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.

Progetto C++ del 14/04/2025

Data ultima di consegna: entro le 23.59 del 05/04/2025

Si richiede la progettazione e realizzazione di una coda FIFO (**Queue**) di elementi generici **T**. In una coda FIFO è possibile inserire e rimuovere un elemento alla volta. L'elemento rimosso è sempre il più vecchio inserito nella coda. **Le operazioni d'inserimento e rimozione DEVONO essere garantite dall'essere eseguite in tempo costante.**

A parte i metodi essenziali per la classe (tra cui anche conoscere il numero degli elementi inseriti) devono essere implementate le seguenti funzionalità:

1. Inserimento di un singolo elemento di tipo **T**.

2. Inserimento di un insieme di elementi presi da una sequenza identificata da due iteratori generici (di inizio e fine sequenza rispettivamente). L'ordine di inserimento determina l'anzianità dell'elemento.

3. Rimozione dell'elemento più vecchio.

4. Accesso in lettura e scrittura all'elemento più recente e all'elemento più vecchio. La sovrascrittura di un elemento non ne cambia l'anzianità.

5. La classe deve includere il supporto agli iteratori in sola lettura. Gli elementi devono essere ritornati seguendo la logica FIFO.

6. Deve essere possibile chiedere alla classe se contiene almeno un elemento di un certo valore dato.

   Scrivete una funzione globale generica **transformif** che data una coda **Q**, un predicato **P**, e un funtore **F**, modifica con il funtore i valori contenuti nella coda che soddisfano il predicato. In pratica se Qi è un elemento della coda:

   ```
   Qi=F(Qi)   se P(Qi)==true

   Qi=Qi      se P(Qi)==false
   ```

Utilizzare dove opportuno la gestione degli errori tramite asserzioni o eccezioni.

Fate inoltre attenzione a che l'operatore di assegnamento e il copy constructor mantengano l'ordine degli elementi copiati.

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. 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 14/04/2025

Data ultima di consegna: entro le 23.59 del 05/04/2025

L'obiettivo del progetto è sviluppare un editor di testo minimale con funzionalità di base per la gestione di file di testo. Funzionalità richieste:

1. Creazione di un nuovo file vale a dire implementare la funzione **Nuovo**, che consenta di creare un file vuoto;
2. Apertura e visualizzazione di un file attraverso il pulsante **Apri**, che permetta di selezionare e visualizzare il contenuto di un file di testo con estensione *.txt* o *.md*;
3. Salvataggio del file con opzioni di salvataggio (i) **Salva** che sovrascrive la versione precedente del file e (ii) **Salva con nome…** che consente di specificare un nuovo nome per il file e scegliere la destinazione di salvataggio;
4. Funzione Trova e Sostituisci costituita da un pulsante **Cerca** che apre una dialog per la ricerca e sostituzione di testo. La ricerca deve includere le seguenti opzioni:
    a. Case sensitive: distinzione tra maiuscole e minuscole;
    b. Trova parola intera: il termine cercato deve corrispondere esattamente a una parola intera nel testo;

Nota 1: Non è richiesto il supporto per espressioni regolari.

Nota 2: Utilizzare la versione 5.12.11 della libreria Qt (la stessa installata sulla VM).

Nota 3: 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. **Per il "Progetto Qt" mettete 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
     |--…
```