Politecnico di Torino

Laurea Magistrale in Ingegneria Informatica

Dal testo al codice: uno strumento per incoraggiare l'analisi degli esercizi di programmazione

Candidato: Andrea Bruno
Relatori: Luigi De Russis, Juan Pablo Sáenz, Fulvio Corno

Introduzione

Gli esercizi di programmazione sono uno strumento fondamentale a disposizione degli studenti neofiti di informatica. Tramite la loro risoluzione, lo studente non solo mette in pratica le regole di sintassi apprese dalla teoria, ma impara ad adottare buone pratiche di programmazione e strategie di problem solving.
Gli studenti svolgono questi esercizi avvalendosi del supporto di IDE (ambienti di sviluppo integrato dove si può scrivere ed eseguire il proprio codice) e applicazioni interattive che forniscono un numero sempre maggiore di funzionalità per il supporto alla didattica, come per esempio la visualizzazione delle strutture dati presenti nel codice, o la verifica automatica della correttezza del programma tramite una serie di test.
Lo svolgimento di un esercizio è un processo complesso che non prevede solo la scrittura di un programma risolutivo, ma anche una fase preliminare di comprensione e analisi del testo del problema, decomposizione in sottoproblemi e passaggio dalle richieste dell'esercizio (espresse in linguaggio naturale) a concetti di natura informatica. Gli studenti tendono però a trascurare o sottovalutare tali fasi, preferendo un approccio di programmazione meno ragionato in cui scrivono direttamente il loro programma, andando però a discapito delle loro prestazioni e della qualità del loro codice.
Spesso anche gli IDE e le applicazioni precedentemente citati non aiutano gli studenti ad affrontare al meglio i loro esercizi: nella maggior parte dei casi il supporto fornito da tali strumenti si concentra sulla scrittura del codice, lasciando scoperte le fasi iniziali di analisi e comprensione del problema.

Obiettivi della tesi

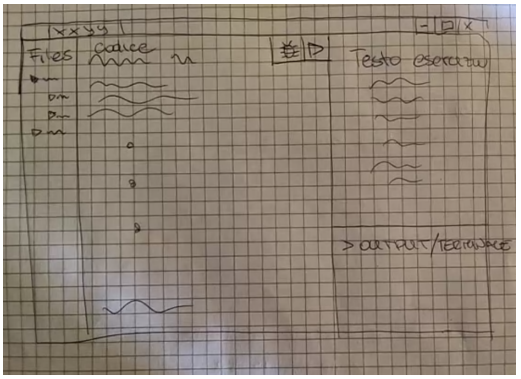
L'obiettivo della tesi è supportare gli studenti universitari dei corsi introduttivi di informatica nelle fasi di analisi e decomposizione degli esercizi di programmazione.
A tale scopo sono state quindi individuate le necessità e le difficoltà incontrate dagli studenti nel compiere tali operazioni, oltre alle funzionalità fornite da applicazioni attualmente disponibili con caratteristiche rilevanti ai fini della tesi. Queste informazioni sono poi state utilizzate per progettare, implementare e validare un'interfaccia (sotto forma di applicazione web) che fornisca supporto nelle fasi iniziali di analisi e decomposizione di un esercizio di programmazione.

Revisione della letteratura

Per supportare gli studenti durante la fase di analisi di un esercizio, è importante analizzare il modo in cui loro affrontano tali esercizi, e in particolare quali sono le criticità su cui si può intervenire e le tecniche già adottate da altri per risolvere problemi simili.
Le pubblicazioni prese in esame hanno evidenziato come l'analisi del testo di un esercizio e la decomposizione in sottoproblemi abbiano effetti positivi sul processo di apprendimento degli studenti e sulla qualità dei loro programmi. Gli studenti tendono però a preferire metodi meno strutturati di risoluzione degli esercizi. In alcuni casi gli studenti si preoccupano solo della correttezza sintattica del loro programma, in altri casi hanno problemi nell'analisi dell'esercizio o non pensano neanche alla possibilità di esaminarlo nel dettaglio prima di scrivere il codice, mentre alcuni studenti pensano che impiegare del tempo per analizzare un esercizio sia un indicatore di una loro scarsa capacità di programmazione.
È inoltre emerso che gli strumenti didattici per il supporto alla programmazione sono attualmente in uso e apprezzati. Nonostante il numero e varietà di tali applicazioni, non ne è emersa una con tutte le caratteristiche auspicate negli obiettivi di questa tesi. In tal senso, le applicazioni usate in ambito didattico esaminate in precedenza hanno proposto funzionalità come la visualizzazione delle strutture dati, la proposta

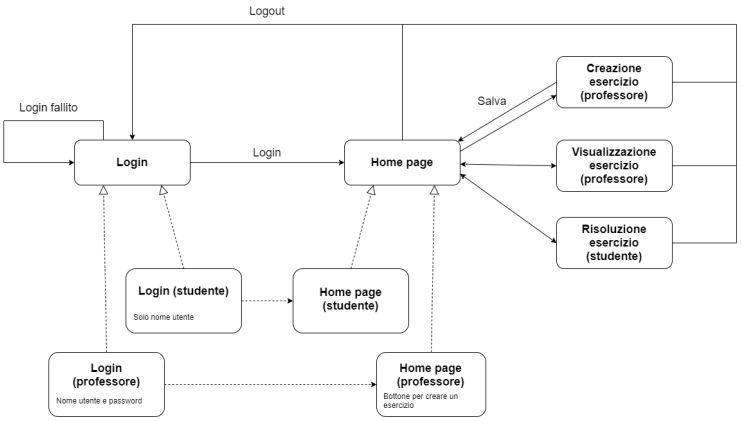
di domande per guidare lo studente, la pianificazione del proprio lavoro e la suddivisione dell'esercizio in sottoproblemi. Gli strumenti e IDE professionali hanno invece messo in risalto in modi diversi l'importanza del collegare il proprio codice con artefatti e documenti che ne forniscano requisiti e contesto.

Needfinding



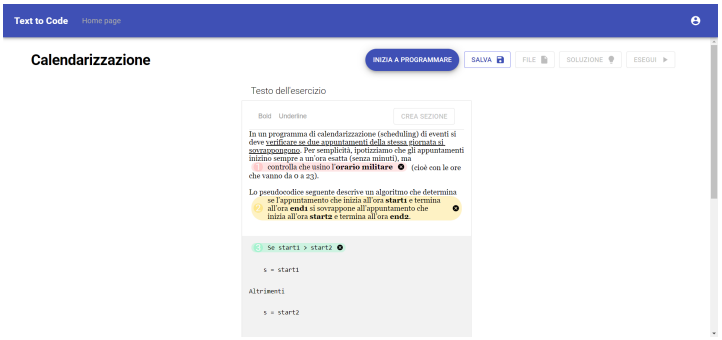
Le informazioni ottenute durante la fase precedente sono state integrate da uno studio che ha permesso di approfondire le abitudini di programmazione degli studenti e, più nello specifico, il loro approccio al problem solving durante lo svolgimento dei loro esercizi di informatica.
Lo studio ha coinvolto dieci studenti al primo anno di laurea triennale e ha previsto la compilazione di un questionario e la successiva partecipazione a un'intervista, al termine della quale è stato chiesto di immaginare e disegnare un'interfaccia che lo studente pensa lo aiuterebbe a risolvere esercizi di informatica.
Da quanto emerso nello studio, durante la fase di analisi del problema gli studenti tendono ad annotare il testo dell'esercizio, mentre non ricorrono a commenti all'interno del codice. È stato inoltre riportato l'uso di annotazioni e schemi, ma tali artefatti vengono prodotti solo per risolvere dubbi di natura immediata, e una volta prodotti non vengono più consultati.
Non sono emerse difficoltà specifiche legate all'analisi e traduzione degli esercizi, i problemi più citati sono legati alla scelta di algoritmi e strutture dati da usare. Nella maggior parte dei casi gli studenti hanno detto di consultare i loro professori per dubbi legati a comprensione e analisi del problema, mentre si confrontano con i colleghi per dubbi legati al loro programma.
Gli studenti che analizzano approfonditamente il testo trovano lento il loro processo di risoluzione degli esercizi, ma riconoscono il vantaggio di una maggiore chiarezza del loro processo logico e visione d'insieme. Chi ignora la fase di analisi, al contrario, lamenta la tendenza a rimanere bloccato in soluzioni più complesse del necessario e il rischio di dover modificare del tutto la struttura del programma che ha inizialmente scritto.

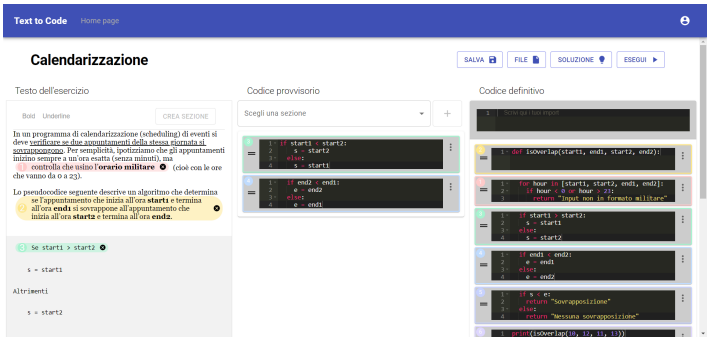
Progettazione



L'analisi della letteratura e i dati ottenuti dallo studio hanno contribuito a definire una serie di caratteristiche che sono state utilizzate come punto di partenza per la progettazione dell'applicazione. Le funzionalità più importanti emerse da tale analisi sono state la suddivisione del testo in sottoproblemi, la presentazione ravvicinata del testo del problema e del codice, la necessità di sollecitare l'analisi del problema prima di iniziare a scrivere il programma, un processo di analisi relativamente veloce, il supporto nella scelta di algoritmi e strutture dati, la possibilità di annotare il testo e il supporto da parte del professore.
L'applicazione supporterà due tipi di utenti: i professori (autorizzati a caricare nuovi esercizi) e gli studenti (che possono risolvere gli esercizi presenti nel sistema). Una volta autenticato, l'utente avrà accesso alla pagina principale in cui può scegliere di risolvere un problema di informatica (se studente), oppure creare un nuovo esercizio o visualizzarne uno già esistente (se professore).
L'interfaccia di risoluzione dell'esercizio permetterà allo studente di prendere visione del testo dell'esercizio e di suddividerlo in sottoproblemi evidenziando il testo stesso. Per incoraggiare l'analisi del problema, inizialmente lo studente interagirà solo con il testo e, una volta analizzato il problema, si potrà passare alla schermata completa in cui è possibile scrivere il proprio programma sotto forma di frammenti di codice, ciascuno assegnato a uno specifico sottoproblema. I frammenti potranno essere provvisori o definitivi, per

lasciare allo studente la libertà di sperimentare soluzioni diverse nel proprio programma. I frammenti di codice definitivo, in ordine, costituiranno il testo del problema in fase di esecuzione.
All'interno della pagina lo studente può inoltre salvare il proprio lavoro, gestire file di testo, visualizzare la suddivisione in sottoproblemi proposta dal professore ed eseguire il proprio programma.
Il professore, durante la creazione dell'esercizio, potrà supportare lo studente nell'analisi del problema. In punti specifici del testo dell'esercizio potranno essere inserite delle domande che sollecitino la riflessione sull'approccio da seguire, e il professore stesso potrà proporre una sua suddivisione del testo in sottoproblemi che aiuti lo studente in caso di dubbi.





Implementazione del sistema

Il progetto è stato implementato in JavaScript: il backend si basa su Express, mentre il frontend usa React.
Il backend comunica con il frontend tramite un'interfaccia REST, e si occupa di memorizzare i dati del sistema usando un database non relazionale in-memory sfruttando la libreria NeDB (che a sua volta implementa MongoDB), che memorizza le collezioni all'interno di file binari invece che in un DBMS, data la natura prototipale del sistema. La scelta di un database non relazionale è dovuta alla natura stratificata dei dati, rappresentabili in modo più efficace tramite oggetti.

Valutazione

L'interfaccia è stata infine valutata tramite dei test di usabilità per individuare eventuali problemi di efficacia nelle singole funzionalità offerte. Durante lo svolgimento dei test è stato chiesto a otto studenti del Politecnico di Torino di svolgere una serie di brevi attività incentrate sulle funzionalità principali dell'interfaccia, come ad esempio la suddivisione di un esercizio in sottoproblemi, oppure la creazione, il riordinamento e la modifica di frammenti di codice. Gli studenti hanno poi risposto a delle domande e a un questionario relativi alla loro esperienza con l'interfaccia.
Dallo svolgimento delle attività e dalle risposte dei partecipanti sono emersi una serie di possibili cambiamenti per migliorare l'usabilità dell'interfaccia, tra cui avere un qualche tipo di introduzione ai concetti di sezioni di testo e frammenti di codice, rendere più evidente la differenza tra domande opzionali e obbligatorie e rendere più intuitiva l'interazione con la soluzione proposta dal professore.

Conclusioni e sviluppi futuri

Il lavoro svolto nell'ambito di questa tesi ha permesso di raggiungere gli obiettivi prefissati, cioè comprendere le difficoltà e le abitudini degli studenti di informatica nella risoluzione di esercizi di programmazione, per poi costruire e validare un'applicazione web pensata a tale scopo.
I risultati ottenuti in questa tesi possono essere ampliati svolgendo ulteriori osservazioni e test dell'interfaccia, ad esempio durante le esercitazioni di laboratorio. Altre possibilità di sviluppo di questa tesi possono partire dai risultati dei test di usabilità che hanno fornito utili spunti di miglioramento, come ad esempio una presentazione iniziale dei concetti di sezione di testo e frammento di codice.
L'applicazione ha inoltre il potenziale per essere utilizzata all'interno di un corso di Informatica come supporto allo studio individuale o nelle esercitazioni di laboratorio. Dato che lo strumento è attualmente un prototipo, si renderebbero necessarie modifiche come la pubblicazione in rete, il potenziamento delle misure di sicurezza e l'ampliamento delle funzionalità offerte ai professori.