Un `LegoSet` rappresenta una scatola di mattoncini Lego®. Una scatola può contenere un numero illimitato di mattoncini. Ogni mattoncino di una scatola è caratterizzato da una descrizione e un colore. La descrizione può essere di al più 30 caratteri, mentre il colore può essere nero, bianco, rosso, verde, o blu.

Implementare le seguenti operazioni che possono essere effettuate su un `LegoSet`.

--- PRIMA PARTE ---

✓ `LegoSet set;`
Costruttore di default che inizializza un `LegoSet set` vuoto, cioè senza nessun mattoncino.

✓ `cout << set;`
Operatore di uscita per il tipo `LegoSet`. I mattoncini vengono stampati in ordine di aggiunta. L'uscita ha la forma seguente:

```
quattro per quattro piatto, bianco
due per quattro alto, verde
due per quattro a scivolo, bianco
uno per quattro piatto, rosso
uno per quattro liscio, blu
```

In questo esempio, il `LegoSet` contiene 5 mattoncini. Il primo mattoncino ha descrizione "quattro per quattro piatto" e colore bianco, il secondo ha descrizione "due per quattro alto" e colore verde, e così via. Notare il carattere '`,`' e lo spazio tra la descrizione ed il colore.

✓ `set.aggiungiMattoncino(descr, colore);`

Operazione che aggiunge al `LegoSet set` un nuovo mattoncino con descrizione `descr` e colore `colore`. I colori vengono identificati con un carattere: '`n`' sta per nero, '`b`' per bianco, '`r`' per rosso, '`v`' per verde, '`l`' (elle) per blu. Se un gli input sono in qualche modo non validi, la funzione non ha effetto.

✓ `set.eliminaMattoncino(descr);`

Operazione che elimina dal `LegoSet set` un mattoncino di descrizione `descr`. Se tale mattoncino non è presente nel `LegoSet`, la funzione non ha effetto. Se invece nel `LegoSet` sono presenti più mattoncini con la stessa descrizione `descr`, allora la funzione elimina il primo in ordine di aggiunta.

✓ `set % colore;`
Operatore di resto tra un `LegoSet` e un `char`, che restituisce il numero di mattoncini del colore specificato dal carattere `colore` presenti in `set`. Il carattere `colore` identifica un colore secondo la regola solita.

--- SECONDA PARTE ---

✓ `set.eliminaMattoncino(colore);`

Operazione che elimina dal `LegoSet set` tutti i mattoncini di colore specificato dal carattere `colore`, secondo la regola solita. Se `colore` non è valido, la funzione non ha effetto.

✓ `set.aggiungiMattoncinoComune(codice, descr, colore);`
Operazione che definisce un nuovo *mattoncino comune*, e inoltre lo aggiunge al `LegoSet set`. Un mattoncino comune è un mattoncino che può essere in seguito aggiunto a *qualsiasi* `LegoSet` senza dover specificare nuovamente la sua descrizione e il suo colore (vedi dopo). Il mattoncino comune è identificato universalmente dal numero `codice` che è compreso tra 1 e 100, e ha descrizione `descr` e colore `colore`. Il carattere `colore` identifica un colore secondo la regola solita. Se un mattoncino comune con lo stesso codice è già stato definito, la funzione non ha effetto (nemmeno quello di aggiungere il mattoncino al set).

Stessa cosa se gli input sono in qualche modo non validi. Una volta definito con una certa descrizione e un certo colore, un mattoncino comune resta definito così per tutta la durata del programma.

✓ `set.aggiungiMattoncinoComune(codice);`
Operazione che aggiunge al `LegoSet set` un mattoncino comune con codice `codice`, definito in precedenza da questo o da un *qualsiasi altro* `LegoSet`, senza dover specificare nuovamente la sua descrizione e il suo colore. Se un mattoncino comune con quel codice non è stato ancora definito da *nessun* `LegoSet`, la funzione non ha effetto. Stessa cosa se gli input sono in qualche modo non validi.

✓ `~LegoSet();`
Distruttore.

Mediante il linguaggio C++, realizzare il tipo di dato astratto definito dalle precedenti specifiche. Non è permesso utilizzare funzionalità della libreria STL come il tipo `string`, il tipo `vector`, il tipo `list`, ecc.