ESAME DI PROGRAMMAZIONE C++ (8 CFU)

L'esame deve essere svolto singolarmente e deve essere realizzato unicamente con gli strumenti utilizzati nel corso. Dato che i progetti verranno testati con essi, ogni altro strumento potrebbe far fallire, ad esempio, la compilazione e quindi l'esame. In caso di esito negativo dell'esame, lo studente dovrà presentarsi ad un successivo appello d'esame con il progetto previsto per quella sessione.

Controllate spesso il sito del corso per eventuali aggiornamenti!

Questo documento contiene DUE progetti (leggere le note evidenziate):

Progetto C++

- Creazione di un programma a riga di comando con g++, make e doxygen
- Questo progetto deve essere svolto da tutti gli studenti.

Progetto Qt

- Creazione di un programma visuale con le librerie Qt
- Questo progetto deve essere svolto anche dagli studenti dell'insegnamento di "Programmazione e Amministrazione di Sistema" iscritti a partire dall'AA 17/18.
- **Gli studenti di Programmazione e Amministrazione di Sistema degli anni precedenti al 17/18 devono CONTATTARE IL DOCENTE.**

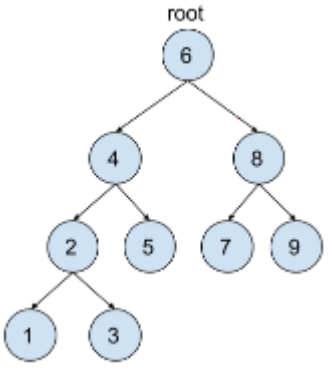
Progetto C++ del 24/07/2024

Data ultima di consegna: entro le 23.59 del 16/07/2024

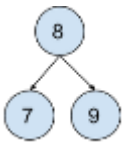
Il progetto consiste nella realizzazione di una classe generica che implementa un **albero** binario di ricerca. L'albero è formato da un insieme di elementi **T** contenuti in nodi connessi in una struttura gerarchica padre-figlio e NON deve permettere l'inserimento di dati duplicati. Deve essere possibile per l'utente scegliere liberamente la strategia usata per confrontare due dati **T**.

Oltre ai metodi fondamentali, la classe deve permettere:

1. Costruire un albero anche con un costruttore secondario che prende in input due iteratori generici di inizio e fine sequenza di dati. Lasciate al compilatore la gestione della conversione tra i tipi contenuti negli iteratori e il tipo **T** dell'albero.
2. Di conoscere il numero totale di dati inseriti nell'albero;
3. Il controllo di esistenza un elemento **T**;
4. Di accedere ai dati presenti nell'albero tramite un iteratore a sola lettura e di tipo forward. L'ordine con il quale sono ritornati i dati non è rilevante.
5. Di stampare il contenuto dell'albero usando operator<<.
6. Implementare inoltre un metodo `subtree` che, passato un dato **d** dello stesso tipo del dato contenuto nell'albero, ritorna un nuovo albero. Il nuovo albero deve corrispondere al sottoalbero avente come radice il nodo con il valore **d.** Ad esempio l'esecuzione di B=A.subtree(8) potrebbe corrispondere alla situazione in figura:



TREE A



TREE B

Implementare una funzione globale `printIF` che dato un albero binario di tipo **T**, e un predicato **P**, stampa a schermo tutti i valori contenuti nell'albero che soddisfano il predicato.

Utilizzare dove opportuno la gestione delle eccezioni. Gestite con una logica opportuna i casi limite/di errore.

Nota 1: Se non indicato diversamente, nella progettazione della classe, è vietato l'uso di librerie esterne e strutture dati container della std library come std::vector, std::list e simili. E' consentito il loro uso nel codice di test nel main.

Nota 2: A parte nullptr, non potete utilizzare altri costrutti C++11 e oltre se non indicato diversamente.

Nota 3: Nella classe, è consentito l'uso della gerarchia di eccezioni standard, delle asserzioni, la gerarchia degli stream e la funzione std::swap.

Nota 4: Per vostra sicurezza, tutti i metodi dell'interfaccia pubblica che implementate devono essere esplicitamente testati nel main anche su tipi custom. Evitate di fare dei test interattivi. Fato solo test automatici.

Nota 5: Non dimenticate di usare Valgrind per testare problemi di memoria

Nota 6: Evitate di usare "test" come nome dell'eseguibile. Potrebbe dare dei problemi sotto msys. Usate sempre il nome **main.exe**.

Alcune note sulla valutazione del Progetto C++

- Se in seguito a dei test effettuati dai docenti in fase di valutazione (es. chiamate a funzioni non testate da voi), il codice non compila, l'esame NON è superato.
- Implementazione di codice non richiesto non dà punti aggiuntivi ma se non corretto penalizza il voto finale.
- Gli errori riguardanti la gestione della memoria sono considerati GRAVI.
- La valutazione del progetto non dipende dalla quantità del codice scritto.
- NON usate funzionalità C di basso livello come memcpy, printf, FILE ecc... Se c'e' una alternativa C++ DOVETE usare quella.
- NON chiedete ai docenti se una VOSTRA scelta implementativa va bene o meno. Fa parte della valutazione del progetto.
- PRIMA DI SCRIVERE CODICE LEGGETE ACCURATAMENTE TUTTO IL TESTO DEL PROGETTO.

Progetto Qt del 24/07/2024

Data ultima di consegna: entro le 23.59 del 16/07/2024

Il progetto richiede la creazione di un sistema per la risoluzione automatica di un Sudoku. Il Sudoku è costituito da una griglia di numeri 9 x 9, e l'intera griglia è anche divisa in caselle 3 x 3.

9		6		7		4		3
			4			2		
	7			2	3		1	
5						1		
	4		2		8		6	
		3						5
	3		7				5	
		7			5			
4		5		1		7		8

Le regole per risolvere il Sudoku sono:

1. Usare cifre da 1 a 9.
2. Una cifra non può essere ripetuta in una riga, una colonna o in una casella 3 x 3.

Utilizzando un algoritmo di backtracking, risolvere il problema del Sudoku:

- Ricorsivamente cerca quali celle della griglia sono vuote;
- Quando trova una cella vuota, la riempie con una cifra nel range consentito (punto 1 delle regole) e controlla se è valida o meno (se soddisfa il punto 2 delle regole);
- Se non è valida, verifica la presenza di altri numeri.
- Se tutte le cifre nel range consentito sono state verificate e non è stata trovata alcuna cifra valida da posizionare, si torna all'opzione precedente.

Devono essere implementate quindi le seguenti funzionalità:

1. Visualizzare una griglia 9 x 9 vuota.
2. Consentire all'utente di modificare il contenuto delle celle per poter inizializzare le cifre nella griglia del Sudoku.
2. Avere un pulsante "Risolvi" per avviare l'algoritmo di risoluzione.
3. Visualizzare la griglia del Sudoku risolta o un messaggio d'errore nel caso in cui non esiste soluzione (il contenuto della griglia a questo punto non deve essere modificabile).
4. Consentire all'utente di ripercorrere *step-by-step* la risoluzione del Sudoku.
5. Avere un pulsante per resettare la griglia del Sudoku e procedere con un nuovo inserimento.

Nota 1: Utilizzare preferibilmente la versione 5.12.11 della libreria Qt (la stessa installata sulla VM).

Nota 2: Si renda il contenuto dell'applicazione adattivo rispetto alla dimensione della finestra.

Alcune note sulla valutazione del Progetto Qt

- Se in seguito a dei test effettuati dai docenti in fase di valutazione (es. chiamate a funzioni non testate da voi), il codice non compila, l'esame NON è superato.
- Implementazione di codice non richiesto non dà punti aggiuntivi ma se non corretto penalizza il voto finale.
- NON verrà valutata l'efficienza dell'applicativo sviluppato.
- Gli errori riguardanti la gestione della memoria sono considerati GRAVI.
- La valutazione del progetto non dipende dalla quantità del codice scritto.
- NON chiedete ai docenti se una VOSTRA scelta implementativa o la configurazione dell'interfaccia grafica va bene o meno. Fà parte della valutazione del progetto.
- NON chiedete ai docenti come installare QtCreator e le librerie Qt

PRIMA DI SCRIVERE CODICE LEGGETE ACCURATAMENTE TUTTO IL TESTO DEL PROGETTO.

Consegna

La consegna del/dei progetti avviene tramite la piattaforma di eLearning ed è costituita da un archivio .tar.gz **avente come nome la matricola dello studente**. L'archivio deve contenere una e solo una cartella con lo stesso nome dell'archivio (senza estensione .tar.gz). Nella root della cartella devono essere presenti:

1. Un **makefile** (per poter compilare il progetto DA RIGA DI COMANDO) che deve compilare tutto il progetto chiamato "Makefile" (attenzione alle maiuscole). Se la compilazione fallisce, il progetto non viene considerato.
2. Tutti i **sorgenti** (commentati come avete visto ad esercitazione) del progetto e organizzati a vostro piacimento.
3. Il file di **configurazione di Doxygen** per la generazione della documentazione chiamato "Doxyfile" modificato per generare documentazione HTML.
4. **Relazione in PDF** con descrizione del progetto contenente informazioni relative al design e/o analisi del progetto. La relazione serve per capire il perchè delle vostre scelte nell'implementazione o di design. Nella relazione mettere anche Nome, Cognome, Matricola ed E-Mail.
5. **Chi deve consegnare anche il "Progetto Qt", metta tutti i file sorgenti corrispondenti in una sotto-cartella "Qt".**
6. L'archivio NON deve contenere file di codice oggetto, eseguibili etc..

L'eseguibile che verrà prodotto non deve richiedere alcun intervento esterno (es. input da tastiera).

Per creare l'archivio è sufficiente lanciare il comando (di msys o console Linux):

- tar –cvzf 123456.tar.gz 123456

dove "123456" è la directory che contiene tutti i file da consegnare.

Ad esempio, una struttura dell'archivio può essere questa:

```
123456
  |--main.cpp
  |--project.h
  |--Doxyfile
  |--…
  |--Qt  (SOLO PER PROGETTO Qt)
     |—*.pro
     |--MainWindow.cpp
     |--Main.cpp
     |--MainWindow.ui
     |--…
```