complessità O(1)**: scegli la struttura dati di conseguenza. Ogni prestito registra data di consegna e scadenza (30 giorni). **Un utente non può avere più di tre prestiti attivi**. Gestisci con **eccezioni tipizzate**: libro inesistente, libro non disponibile, utente inesistente, limite prestiti raggiunto, restituzione di un prestito non registrato. Fornisci una **test suite con asserzioni** per ciascuno di questi cinque casi più un prestito andato a buon fine."*

In un solo prompt: requisiti · casi limite · vincoli architetturali · criteri di accettazione

Validazione: il disegno previsto

Il disegno

- Prima: affidabilità inter-valutatore della rubrica
- Quasi-sperimentale, su classi parallele
- Progressione a modalità vs uso non regolato della GenAI
- Almeno due cicli tematici completi

L'ipotesi falsificabile

- Comprensione del codice **non inferiore** a gruppi senza GenAI
- Specificazione e progettazione **superiori** a gruppi con uso non regolato

Conclusioni

La proposta

Un framework a quattro modalità che regola il ruolo dell'IA, con verifiche e rubrica.

Perché è utile a scuola

Una terza via tra divieto e laissez-faire, sul terreno della disciplina.

Che cosa manca ancora

Validazione empirica: affidabilità della rubrica, poi studio quasi-sperimentale.

[ITADINFO]

CONVEGNO ITALIANO
SULLA DIDATTICA DELL'INFORMATICA

Grazie per l'attenzione

gi.pedroncelli@gmail.com

IISS "G. Marconi – M. Hack", Bari