**PROVA PRATICA 15 GENNAIO 2014**

**FONDAMENTI DI INFORMATICA I**
**FOND. DI INFORMATICA E PROGRAMMAZIONE A OGGETTI**

Molte applicazioni ingegneristiche fanno uso di vettori **di interi** di grandi dimensioni in cui la maggior parte degli elementi è nulla e dunque solo una piccola parte di essi è diversa da zero. In questi casi, per risparmiare memoria, è conveniente memorizzare solo le coppie (*valore,indice*) corrispondenti ai soli valori diversi da zero (tali coppie a valore non nullo sono dette *entrate*). Un vettore con queste caratteristiche è chiamato *vettore sparso* (i vettori convenzionali sono infatti detti anche *vettori densi*). Ad esempio di vettore denso **< 0 -2 0 4 0 8 0 >** è rappresentabile come vettore sparso utilizzando le entrate **(-2,1)**, **(4,3)** e **(8,5)**.

Si implementi il tipo di dato astratto `VettoreSparso`, fornendo le seguenti funzionalità:

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

- ✓ **`VettoreSparso s(d);`**
  Costruttore che crea un `VettoreSparso s` di dimensione `d`. Inizialmente `s` è privo di entrate.
- ✓ **`s.set(v,i);`**
  Operazione che aggiunge l'entrata (`v`,`i`) al vettore sparso `s`. In particolare, l'operazione deve lasciare `s` invariato:
  - nel caso in cui l'indice `i` sia fuori dall'intervallo ammissibile, oppure
  - nel caso in cui `v` sia pari a zero (*sarebbe sbagliato aggiungere un'entrata a valore nullo, in quanto la non presenza di una entrata con quello stesso indice indica già che quell'elemento del vettore vale zero*).

  Nel caso in cui sia `i` che `v` siano validi, l'operazione:
  - se è già presente una entrata con indice `i`, si limita ad aggiornarne il campo valore a `v`;
  - altrimenti, aggiunge alla lista la nuova entrata (`v`,`i`).
- ✓ **`cout<<s;`**
  Operatore di uscita per il tipo `VettoreSparso`. L'uscita ha la seguente forma:

  ```
  [7]{(-2,1)(4,3)(8,5)}
  ```

  dove viene riportata prima la dimensione del vettore fra parentesi quadre e poi la lista delle entrate, ***ordinate per indice crescente***, fra parentesi graffe. In questo caso `s` è un vettore sparso di 7 elementi, contenente solo tre elementi non nulli, agli indici 1, 3 e 5, i cui valori sono, rispettivamente, -2, 4 e 8. Se il vettore fosse stato tutto nullo (nessuna entrata), sarebbe stato visualizzato così: `[7]{}` (senza spazio fra graffe). L'operatore ***non*** aggiunge il *new line* alla fine.

**SECONDA PARTE** ***(si invita a mettere sotto commento le operazioni di questa seconda parte che dovessero impedire la compilazione, il collegamento o la corretta esecuzione del codice)***

- ✓ **`s.visualizzaComeDenso();`**
  Operazione che mostra a video il vettore sparso `s` come viene di solito visualizzato un vettore denso, ossia come la sequenza dei suoi elementi (**separati da uno spazio** in questo caso) e fra parentesi angolari:

  ```
  < 0 -2 0 4 0 8 0 >
  ```

- ✓ **`s.reset(i);`**

  Operazione che elimina, se presente, l'entrata d'indice `i` dalla lista. Altrimenti lascia il vettore sparso `s` inalterato.

- ✓ **`s*=k;`**

  Operazione che moltiplica il vettore sparso per l'intero `k`. Se `k` vale 0 vanno rimosse tutte le entrate in `s`. Altrimenti `s` va modificato in modo tale che ciascuna entrata abbia valore pari al vecchio valore moltiplicato per `k`.

- ✓ **`~VettoreSparso();`**
  Distruttore.

Mediante il Linguaggio C++, realizzare il tipo di dato astratto `VettoreSparso`, definito dalle precedenti specifiche. Individuare le eventuali situazioni di errore e metterne in opera un corretto trattamento.

## 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 spostarsi sulla cartella 'elaborato' (`$ cd ~/elaborato`). Si utilizzi il comando `pwd` per verificare che ci si trovi nella cartella corretta `/home/studenti/elaborato`.

• 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 abbiamo 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 avendo preso nota dell'identificativo della macchina (g34, s23, …).