

Dipartimento Di Informatica

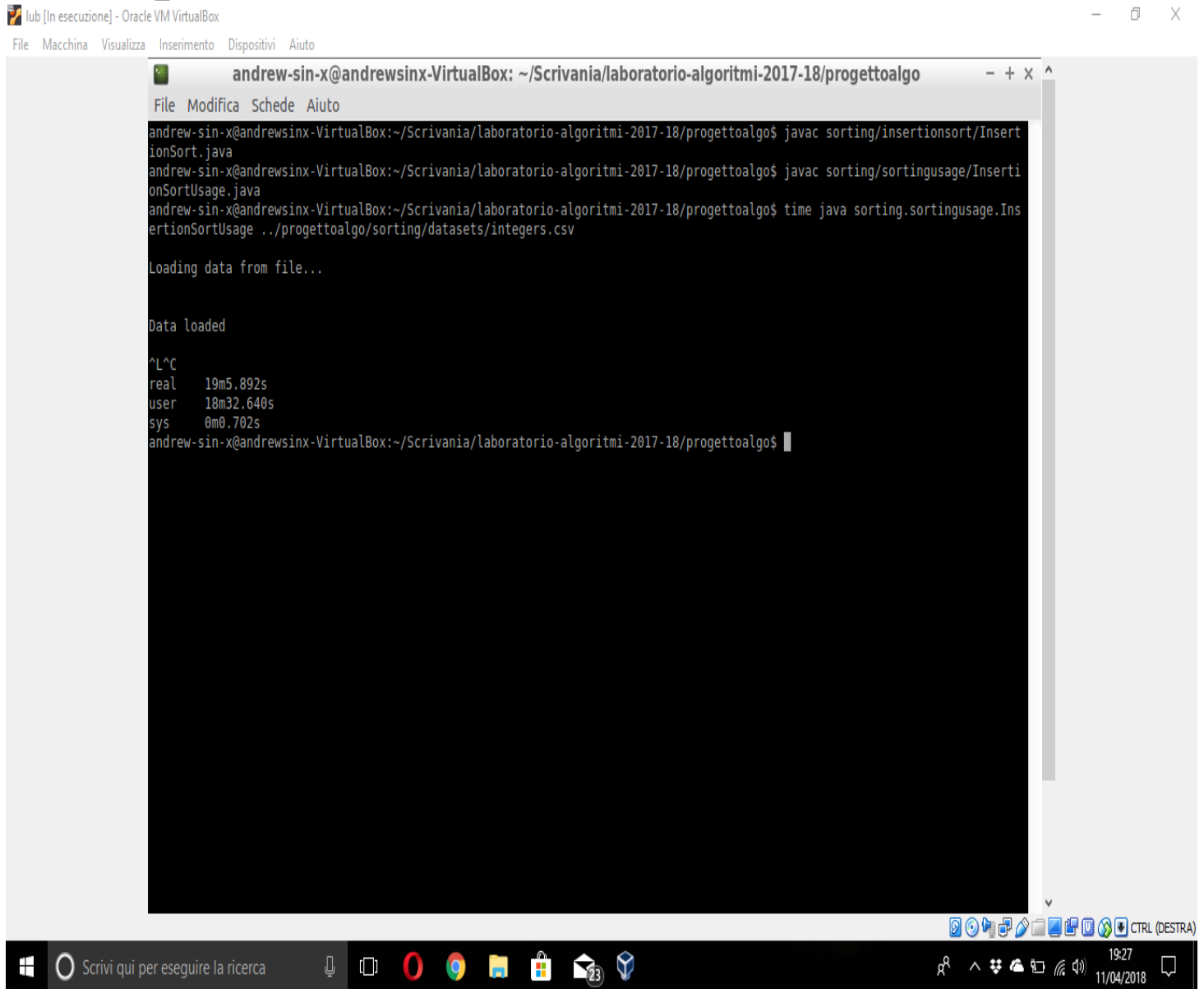
Algoritmi Di Ordinamento



*Relazione Sulla
Sperimentazione Dei Tempi
Impiegati Dagli Algoritmi.
Studente : Senese Andrea
Matricola : 811634*

Esercizio 1 : Algoritmi Di Ordinamento

1) Insertion Sort : Durante i testing del seguente algoritmo si è dedotto il seguente tempo di Esecuzione :



```
andrew-sin-x@andrewsinx-VirtualBox: ~/Scrivania/laboratorio-algoritmi-2017-18/progettoalgo
File Modifica Schede Aiuto
andrew-sin-x@andrewsinx-VirtualBox:~/Scrivania/laboratorio-algoritmi-2017-18/progettoalgo$ javac sorting/insertionsort/InsertionSort.java
andrew-sin-x@andrewsinx-VirtualBox:~/Scrivania/laboratorio-algoritmi-2017-18/progettoalgo$ javac sorting/sortingusage/InsertionSortUsage.java
andrew-sin-x@andrewsinx-VirtualBox:~/Scrivania/laboratorio-algoritmi-2017-18/progettoalgo$ time java sorting.sortingusage.InsertionSortUsage ../progettoalgo/sorting/datasets/integers.csv

Loading data from file...

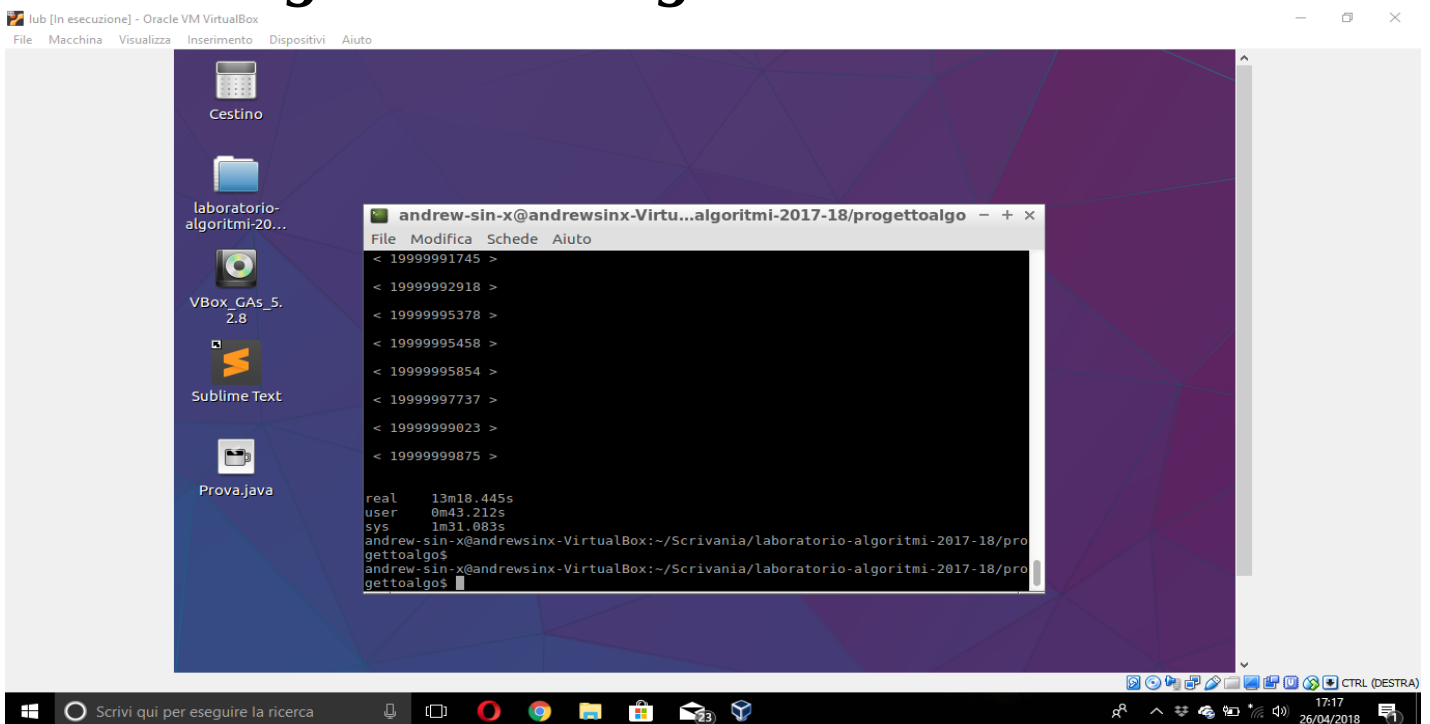
Data loaded

^L^C
real    19m5.892s
user    18m32.640s
sys     0m0.702s
andrew-sin-x@andrewsinx-VirtualBox:~/Scrivania/laboratorio-algoritmi-2017-18/progettoalgo$
```

Il Seguento Algoritmo è inefficiente, dopo ben 19 minuti ancora è arrivato al termine. Si è dedotto che l'intuitività che sfrutta l'algoritmo è abbastanza inefficiente, in

quanto si potrebbe ridurre il lavoro con qualcosa di più intuitivo.

1) MergeSort : *Durante i testing si è dedotto che l'algoritmo è accettabile, in quanto termina in tempi abbastanza ragionevoli(dipendentemente da come sia organizzato il file di record), ovviamente nel caso peggiore impiega un tempo minore. Si è dedotto dal testing che l'intuitività dell'algoritmo riguardo al criterio di ordinamento è abbastanza efficiente, ma esistono criteri differenti sicuramente migliori. Il Tempo di esecuzione è riportato nella seguente immagine :*



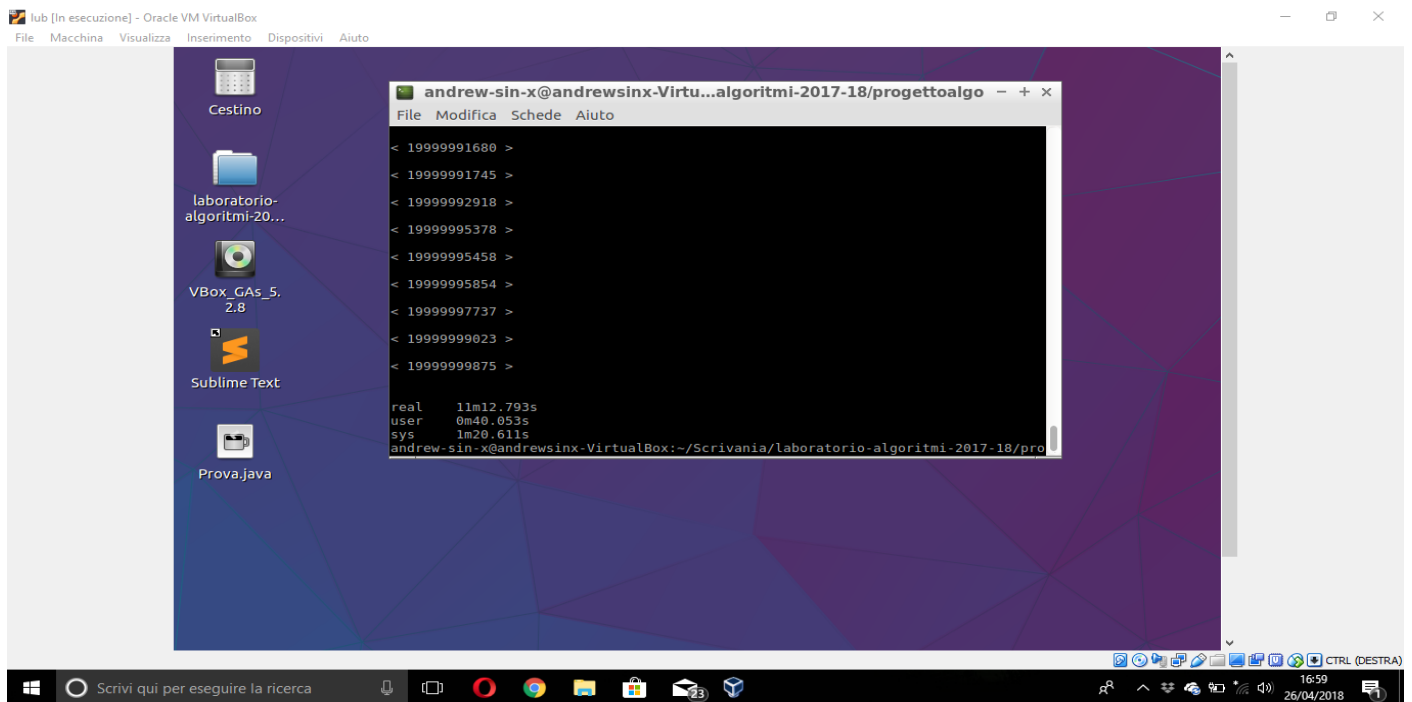
The screenshot shows a Windows desktop environment. On the left, there is a sidebar with icons for 'Cestino', 'laboratorio-algoritmi-20...', 'VBox_GAs_5.2.8', 'Sublime Text', and 'Prova.java'. The main area displays a terminal window titled 'andrew-sin-x@andrewsinx-Virtu...algoritmi-2017-18/progettoalgo'. The terminal output shows a series of numbers being processed by the MergeSort algorithm, followed by execution statistics:

```
< 19999991745 >
< 19999992918 >
< 19999995378 >
< 19999995458 >
< 19999995854 >
< 19999997737 >
< 19999999023 >
< 19999999875 >

real    13m18.445s
user    0m43.212s
sys     1m31.083s
andrew-sin-x@andrewsinx-VirtualBox:~/Scrivania/laboratorio-algoritmi-2017-18/progettoalgo$
andrew-sin-x@andrewsinx-VirtualBox:~/Scrivania/laboratorio-algoritmi-2017-18/progettoalgo$
```

The taskbar at the bottom shows the Windows Start button, a search bar, and several application icons. The system clock indicates the time is 17:17 on 26/04/2018.

3) QuickSort: Anche se non richiesto espressamente, si è voluto testare sperimentalmente anche questo algoritmo, il criterio intuitivo di ordinamento che sfrutta è davvero ottimo, in quanto lo svantaggio riguarda proprio l'organizzazione dei record nel file e nella scelta del pivot, il miglior caso riportato è bilanciando bene le due partizioni (di dimensione non necessariamente uguale). Si è dedotto che il criterio intuitivo sfruttato dall'algoritmo non è male, ma il problema sorge dalla scelta del pivot fatta dall'utente (il miglior modo per agire è proprio di ragionare sulla propria scelta, ossia l'algoritmo è un beneficio quando si fa una scelta ragionevole). Il Tempo di esecuzione è riportato nella seguente immagine :

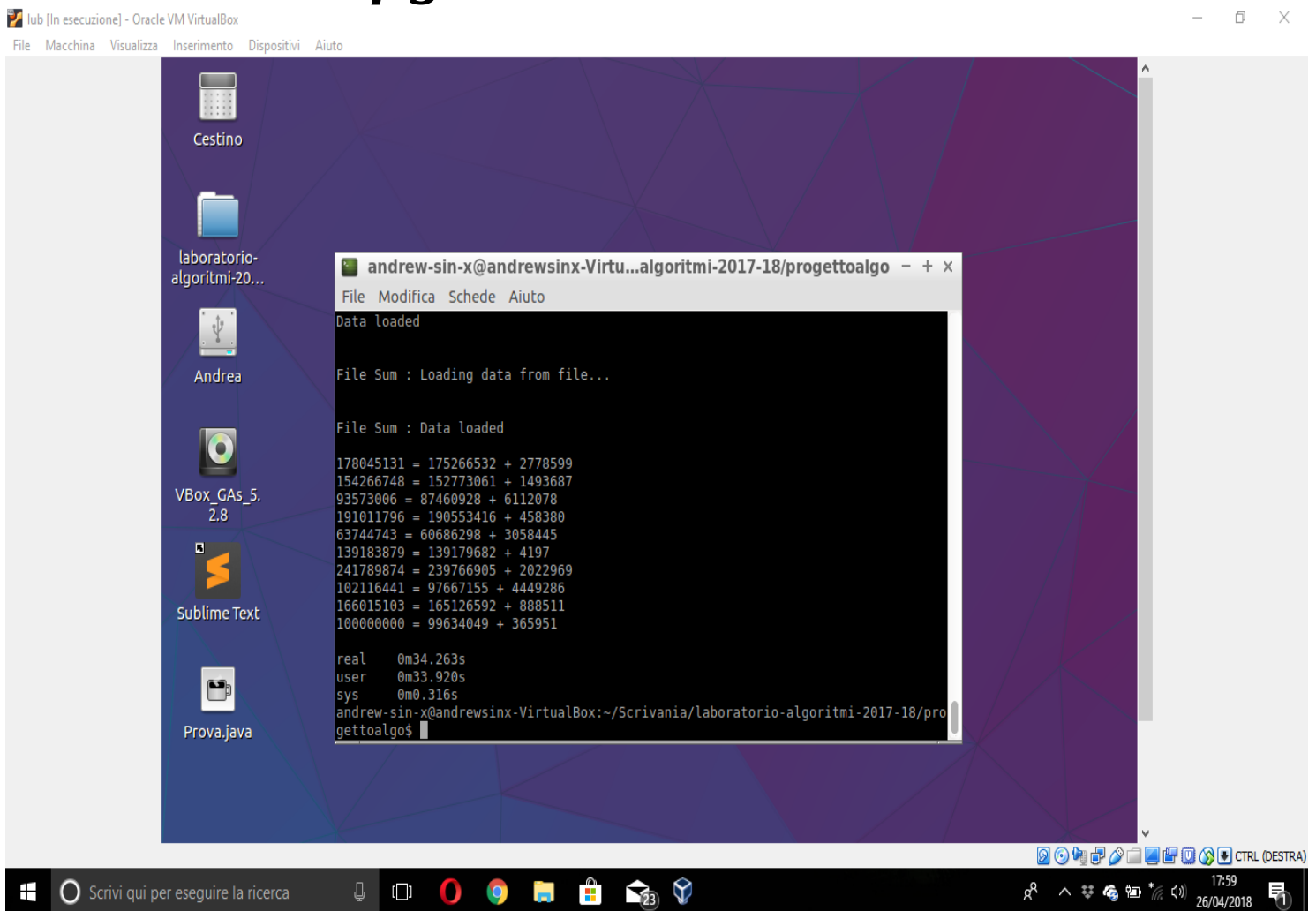


Esercizio 1.1 : SUMINTEGERS

Come richiesto dal testo, si è scelto di implementare questo algoritmo di complessità $O(n \log n)$ sfruttando l'algoritmo della ricerca binaria in quanto fa in modo di restituirne la complessità richiesta, se adottate le seguenti precauzioni : La scelta è stata di salvarsi temporaneamente man mano i valori che si trovano nell'array Sum, il membro conseguente della somma verrà determinato tramite una piccola operazione(sottrazione con il dato

parametro del metodo per determinarne la differenza, ossia l'intero da ricercare da inserire come conseguente della somma) e dato come parametro alla ricerca binaria che restituirà il seguente valore in tempo $O(\log n)$. Quindi il caso peggiore dell'algoritmo per trovare un conseguente è che venga visitato l'array intero ossia $O(n)$ e la ricerca dicotomica ne restituisce un intero opportuno(ricercato sempre all'interno dell'array) che sommato con l'intero all'intero dell'array dia l'elemento del file "sums.txt" . Prima di effettuare il seguente procedimento si ha bisogno di ordinare il seguente array perche è preferibile avere l'array parzialmente ordinato per ridurre il numero di confronti e quindi si è scelto di ordinare tramite MergeSort di complessità $O(n \log n)$ in quanto in qualsiasi caso verrà restituita una complessità di $O(n \log n)$

perche la sua complessità nel caso peggiore è $O(n \log n)$ quindi $O(n \log n) + O(n) * O(\log n) = O(n \log n) + O(n \log n) = O(n \log n)$. Il tempo di esecuzione del seguente algoritmo è molto breve come si illustra in figura:



```
andrew-sin-x@andrewsinx-Virtu...algoritmi-2017-18/progettoalgo - + x
File Modifica Schede Aiuto
Data loaded
File Sum : Loading data from file...
File Sum : Data loaded
178045131 = 175266532 + 2778599
154266748 = 152773061 + 1493687
93573006 = 87460928 + 6112078
191011796 = 190553416 + 458380
63744743 = 60686298 + 3058445
139183879 = 139179682 + 4197
241789874 = 239766905 + 2022969
102116441 = 97667155 + 4449286
166015103 = 165126592 + 888511
100000000 = 99634049 + 365951
real    0m34.263s
user    0m33.920s
sys     0m0.316s
andrew-sin-x@andrewsinx-VirtualBox:~/Scrivania/laboratorio-algoritmi-2017-18/progettoalgo$
```