Laboratorio di Algoritmi
e strutture dati
Esercizio 3

Giorgio Mecca
Mat: 880847

```
giorgio@giorgio-VirtualBox:~/Scrivania/laboratorio-algoritmi-2019-20/Esercizio3$ make
cc -g -O0 -I include -I Resources/C/Unity/ -c src/hashMap.c -o hashMap.o
cc -g -O0 -I include -I Resources/C/Unity/ application/map_app.c hashMap.o -o app
giorgio@giorgio-VirtualBox:~/Scrivania/laboratorio-algoritmi-2019-20/Esercizio3$ make run
cc -g -O0 -I include -I Resources/C/Unity/ -c src/hashMap.c -o hashMap.o
cc -g -O0 -I include -I Resources/C/Unity/ application/map_app.c hashMap.o -o app
./app hashes.csv
Caricamento dati su HashMap
        Tempo impiegato :       3.842402 secondi

Caricamento dati su Array statico
        Tempo impiegato :       4.535678 secondi

Recupero dati su HashMap
        Tempo impiegato :       6.219228 secondi

Recupero dati su array statico
        Tempo impiegato :       14.923388 secondi

valori reperiti: HasHMap        Array
                6321031         6321031
Recupero avvenuto con SUCCESSO
giorgio@giorgio-VirtualBox:~/Scrivania/laboratorio-algoritmi-2019-20/Esercizio3$
```

Il terzo esercizio ci proponeva di scrivere in c delle funzioni per la creazione ed uso si una hash Map. Questa è stata creata in modo generico e per far ciò alla creazione vuole come uno dei parametri una funzione di comparazione di elementi. Dopo aver creato l' Hash map abbiamo comparato i tempi di caricamento e utilizzo di dati in in una hash map con quelli ottenuti utilizzando un array statico. L'hash map creata usa una struttura con trabocco, questo vuol dire che la funzione di hash usata influisce molto sui tempi, infatti utilizzando una funzione in modulo 1'000'000 si ottengono tempi inferiori rispetto all'array, mentre utilizzando una funzione in modulo 1'000 si ottengono tempi eccessivi poiché si vanifica l'uso di una hash map.