Calcolatrice estendibile

Progetto Programmazione ad Oggetti, a.a. 2011/2012

dott. Francesco Tapparo

1 Scopo

Lo scopo del progetto è lo sviluppo in C++ di una calcolatrice estendibile mediante una interfaccia utente grafica (GUI) sviluppata in Qt©; questa calcolatrice dovrà essere in grado di gestire almeno tre diversi tipi di dati (non necessariamente numerici), la cui implementazione è lasciata allo studente.

La specifica **dettagliata** di quali funzionalità il progetto debba fornire è lasciata in massima parte a libera scelta dello studente, che può trarre ispirazione dal buon senso, dai propri interessi e dalla fantasia. Al fine di chiarire meglio le funzionalità richieste il progetto è integrato da un eseguibile GNU/Linux che implementa una bozza delle specifiche minimali richieste. Per ragioni di tempo non tutti i dettagli potrebbero essere implementati pienamente o testati ma l'eseguibile può tornare utile per chiarire dubbi sull'interpretazione della specifica. Al solito, il docente rimane disponibile nell'orario di ricevimento per chiarire i dubbi degli studenti.

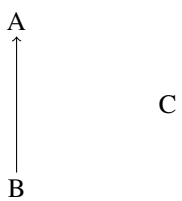
In particolare la calcolatrice deve soddisfare i seguenti requisiti:

1. rigida separazione tra il codice che gestisce la GUI e il codice che gestisce la parte logica della calcolatrice; quest'ultima deve essere incapsulata in una o più classi che non hanno alcuna relazione con le librerie o il codice grafico e che potrebbero altrettanto legittimamente essere usate come parte logica di una calcolatrice a riga di comando.

2. la parte logica della calcolatrice deve includere almeno tre classi non triviali (cioè che non siano semplici wrapper attorno a tipi primitivi del c++), e che non siano una delle classi sviluppate durante il corso, come `orario` o `dataora`, corrispondenti ai tre tipi gestiti dalla calcolatrice. Lo studente non è per nulla obbligato a scegliere tipi matematici o numerici, a patto che gli altri requisiti del progetto vengano rispettati.

3. per quanto concerne la parte logica del progetto vi deve essere una rigida separazione tra il codice generale della parte logica (ad esempio il codice che si occupa dell'interpretazione dei comandi dati alla calcolatrice) e il codice che si occupa dei tipi gestiti dalla calcolatrice. Ad esempio se la calcolatrice gestisce operazioni tra complessi, tutte le operazioni tra complessi e tutta la gestione di errori relativi a complessi (come ad esempio un'errore "divisione per zero") potrebbero essere gestite all'interno di una classe `Complex`.

4. lo studente è libero di scegliere la gerarchia delle classi che gestiscono i tipi su cui lavora la calcolatrice, ma vi deve essere una gerarchia minima come segue:



La gerarchia deve quindi comprendere due classi di cui una eredita dall'altra, e un'altra classe che non eredita nè è classe base di nessuna delle altre due. È ovviamente

permesso ereditare tutte le classi da una ulteriore classe base comune. Inoltre la classe `B` dell'esempio sopra deve offrire delle funzionalità aggiuntive rispetto a quelle offerte dalla classe base (ad esempio se la classe base è `Reale` e la classe derivata è `Intero`, `Intero` deve offrire un'operazione che ha senso solo per gli interi e che è si attiva solo quando il tipo corrente è un intero, ad esempio l'operazione di calcolo del fattoriale). Si intende che la calcolatrice deve comportarsi coerentemente con questa gerarchia: ad esempio se il tipo corrente è `B`, devono essere disponibili non solo tutte le operazioni disponibili per `A` ma anche le operazioni addizionali disponibili per il tipo `B`. Inoltre se qualche operazione applicata a oggetti di tipo `B` porta ad altri oggetti di tipo `B`, questo deve rispecchiarsi nell'operatività della calcolatrice. Nell'esempio di prima se l'oggetto corrente è un intero deve essere disponibili tutte le operazioni della calcolatrice, come l'addizione o la radice quadrata, ma se eseguo la radice quadrata il tipo corrente diventerà un reale, mentre se eseguo l'addizione otterrò un altro intero.

5. la calcolatrice deve essere dotata di una completa gestione degli errori; ogni tipo di errore (ad esempio la divisione per zero) deve provocare un messaggio all'utente, senza corrompere lo stato della calcolatrice o peggio provocarne un crash.

2 Interfaccia Grafica

L'eseguibile che accompagna e complementa il presente documento illustra l'impostazione di base di una semplice GUI. La libreria Qt è dotata di una documentazione completa e precisa che insieme alle lezioni di laboratorio tenute durante il corso sarà la principale guida di riferimento nello sviluppodella GUI. La libreria Qt offre una moltitudine di classi e metodi per lo sviluppo di GUI curate, dettagliate, accattivanti e "user-friendly". Sarà apprezzato l'uso di funzionalità di Qt diverse da quelle minimali.

3 Valutazione del Progetto

Un buon progetto dovrà essere sviluppato seguendo i principi basilari della programmazione orientata agli oggetti, anche per quanto concerne l'interfaccia grafica. La valutazione del progetto prenderà in considerazione i seguenti criteri:

1. Correttezza del progetto: deve compilare e funzionare correttamente, ed assolvere pienamente e correttamente lo scopo del progetto.

2. Orientazione agli oggetti, modularità e in particolare massima separazione sia tra il codice logico del progetto ed il codice della GUI, sia tra i valori oggetto del calcolo e il codice di gestione logico della calcolatrice.

3. Qualità ed usabilità del codice

4. Quantità e qualità delle funzionalità del progetto (inclusa la gestione della memoria)

4 Esame Orale e Registrazione Voto

La partecipazione all'esame orale è possibile solo dopo:

1. avere superato con successo (almeno 18/30) lo scritto

2. avere consegnato il progetto entro la scadenza stabilita

3. essersi iscritti alla lista Uniweb dell'esame orale

Il giorno dell'esame orale (nel luogo ed all'orario stabiliti) verrà comunicato l'esito della valutazione dei progetti. Tre esiti saranno possibili:

(A) Valutazione positiva del progetto con registrazione del voto complessivo proposto **con esenzione dell'esame orale**. Nel caso in cui il voto proposto non sia ritenuto soddisfacente dallo studente, sarà possibile richiedere l'esame orale, che potrà portare a variazioni in positivo o negativo del voto proposto.

(B) Valutazione del progetto da completarsi con un **esame orale obbligatorio**. Al termine dell'esame orale, o verrà proposto un voto complessivo sufficiente oppure si dovrà riconsegnare il progetto per un successivo esame orale.

(C) Valutazione negativa del progetto che comporta quindi la riconsegna del progetto per un successivo esame orale (il voto dell'esame scritto rimane valido).

Si ricorda inoltre che all'eventuale esame orale lo studente dovrà saper motivare **ogni** scelta progettuale e dovrà dimostrare la **piena conoscenza** di ogni parte del progetto.

5 Regole

Il presente documento va inteso come una "specifica minimale" di progetto, ossia tutto ciò che non è espressamente richiesto è lasciato a scelta libera. Il progetto dovrà essere realizzato da ogni singolo studente in modo **indipendente** da terze persone.

Relazione: Il progetto dovrà essere accompagnato da una **breve** (massimo 10 pagine in formato 10pt) relazione scritta che descriva **sinteticamente** le principali scelte progettuali. La relazione deve essere presentata come un file PDF di nome (preciso) `relazione.pdf`. La relazione deve anche specificare il sistema operativo di sviluppo e le versioni precise del compilatore e della libreria Qt.

Compilatore e libreria Qt: Il progetto deve compilare (e quindi eseguire) correttamente sulle macchine **Linux** del laboratorio informatico del plesso Paolotti con il compilatore GNU `g++` 4.x. È naturalmente possibile sviluppare il progetto su altri sistemi operativi come MacOSX/Windows. In tal caso, prima di consegnare il progetto, ricordarsi di effettuare (anche remotamente tramite `ssh -X`) una prova di compilazione, esecuzione e funzionamento sulle macchine Linux del laboratorio.

Cosa consegnare: tutti i file sorgente `.h` e `.cpp` ed il file `relazione.pdf` contenente la relazione. Se la compilazione del progetto necessita di un project file (.pro) per `qmake` diverso da quello ottenibile tramite l'invocazione di `qmake -project` allora deve anche essere consegnato un file `progetto.pro` che permetta la generazione automatica tramite `qmake` del `Makefile`.

Cosa non consegnare: codice oggetto, eseguibile, file di back-up generati automaticamente da editor o IDE e tutto quanto non necessario per la compilazione.

Come consegnare: dalle macchine del laboratorio invocando il comando

```
consegna progetto-pao-2011
```

dalla directory contenente **tutti e soli** i file da consegnare. **Non** saranno accettate altre modalità di consegna (ad esempio, via email). Naturalmente è possibile consegnare remotamente il progetto tramite il server

```
ssh.studenti.math.unipd.it
```

e opportuni comandi/programmi come `ssh`, `sftp`, `scp`, etc.

Scadenze di consegna: Il progetto dovrà essere consegnato rispettando **tassativamente** le scadenze **ufficiali** (data e ora) previste che verranno rese note tramite il sito web del docente. Approssimativamente la scadenza sarà circa 7-10 giorni prima dell'esame orale. Per i progetti ritenuti insufficienti, lo studente dovrà consegnare una nuova versione del progetto per un successivo esame orale.

Assistenza: per domande e questioni relative al progetto rivolgersi via email al docente (`tapparo@math.unipd.it`).