Corso di Laurea in Ingegneria Informatica

DynaCheq

- Progetto per il corso di Fondamenti di Informatica II -

Anno accademico 2002/2003

Finalità

Il progetto consiste nello sviluppo di una semplice applicazione software ed è finalizzato a familiarizzare lo studente con un corretto utilizzo di concetti di programmazione object-oriented (tipici di C++).
Si richiede allo studente di organizzare in modo chiaro e organico la struttura del programma sviluppato, fornendone inoltre un'opportuna documentazione.
I concetti da utilizzare sono:

- derivazione tra classi (eventualmente classi astratte) e metodi virtuali
- comportamenti polimorfici
- gestione dell'I/O su file
- classi template (e/o STL) – *opzionale*
- gestione delle eccezioni – *opzionale*

Il giudizio finale sul progetto è teso a valutare se le finalità sono state raggiunte o meno.

Organizzazione del progetto

Si consiglia di sviluppare il progetto procedendo per passi successivi, come sotto indicato:

- Identificazione delle funzionalità dell'applicazione
- Definizione delle classi da utilizzare nel programma, con relative gerarchie
- Definizione della struttura del programma
- Progetto dettagliato delle classi e delle strutture dati utilizzate dal programma
- Implementazione di un **prototipo** del programma con interfaccia testuale
- Arricchimento e consolidamento del prototipo, fino a giungere alla **versione finale** del prodotto.

Traccia: DynaCheq

Il progetto, denominato "DynaCheq", consiste nello sviluppo di un programma che gestisce una partita a scacchi tra due giocatori.
A questo scopo si deve prevedere l'utilizzo una struttura dati che modellizza la scacchiera, e di un insieme di oggetti che rappresentano i pezzi del gioco.
Una opportuna gerarchia di classi dovrà essere progettata per organizzare le proprietà dei diversi tipi di pezzi e i loro movimenti.
L'esecuzione del programma consiste in una serie di passi, e in ciascun passo si deve specificare la mossa da effettuare. E' compito del programma, in conseguenza dell'input ricevuto ad ogni passo,

1) Verificare la liceità della mossa
2) Aggiornare lo stato del gioco
3) Fornire la rappresentazione a video della situazione aggiornata.

Le regole di gioco adottate sono quelle classiche degli scacchi, eventualmente con qualche semplificazione (p.es. senza alcun tipo di arrocco).

La rappresentazione a video potrà essere fatta tramite l'interfaccia testuale, oppure *opzionalmente* tramite un file di output interpretabile da un altro strumento di visualizzazione (p.es. un web browser).
Si deve poter caricare da file una configuarazione iniziale da cui far iniziare il gioco.

Si deve tener traccia di tutte le mosse effettuate, e poter salvare una partita effettuata; si deve poter caricare una partita, visionarla, andare all'ultima mossa presente e proseguirla.
Opzionalmente, si può fornire la funzionalità di UNDO sulle mosse.
Opzionalmente, si può implementare una semplicissima strategia di gioco da parte del calcolatore.

Requisiti aggiuntivi

- Ogni singolo progetto deve essere portato avanti da un gruppo di due persone.
- Il programma deve essere sviluppato utilizzando l'ambiente DEV C++; l'adozione di altri IDE dovrà essere preventivamente discussa con il docente.
- Il programma dovrà utilizzare almeno una gerarchia di classi in cui sia presente almeno un livello di derivazione.
- Il funzionamento del programma dovrà prevedere l'accesso ad almeno una struttura dati dinamica.
- Per qualsiasi problematica inerente il progetto, gli esercitatori a cui gli studenti devono far riferimento sono Alessio Bechini per il gruppo A-L, e Luigi Palopoli per il gruppo M-Z.

Documentazione richiesta

Si richiede una adeguata documentazione del lavoro svolto, da organizzare in due sezioni.
La prima sezione deve contenere una descrizione **funzionale** dell'applicazione sviluppata, che deve anche assumere il ruolo di "manualistica" minimale del programma per tutte le categorie di utenti a cui è destinato.
La seconda sezione deve essere dedicata alla documentazione **strutturale** del sistema, riportando ***necessariamente*** i class diagram (con nortazione UML) delle classi utilizzate, una descrizione chiara della struttura del programma, ed eventualmente qualche cenno agli algoritmi utilizzati.

Suggerimenti

Sviluppare il progetto *in maniera incrementale*, implementando poche classi alla volta, compilandole e successivamente testandole.
Leggere le informazioni fornite dal compilatore sugli errori rilevati: spesso contengono indicazioni utili per individuare l'errore, *che non necessariamente si troverà alla linea indicata*.
Tener presente che molti degli errori segnalati dal compilatore sono fittizi, e scompariranno una volta eliminati gli errori "a monte" nel codice.
Ricordarsi di includere sempre i file header delle librerie utilizzate.
Cercare di adottare, per quanto possibile, una organizzazione standard dei file sorgenti e dei file con le dichiarazioni.
Per individuare e correggere gli errori che si manifestano a tempo di esecuzione, utilizzare le funzionalità offerte dal debugger (in particolare, breakpoint e ispezione del contenuto delle variabili).

Non andare a ricevimento con errori di sintassi nel codice presentato, per evitare di fare figure meschine con i docenti: provare almeno a compilare preventivamente le classi del progetto.

Consegna del progetto

Lo studente, una volta terminato il progetto, dovrà provvedere a consegnare nelle date stabilite quanto segue:

- Documentazione sotto forma cartacea, stilata secondo i criteri precedentemente menzionati
- Codice sorgente *dell'intero programma* su floppy disk, che deve essere *direttamente compilabile nell'ambiente DEV*
- Programma eseguibile, su floppy disk.

Al momento della consegna allo studente verrà richiesto di illustrare specifiche porzioni del codice sorgente da lui scritto, ed eventualmente di apportarvi particolari modifiche.