TDD

TDD, o Test-Driven Development (Sviluppo Guidato dai Test), è una metodologia di sviluppo del software che pone l'accento sulla scrittura di test automatici prima dell'implementazione del codice effettivo. L'approccio TDD segue un ciclo noto come "Red-Green-Refactor" (Rosso-Verde-Rifattorizzazione), che comprende i seguenti passaggi:

1. **Fase Rosso (Red):** Scrivere un test automatico che descriva una funzionalità che ancora non esiste. In questa fase, il test fallirà perché la funzionalità non è ancora stata implementata.

2. **Fase Verde (Green):** Scrivere il codice minimo necessario per far passare il test. L'obiettivo è soddisfare il requisito del test.

3. **Fase Rifattorizzazione (Refactor):** Ottimizzare il codice senza cambiare il suo comportamento, mantenendo i test verdi. La rifattorizzazione mira a migliorare la struttura e la leggibilità del codice.

Questo ciclo viene ripetuto iterativamente per ciascuna funzionalità o requisito del software. L'obiettivo principale del TDD è garantire che il codice sia sempre accompagnato da test automatizzati e che ogni pezzo di codice aggiunto o modificato sia supportato da test.

Ecco alcuni principi chiave del TDD:

1. **Test Automatici Prima del Codice:**
    - Scrivere i test prima di iniziare a implementare il codice effettivo.

2. **Test Incrementali:**
    - Aggiungere test incrementalmente per ciascuna funzionalità o requisito.

3. **Semplicità e Minimalismo:**
    - Scrivere solo il codice sufficiente per far passare i test.

4. **Rifattorizzazione Sicura:**
    - Eseguire la rifattorizzazione solo quando tutti i test sono verdi per evitare la regressione.

    - Seguire rigorosamente il ciclo Red-Green-Refactor per ogni iterazione.

I benefici del TDD includono:

- **Miglioramento della Qualità del Codice:** Poiché ogni pezzo di codice è accompagnato da test automatizzati, è meno probabile che vengano introdotti errori durante le modifiche successive.
- **Documentazione Vivente:** I test servono anche come documentazione eseguibile del comportamento del sistema.
- **Riduzione dei Bug:** Identificare e correggere i bug in fasi più precoci dello sviluppo.
- **Design Migliore:** TDD incoraggia un design del software che è facilmente testabile, modulare e adattabile.
- **Rapid Feedback:** Il ciclo veloce di scrittura di test, implementazione e rifattorizzazione fornisce un feedback rapido sullo stato del software.

L'adozione di TDD richiede una disciplina e una pratica costanti, ma può portare a un