

# Algoritmi e Strutture Dati – Prova di Laboratorio

18/07/2019

## Istruzioni

Risolvere il seguente esercizio implementando un programma in un singolo file .cpp, completo di funzione *main*. Si presti particolare attenzione alla formattazione dell'input e dell'output, e alla complessità target indicata per ciascuna funzionalità. Nel caso la complessità target non sia specificata, si richiede che sia la migliore possibile. La lettura dell'input e la scrittura dell'output **DEVONO** essere effettuate tramite gli stream **cin** e **cout** rispettivamente. La correzione avverrà prima in maniera automatica inviando il file .cpp al server indicato in aula. Quest'ultimo esegue dei test confrontando l'output prodotto dalla vostra soluzione con l'output atteso. In caso la verifica abbia esito positivo sarà possibile consegnare il compito, il quale verrà valutato dai docenti in termini di complessità. Si ricorda che è possibile testare la correttezza del vostro programma in locale su un sottoinsieme dei input/output utilizzati nella seguente maniera. I file di input e output per i test sono nominati secondo lo schema: **input0.txt output0.txt input1.txt output1.txt ...** Per effettuare le vostre prove potete utilizzare il comando del terminale per la redirectione dell'input. Ad esempio

```
./compilato < input0.txt
```

effettua il test del vostro codice sui dati contenuti nel primo file di input, assumendo che **compilato** contenga la compilazione della vostra soluzione e che si trovi nella vostra home directory. Dovete aspettarvi che l'output coincida con quello contenuto nel file **output0.txt**. Per effettuare un controllo automatico sul primo file input **input0.txt** potete eseguire la sequenza di comandi

```
./compilato < input0.txt | diff - output0.txt
```

Questa esegue la vostra soluzione e controlla le differenze fra l'output prodotto e quello corretto.

## Esercizio

Si consideri un sistema di memorizzazione dei dati relativi a dei veicoli. Ogni veicolo è rappresentato da un valore *targa* univoco, intero e positivo, e da un valore intero *categoria* compreso tra 0 e  $C - 1$ . Il sistema legge i dati relativi a  $N$  veicoli e li inserisce dentro una *tabella hash*, utilizzando la *targa* come etichetta. La tabella hash è realizzata con il metodo di concatenazione.

Per ogni indice  $i$  della tabella hash si definisce  $M(i)$  come la categoria con più veicoli in  $i$ , e  $V(i)$  come il numero di veicoli in  $i$  appartenenti a  $M(i)$ . A parità di numero di veicoli, si considerino le *categorie* in ordine crescente. Scrivere un programma che:

- legga da tastiera una sequenza di  $N$  coppie di interi  $\{targa, categoria\}$  e le inserisca nella tabella hash all'indirizzo dato dalla seguente funzione hash:

$$h(x) = \{[(a \times x) + b] \% p\} \% (C)$$

dove  $p=999149$ ,  $a=1000$  e  $b=2000$ ;

- Stampi a video i primi  $K$  indirizzi  $i$  della tabella hash in ordine di  $V(i)$  decrescente. A parità di  $V(i)$ , si considerino gli indirizzi in ordine crescente.

L'**input** è formattato nel seguente modo: la prima riga contiene gli interi  $N$ ,  $K$  e  $C$  separati da uno spazio. Seguono  $N$  righe contenenti una coppia di interi ciascuna.

L'**output** contiene gli elementi della soluzione, uno per riga.

## Esempio

Input	Output
11 6 7	0
170 1	3
96 2	1
577 0	2
692 0	4
873 6	6
446 6	
732 1	
394 1	
845 1	
715 1	
456 3	