Programmazione

PROVA SCRITTA – 13 Aprile 2023
Parte I – LINGUAGGIO C++ e OOP

1 (12 punti)

Si descrivano i livelli di accesso nelle classi C++ e come questi cambino secondo il tipo di ereditarietà. (8 punti)
Un costruttore può essere privato? Se sì, come si invoca? Fornire un esempio di codice (4 punti)

2 (12 punti)

Si discuta l'implementazione del paradigma di programmazione generica del C++, riferendosi all'implementazione di funzioni e classi (5 punti). Indicare come strutturare il codice nei file sorgenti (2 punti) e discutere il meccanismo della specializzazione e dei requisiti impliciti (5 punti).

3 (10 punti)

Descrivere il meccanismo delle eccezioni nel C++: come si dichiarano, lanciano, prendono (anche in relazione all'ereditarietà) (8 punti). Spiegare le maggiori differenze tra uso di eccezioni e l'uso di valori di ritorno per indicare la presenza di errori a *runtime*. (2 punti)

Programmazione

PROVA SCRITTA – 13 Aprile 2023
Parte II – PROGRAMMAZIONE

4 (12 punti)

Si consideri un programma come Foto di macOS dove si gestiscono collezioni di foto. Si implementi una classe che rappresenta una foto indicando risoluzione (larghezza e altezza), luogo dove è stata scattata (può essere un dato nullo se scattata con dispositivo senza GPS), se è una foto preferita, e nomi delle persone presenti (se note, altrimenti è un elenco nullo). Si scriva una classe per gestire collezioni di foto, con funzioni per aggiungere e cancellare foto e cambiarne lo stato di preferito. Appena l'immagine viene aggiunta al programma viene associata ad un identificatore univoco che viene poi usato per le operazioni di cancellazione e modifica. La classe che gestisce la collezione ha dei metodi per rendere liste di foto preferite, che sono state scattate in un certo luogo o che contengono una certa persona.

5 (10+5 punti)

Scrivere una classe `Inventory` che rappresenti un inventario di un gioco in cui si abbiano come elementi: `Robot`, `Weapon` e `Equipment`. L'inventario può avere una dimensione massima, es. può essere richiesto che l'inventario contenga un numero massimo di elementi, specificando un parametro del costruttore. Deve essere possibile inserire un elemento in coda nell'inventario o in una certa posizione. Deve essere possibile eliminare un elemento in una certa posizione. Se si cerca di inserire un elemento in una posizione oltre il limite o cancellare o prendere un elemento in una posizione non consentita si deve lanciare l'eccezione `std::out_of_range`. L'esercizio può essere risolto con programmazione generica o usando ereditarietà (10 punti)
In caso sia necessario, implementare anche costruttore di copia e operatore di assegnazione. (5 punti)

6 (14+2 punti)

Si consideri una class `AppDownloader`, componente di una applicazione di tipo App Store che serve a scaricare e aggiornare applicazioni su uno smartphone o computer. La classe `AppDownloader` ha un metodo per aggiungere e togliere un'applicazione (rappresentata dalla class `App`, che contiene dimensione in MB e nome) alla lista delle applicazioni da scaricare o aggiornare. La classe ha un metodo per scaricare le app nell'ordine in cui sono state inserite in lista. Si considerino anche le classi `AppIcon` che rappresenta un'icona di app con i metodi `drawBadge()` che riceve un numero da stampare sovrimpresso sull'icona e `drawPercent()` che disegna una barra percentuale sotto l'icona, sulla base del numero ricevuto come argomento. Si abbia anche la classe `AppDownloaderView` che mostra quanti programmi devono essere ancora scaricati e la percentuale di scaricamento del programma che viene scaricato in quel momento. Si implementi un sistema basato sul pattern Observer per mostrare l'avanzamento dello scaricamento delle app (14 punti).
Si disegni il diagramma UML di classe (2 punti).