FONDAMENTI DI PROGRAMMAZIONE **PROVA PRATICA 19 FEBBRAIO 2020**
CORSO DI LAUREA IN INGEGNERIA INFORMATICA

Un `Flotta` rappresenta una flotta di aerei da combattimento. La flotta è organizzata su più file, ognuna indicizzata da un numero intero a partire dalla fila di testa che ha indice 0. Ogni fila è formata da un diverso numero di aerei. Implementare le seguenti operazioni che possono essere effettuate su una `Flotta`:

--- Metodi invocati nella PRIMA PARTE di main.cpp: ---

✓ **`Flotta f(n);`**

Costruttore generico che inizializza una `Flotta` di `n` file, inizialmente non formate.

✓ **`f.forma_fila(i,n);`**

Funzione che forma la fila di indice `i`, e le assegna `n` aerei. Il numero di aerei può essere anche 0, in tal caso la fila viene formata vuota. Se gli input non sono validi, o se la fila è già stata formata, allora la funzione non fa nulla.

✓ **`(int)f;`**

Operatore di conversione a intero per il tipo `Flotta`, che restituisce il numero di aerei della fila di lunghezza maggiore tra quelle formate. Se ancora nessuna fila è stata formata restituisce 0.

✓ **`cout << f;`**

Operatore di uscita per oggetti di tipo `Flotta`. La flotta viene stampata secondo il seguente formato:

```
    A A A
 A A A A A A
   A A A A
A A A A A A A
? ? ? ? ? ? ?

      A
? ? ? ? ? ? ?
   A A A A
```

Ogni riga rappresenta una fila a partire da quella di indice 0, e ogni carattere '`A`' rappresenta un aereo. Le varie file devono essere disegnate in modo che la flotta sia simmetrica lungo un asse verticale. Le righe di caratteri '`?`' rappresentano file non ancora formate. In tali righe, i caratteri '`?`' sono di un numero pari al numero di aerei della fila di lunghezza maggiore. Le righe vuote rappresentano file formate ma con 0 aerei. L'esempio sopra mostra una flotta di 9 file, la prima di 3 aerei, la seconda di 6, la terza di 4, la quarta di 7, la quinta non formata, la sesta di 0, la settima di 1, l'ottava non formata, e la nona di 4.

✓ **`~Flotta();`**

Distruttore per oggetti di tipo `Flotta`.

--- Metodi invocati nella SECONDA PARTE di main.cpp: ---

✓ **`Flotta f2(f);`**

Costruttore di copia per oggetti di tipo `Flotta`.

✓ **`f1 + f2;`**

Operatore di somma tra due oggetti di tipo `Flotta`. Restituisce una `Flotta` risultante dalla concatenazione delle file di `f1` con quelle di `f2`. Le file di `f1` compariranno per prime nella flotta risultante, e le file di `f2` per seconde.

✓ **`f1 -= f2;`**

Operatore di sottrazione e assegnamento tra due oggetti di tipo `Flotta`, che simula un attacco della flotta attaccante `f2` ai danni della flotta difenditrice `f1`. Il combattimento avviene per file: la fila di indice 0 della flotta attaccante attacca quella di indice 0 della flotta difenditrice, la fila di indice 1 attacca quella di indice 1, e così via. Per ogni fila, possono verificarsi tre casi: (1) la fila difenditrice conta più aerei di quella attaccante, in questo caso la fila difenditrice resta *illesa*, cioè il suo numero di aerei resta immutato; (2) la fila difenditrice conta meno aerei di quella attaccante, in questo caso la fila difenditrice viene *distrutta*, cioè viene portata ad avere 0 aerei; (3) la fila difenditrice e quella attaccante hanno un uguale numero di aerei, in questo caso la fila difenditrice viene *semi-distrutta*, cioè viene portata ad avere la metà (approssimato per difetto) degli aerei che aveva prima del combattimento. In ogni caso, la flotta attaccante non subisce danni e resta immutata. Il combattimento può avvenire solo tra flotte con uguale numero di file e completamente formate. Se le flotte hanno un numero di file diverso, oppure una delle due flotte ha file non formate, l'operatore non ha effetto.

✓ **`f += vettore;`**

Operatore di somma e assegnamento tra un oggetto di tipo `Flotta` ed un vettore di interi, che aggiunge aerei alle file della flotta. Il vettore deve contenere tanti interi quanti sono le file della flotta. Alla fila di indice 0 viene aggiunto un numero di aerei pari all'elemento di indice 0 del vettore, alla fila di indice 1 quello di indice 1 del vettore, e così via. Se una fila non è formata, o l'elemento corrispondente nel vettore è negativo, quella fila rimane immutata.

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

USCITA CHE DEVE PRODURRE IL PROGRAMMA

```
--- PRIMA PARTE ---
Test costruttore




Test funzione forma_fila
   A A A
A A A A A A
  A A A A
? ? ? ? ? ?

Test operatore int
Larghezza: 6
Test distruttore
Distruttore chiamato
--- SECONDA PARTE ---
Test costruttore di copia
   A A A
A A A A A A
  A A A A
? ? ? ? ? ?
     A A A
  A A A A A A
    A A A A
A A A A A A A A

Test operatore +
     A A A
  A A A A A A
    A A A A
A A A A A A A A
     A A A
  A A A A A A
    A A A A
? ? ? ? ? ? ? ?

Test operatore -=
       A
     A A A
      A A
A A A A A A A A
```

Note per la consegna:

Affinché l'elaborato venga considerato valido, il programma **deve** produrre almeno la prima parte dell'output atteso. In questo caso, i docenti procederanno alla valutazione dell'elaborato **solo se** lo studente avrà completato l'autocorrezione del proprio elaborato.
In **tutti** gli altri casi (per esempio, il programma non compila, non collega, non esegue o la prima parte dell'output non coincide con quella attesa), l'elaborato è considerato **insufficiente** e, pertanto, **non verrà corretto**.
