Corso Programmazione ad Oggetti
Corso di Laurea in Informatica
Esercitazione di Laboratorio del 01/04/2016

ESERCIZIO 1 Implementare una classe ***Contatto***, che rappresenti il concetto di contatto telefonico, la quale abbia come dati quattro stringhe rappresentanti il nome, il cognome, il numero di telefono e l'indirizzo e-mail. Usare per questi dati, la classe **string** della libreria STL. Per includerla utilizzare: **#include <string>**. Per ulteriori dettagli: http://www.cplusplus.com/reference/string/string/.

Dotare la classe dei seguenti metodi:

- **Costruttori:** uno senza parametri, uno con nome e cognome
- Metodi **get** e **set** per nome, cognome, numero telefonico e indirizzo email
- **bool operator<(const Contatto& c) const**
  bool operator>(const Contatto& c) const
  i quali, rispettivamente, confrontino due contatti in ordine alfabetico per cognome, e a parità di cognome, per nome
- **bool operator==(const Contatto& c) const** che restituisca true se due contatti sono uguali (hanno stesso valore su tutti i campi), false altrimenti
- **bool operator!=(const Contatto& c) const** che restituisca true se due contatti sono differenti (hanno almeno un valore diverso su uno dei campi), false altrimenti
- **friend istream& operator>>(istream& in, Contatto& c)** per leggere da input un contatto
- **friend ostream& operator<<(ostream& out, const Contatto& c)** per stampare su output un contatto

Realizzare quindi, un main che dopo aver creato alcuni contatti verifichi la correttezza degli operatori implementati.

ESERCIZIO 2 Implementare una classe ***NumeroComplesso***.

Un numero complesso è costituito da una coppia di numeri reali **(*a*, *b*)**, dove ***a*** rappresenta la parte reale e ***b*** la parte immaginaria (un multiplo dell'unità immaginaria, indicata con la lettera *i*). Infatti, un numero complesso viene tipicamente rappresentato come ***a+b*i***, ovvero come la somma tra la parte reale e il prodotto della parte immaginaria con l'unità immaginaria.

La somma tra due numeri complessi **(*a*, *b*)** e **(*c*, *d*)** è un numero complesso la cui parte reale è la somma tra le parti reali ***a*** e ***c***, e la parte immaginaria è pari alla somma delle parti immaginarie ***b*** e ***c***, ovvero:

$$(a, b) + (c, d) = (a + c, b + d)$$

Si noti che un numero reale non è altro che un numero complesso la cui parte immaginaria vale 0. Per cui la somma tra un numero reale e un numero complesso è definita nello stesso modo.

Il prodotto tra due numeri complessi **(*a*, *b*)** e **(*c*, *d*)** è un numero complesso la cui parte reale è la differenza tra il prodotto delle parti reali e il prodotto delle parti immaginarie, e la parte immaginaria è la somma tra il prodotto tra la parte immaginaria ***b*** e la parte reale ***c*** e il prodotto tra la parte reale ***a*** e la parte immaginaria ***d***, ovvero:

$$(a, b) * (c, d) = (a*c - b*d, b*c + a*d)$$

Tale classe dovrà dunque avere come dati due float, rappresentanti la **parte reale** e la **parte immaginaria**.

Dotare la classe dei seguenti metodi:

- **Costruttori:** uno senza parametri, uno con entrambi i parametri
- Metodi **get** e **set** per le parti immaginaria e reale
- **NumeroComplesso operator+ (const NumeroComplesso& c) const**
  NumeroComplesso operator* (const NumeroComplesso& c) const
  che implementino, rispettivamente, le operazioni di somma e prodotto
- **NumeroComplesso& operator+= (const NumeroComplesso& c)**
  NumeroComplesso& operator*= (const NumeroComplesso& c)
  che implementino, rispettivamente, le operazioni di somma e prodotto con assegnamento

- **NumeroComplesso& operator++()**
  NumeroComplesso operator++(int)
  che implementino, rispettivamente, le operazioni di pre e post incremento
- **friend istream& operator>> (istream& in, NumeroComplesso& c)** per leggere da input un numero complesso
- **friend ostream& operator<< (ostream& out, const NumeroComplesso& c)** per stampare su output un numero complesso nel formato "***a+b*i***"

Realizzare quindi, un main che dopo aver creato alcuni numeri complessi verifichi la correttezza degli operatori implementati sulla base delle formule precedentemente descritte.

Si noti che non è strettamente necessario che le classi NumeroComplesso e Contatto implementino il costruttore di copia, l'operatore di assegnamento e il distruttore. Per quale motivo?

Inoltre, si noti come il prototipo dei vari operatori presenti nelle classi differisca nel tipo di parametro restituito. Possiamo individuare due tipologie:

- **Operatori che restituiscono un oggetto** (ad esempio l'operatore + di Numero Complesso): ciò avviene perché si tratta di metodi che restituiscono oggetti temporanei (creati localmente nel metodo). La restituzione di un oggetto temporaneo per riferimento va evitata in quanto può comportare effetti indesiderati in fase di esecuzione.
- **Operatori che restituiscono un riferimento ad un oggetto** (ad esempio l'operatore += di Numero Complesso): si tratta di casi in cui non si restituiscono oggetti temporanei, e che permettono la chiamata a cascata.

Alcuni suggerimenti per lo svolgimento:

- Non utilizzare caratteri accentati per nomi di campi, variabili o metodi.
- Per il carattere tilde (~) occorre digitare: ALT + 126 su sistemi Windows, ALT gr + ì su sistemi Linux, ALT + 5 su sistemi Mac OS X.
- Per ogni esercizio realizzare una cartella contenente tutti i file creati.
- Si consiglia di utilizzare un semplice editor di testo (ad esempio, gedit), e compilare da linea di comando. Per compilare occorre passare tutti i file sorgente (.cpp) al compilatore.
- Minimizzare le inclusioni.

ESERCIZIO 3 (Compito per casa) Utilizzare la classe Contatto, per implementare un main, che crei un array di contatti e fornisca un sistema per la gestione dei contatti. Il main dovrà offrire all'utente un menu che permetta di effettuare le seguenti operazioni:

1. Inserire un contatto nell'array letto da standard input
2. Eliminare il contatto il cui nome e cognome coincidano con un nome e un cognome letti da input
3. Stampare su standard output l'array
4. Ordinare i contatti nell'array in ordine alfabetico
5. Uscire dall'applicazione

Dopo che l'utente avrà selezionato l'operazione da effettuare, indicando un numero da 1 a 5, il sistema effettuerà l'operazione richiesta, e continuerà a mostrare il menu fintanto che l'utente non digiterà 5, a quel punto l'applicazione verrà chiusa.