Esercitazione Capitolo 5:

C++

1. Classi e sottosclassi in CPP

Dichiarate una classe Studente e due sottoclassi (pubbliche) StudenteLS e StudenteIL

Dichiarate un metodo virtual (es. getCorsoStudi che stampa il nome della classe) e uno non virtual

Crea poi qualche variabile di tutti e tre e i tipi in tutte le combinazioni:

C1 a = <costruttore C2>

dove C1 = {Studente, StudenteLS, StudenteIL} {normale o puntatore}

dove C2 = {Studente, StudenteLS, StudenteIL} (usa il new)

In totale hai 3 x 2 x3 dichiarazioni di variabili. Quali di queste sono corrette e quali no?

Prova a chiamare i due metodi sopra cosa ti danno come output?

2. Classi derivate private

Dichiara una classe derivata privata e controlla che un metodo public della classe base non è più visibile.

Esempio:

classe Studente{ public ....}

classe StudenteLS : private Studentre{ }

StudenteLS s; // chiamo su questo dei metodi di Studente pur essendo public non riesco ad accedervi.

3. Template

Definiamo la funzione **max** tra due elementi generici di typname T. Cosa succede se passo qualcosa che non ha il < o > ??? Come differisce da Java.

4. STL 1

Scrivi un vettore di interi e riepilo con le prime 10 potenze di 2. Stampale in ordine e in ordine inverso. Usa gli iterator (tipo for (vector<int>::iterator i = v.begin(); i != v.end(); ++i)

5. STL 2

Costruisci l'anagrafica dei voti (interi) di una classe (con nomi unici) come hash_map come associazione

nomi->voti

L'anagrafica ha tre funzioni:

inserimento: chiedi nome e voto e inserisci il dato

elenco: stampa elenco nomi, voti

interrogazione: chiedi il nome e stampa il voto associato

6. STL e ereditarietà

Dall'esercizio 1. Costruite un std::list o std::verctor di puntatori in cui mettete oggetti di tutte tre le classi.

Chiamate il metodo virtuale e quello non virtuale.

Per scorrere la collezione usa o A.at(i) oppure meglio

Fate lo stesso con un array di elementi (non puntatori). Cosa osservi

7. Ereditarietà multipla e distruttori

Dichiarate una classe Persona, caratterizzata da codice fiscale, nome (memorizzato come stringa), cognome (memorizzato come array di caratteri) ed età. Le classi Studente e Lavoratore sono sottoclassi di Persona. Nel caso dello Studente si vuole gestire la matricola e la media. Nel caso del Lavoratore, invece, si vuole memorizzare lo stipendio annuale ed il nome della ditta presso cui lavora. La classe StudenteLavoratore eredita sia da Studente che da Lavoratore ed aggiunge la percentuale di tempo dedicato all'attività da studente.

Aggiungi un metodo stampa che, in base alla classe a cui appartiene l'oggetto, ne stampa tutti i dati, ed un metodo distruttore. Prova sia con eredità virtual che senza. Cosa cambia nei due casi?

8. Esercizio completo

Scegli un dominio di tuo interesse e scrivi un progetto C++ che però deve avere:

- almeno 5 classi e il main di prova
- qualche campo privato
- qualche costruttore che usi l'inizialization list (anche di una classe base)
- qualche distruttore – possibilmente che chiama qualche delete di campi della classe
- ereditarietà privata
- erteditarietà multipla, possibilemente virtual (diamante)
- un metodo virtual con overriding e un metodo non virtual ridefinito in una classe derivata (quindi non overriding)
- uso di STL di un contenitore (list) e di un iteratore (ad esempio ciclo for)