



Esercizio n. 1

In una applicazione in cui si richiede di gestire i dati relativi ai pezzi di ricambio per automobili si fa uso delle classi: Ricambio e VettoreRicambi. La prima classe consente di memorizzare i dati relativi ad un pezzo, mentre la seconda implementa un array dinamico di oggetti di tipo VettoreRicambi. Seguono le intestazioni delle classi:

```
class Ricambio
{
friend ostreamOperator<<(ostream&, const Ricambio&);
private:
    MyString codice;
    MyString produttore;
    MyString descrizione;
public:
    Ricambio(MyString c, MyString p, MyString d);
    Ricambio(const Ricambio&);
    bool operator==(const Ricambio&);
    MyString getCodice();
    MyString getProduttore();
    MyString getDescrizione();
    void setCodice(MyString);
    void setProduttore(MyString);
    void setDescrizione(MyString);
};

class VettoreRicambio
{
friend ostreamOperator<<(ostream&, const VettoreRicambio&);
private:
    Ricambio* dati;
    int lunghezza;
public:
    VettoreRicambio(int = 0);
    VettoreRicambio(const VettoreRicambio&);
    Ricambio& operator[](int);
    int getLunghezza();
    ~VettoreRicambio();
};
```

Si implementino **solo** i metodi di seguito elencati, supponendo già implementati i rimanenti.

- Costruttore della classe Ricambio
- Metodo setCodice() della classe Ricambio
- Operatore << della classe Ricambio
- Costruttore di copia della classe VettoreRicambio
- Operatore[] della classe VettoreRicambio
- Distruttore classe VettoreRicambio

Nota: E' possibile aggiungere, qualora lo studente lo ritenga necessario, ulteriori membri pubblici o privati alle classi. Nel caso in cui venga aggiunta una nuova funzione deve esserne fornita l'implementazione.

Esercizio 2

Siano date le classi Ricambio (definita nell'esercizio 1) e Produttore, che rappresentano rispettivamente i dati dei ricambi per automobili e delle industrie produttrici che li realizzano.

Sia l'istestazione della classe Produttore la seguente:

```
class Produttore
{
friend ostreamOperator<<(ostream&, const Produttore &);
private:
    MyString nome;
    MyString codice_produttore;
public:
    Produttore(MyString d, MyString c);
    Produttore (const Produttore &);
    bool operator==(const Produttore &);
    MyString getCodice();
    MyString getNome();
};
```

Si assume che i codici dei produttori e dei ricambi siano univoci nell'identificare industrie produttrici e parti; inoltre, si assume essere data l'implementazione delle classi Ricambio e Produttore oltre a quella delle classi ListaProduttore, ListaRicambio, IteratoreProduttore ed IteratoreRicambio (che sono definite come di consueto).

Utilizzando gli iteratori scrivere le funzioni:

- `bool prodottoOriginale(const Ricambio& ric, const ListaProduttore& lprod);`
- `ListaRicambio trovaRicambio(const MyString& descrizione, const ListaRicambio& lric, const ListaProduttore& lprod, bool soloOriginali = false);`

La prima funzione, dato un ricambio *ric* ed una lista di produttori *lprod* restituisce true se il ricambio è originale. Per semplicità assumiamo che un ricambio è originale se il nome del produttore è uno dei seguenti: Audi, BMW, Citroën, Fiat, Mercedes, Peugeot, Renault.

La seconda funzione, data una descrizione (es. "Filtro Olio") identifica i ricambi presenti nella lista *lric* aventi tale descrizione e li restituisce raccolti in una in una lista di ricambi. Si noti che nel caso in cui il valore del parametro *soloOriginali* (che per default è falso) è vero, allora la lista in output dovrà contenere solo ricambi originali.

Nota: E' possibile che ricambi equivalenti (che hanno, cioè, la stessa descrizione) siano prodotti da diversi produttori; in questo caso, i codici dei ricambi equivalenti sono distinti. Nella seconda funzione la lista *lprod* è fondamentale per determinare se un pezzo è o meno originale. È possibile utilizzare la funzione *prodottoOriginale* per realizzare la funzione *trovaRicambio*.

Esercizio 3

Data la classe Produttore descritta nell'esercizio 1, si utilizzi l'ereditarietà per progettare una estensione dal nome *ProduttoreCertificato*. Un particolare produttore è certificato nel caso in cui esso abbia ottenuto una certificazione di qualità. Si assume che una certificazione di qualità possa essere rappresentata attraverso una stringa che ne contiene una descrizione.

È richiesta l'implementazione dell'interfaccia della classe ProduttoreCertificato e di un costruttore.