FONDAMENTI DI INFORMATICA I **PROVA PRATICA 15 GIUGNO 2015**

CORSO DI LAUREA IN INGEGNERIA INFORMATICA

Un `VettoreCrescente` è un vettore di interi di dimensione stabilita dall'utente. Tale dimensione può essere incrementata, per permettere all'utente di scrivere elementi anche oltre la dimensione corrente del vettore (gli eventuali elementi intermedi vengono inizializzati a zero). Si supponga che la maggior parte delle letture e scritture avvenga all'interno della dimensione corrente e che l'accesso oltre tale dimensione sia *piuttosto sporadico*. Inoltre le operazioni di accesso ad un elemento all'interno della dimensione corrente *debbono essere veloci*.

Implementare il tipo di dato astratto `VettoreCrescente`, dotandolo delle seguenti operazioni:

--- PRIMA PARTE --- *(qualora siano presenti errori di compilazione, collegamento o esecuzione in questa parte,* ***l'intera prova sarà considerata insufficiente*** *e pertanto* ***non sarà corretta****)*

- ✓ **`VettoreCrescente v(dim);`**
  Costruttore che crea un vettore crescente di `dim` interi, tutti inizializzati a zero.
- ✓ **`v.set(ind,val);`**
  Funzione che imposta il valore `val` all'elemento `ind`-esimo del vettore (`ind` parte da zero). Se `ind` è minore di zero, la funzione imposta a `val` l'elemento di posizione –`ind`. Se il modulo di `ind` è maggiore o uguale alla dimensione corrente, il vettore viene esteso e la nuova dimensione sarà pari a `ind`+1, **inizializzando gli eventuali elementi intermedi a zero** (si noti come l'operazione `set` *non fallisca mai*).
  Esempio:
  Dato il vettore crescente `v` di 4 elementi 5, 7, -3, e 6, l'istruzione `v.set(5,7);` modificherebbe `v` estendendolo su 6 elementi e mettendo a zero il quinto: 5, 7, -3, 6, **0**, 7
- ✓ **`cout<<v;`**
  Operazione di uscita per il vettore crescente. Nel caso `v` abbia dimensione corrente pari a 4 e valori 5, 7, -3 e 6, l'uscita dovrà essere la seguente:
  `< 5 7 -3 6 >`

--- SECONDA PARTE ---

- ✓ **`int(v)`**

  Operazione che converte il vettore crescente in un intero. Tale intero è il numero di **minimi relativi** presente nel vettore. Un minimo relativo è un elemento del vettore che non si trovi agli estremi e che sia strettamente minore sia dell'elemento precedente che di quello successivo.
  Esempio:

  Dato il vettore crescente **`< 5 7 -3 6 0 7 >`** l'operatore restituirà 2.
- ✓ **`v.azzera();`**
  Qualora nel vettore lo stesso massimo sia raggiunto in due o più posizioni distinte, azzera tutti gli elementi fra il primo e l'ultimo massimo.

  Esempio:

  Dato `v` = **`< 5 7 -3 6 0 7 >`**, il nuovo contenuto di `v` sarà **`< 5 7 0 0 0 7 >`**.
- ✓ **`v2 = v;`**
  Operatore di assegnamento, che sostituisce il valore del vettore `v2` con il valore del vettore `v`.
- ✓ **`~VettoreCrescente();`**
  Distruttore.

Mediante il Linguaggio C++, realizzare il tipo di dato astratto **`VettoreCrescente`**, definito dalle precedenti specifiche. **Gestire le eventuali situazioni di errore**.

NOTE SULLO SVOLGIMENTO DELLA PROVA PRATICA

AVVIO E IDENTIFICAZIONE

• Avviare la macchina in modalità diskless, scegliere "Fondamenti di Informatica I"
ed effettuare il login: **nome**: studenti **password**: studenti

• Aprire un terminale e al prompt spostarsi sulla cartella 'elaborato' (`$ cd ~/elaborato`). Si utilizzi il comando `pwd` per verificare che ci si trovi nella cartella corretta `/home/studenti/elaborato`.

• Sempre al prompt dare il comando `ident`, sempre da dentro la cartella. Lo script richiede i propri dati (cognome, nome, numero di matricola e password (la password **non va dimenticata** in quanto è indispensabile per scaricare da internet il proprio elaborato a consegna avvenuta). Il comando `ident` crea il file *matricola.txt* nella cartella corrente. Lo script può essere lanciato più volte, in tal caso il file *matricola.txt* viene sovrascritto. Per verificare che il file sia stato creato e che il contenuto sia quello giusto dare il comando (la password è codificata):

```
$ cat /home/studenti/elaborato/matricola.txt
```

• A questo punto il docente verifica che tutti gli studenti abbiano effettuato l'identificazione, dopodiché provvede a inviare i seguenti file nella cartella `elaborato` del proprio PC: *compito.h, compito.cpp, main.cpp.*
Controllare pertanto che questi file, insieme al file *matricola.txt*, siano presenti sul proprio elaboratore.

SVOLGIMENTO DELLA PROVA

• Definire ed implementare il tipo di dato astratto richiesto e le relative funzioni nei file *compito.h* e *compito.cpp*. Il file *main.cpp* contiene la funzione principale `main()` ed è utilizzato dallo studente per testare la sua implementazione della classe. Il file *main.cpp* può essere modificato a piacere. In sede di valutazione dell'elaborato verrà considerato **esclusivamente il contenuto dei file compito.h e compito.cpp** ed è pertanto **vietato cambiare nome a tali file**.
Per compilare e linkare dare il comando:

`$ g++ main.cpp compito.cpp` (eseguibile invocabile tramite `$ ./a.out`)
(utilizzare `g++ -g` per includere le informazioni di debug qualora si intenda debuggare con `ddd`).

PER CONSEGNARE O RITIRARSI

Recarsi dal docente **dopo aver preso nota dell'identificativo della macchina** (esempi: g34, s23, c22, …).

USCITA CHE DEVE PRODURRE IL PROGRAMMA

```
Test del costruttore. Deve stampare: < 0 0 0 0 >
< 0 0 0 0 >


Test della set. Deve stampare: < 2 7 -3 6 >
< 2 7 -3 6 >


Test della set oltre l'attuale dimensione. Deve stampare: < 2 7 -3 6 0 7 >
< 5 7 -3 6 0 7 >

Conto il numero di minimi relativi. Deve stampare: 2
2

Testa della azzera. Deve stampare: Deve stampare: < 5 7 0 0 0 7 >
< 5 7 0 0 0 7 >


Test dell'operatore di assegnamento. Deve stampare: < 5 7 0 0 0 7 >
< 5 7 0 0 0 7 >


Test del distruttore (v2 sta per essere distrutto)
```