*Studio Amico*

## Sommario

[Descrizione del problema 2](#_Toc86036505)

[File di input 2](#_Toc86036506)

[Assunzioni 2](#_Toc86036507)

[Assegnazione del punteggio 2](#_Toc86036508)

[Descrizione dell’algoritmo tramite pseudo-codice 3](#_Toc86036509)

[Analisi di complessità dell’algoritmo nel caso pessimo 4](#_Toc86036510)

[Screenshot della piattaforma di valutazione con i test passati o falliti 5](#_Toc86036511)

[Descrizione delle procedure di test effettuate 5](#_Toc86036512)

[Lista di problemi noti 5](#_Toc86036513)

[Sitografia: 5](#_Toc86036514)

Descrizione del problema  
  
Il problema consiste nel voler abbinare degli studenti di seconda con degli studenti di quinta, affinché questi ultimi possano dar loro ripetizioni di informatica. Il problema è risolto se ogni studente di quinta viene abbinato con uno di seconda. L'abbinamento è possibile se lo studente di quinta ha ricevuto un voto maggiore rispetto a quello di seconda.

Il compito dell'algoritmo è di ritornare un booleano che indica se l'abbinamento di tutti gli studenti è possibile (true) oppure no (false).

### File di input

Il file di input contiene tre righe:

* La prima riga contiene il numero di studenti considerati.
* La seconda riga contiene i voti presi da ogni studente di seconda (1-10).
* Mentre l'ultima riga contiene i voti di ogni studente di quinta (1-10).

### Assunzioni

Le assunzioni di questo progetto sono:

* Il numero di input può variare da 1 a 10 000 000.
* I voti degli studenti sono numeri interi tra 1 e 10.
* L’associazione tra gli studenti è uno-a-uno, ciò significa che ad ogni studente di seconda viene associato uno e un solo studenti di quinta.

### Assegnazione del punteggio

La valutazione del programma viene eseguito con 4 subtask:

* Subtask 1 [0 punti]: Casi d’esempio.
* Subtask 2 [30 punti]: N ≤ 10.
* Subtask 3 [30 punti]: N ≤ 100 000.
* Subtask 4 [40 punti]: Nessuna limitazione specifica.

## Descrizione dell’algoritmo tramite pseudo-codice

bool Associabili (N, v2, v5)

{

//Vengono creati due array di dimensione 10 che permettono di memorizzare le frequenze dei voti delle due classi.

int voti2[10]

int voti5[10]

//I due array di frequenze vengono riempiti in base ai due array di input.

for (i in 0…N) {

voti2[v2[i]-1]++

voti5[v5[i]-1]++

}

//Viene controllato che non esiste uno studente di seconda che abbia preso 10 e che non esiste uno studente di quinta che abbia perso 1. (Altrimenti non sarebbe possibile associare tutti gli studenti di seconda con gli studenti di quinta)

if (voti2[9] > 0 or voti5[0] > 0)

return false

//Grazie a questo loop possiamo verificare che la somma delle frequenze di un voto degli studenti di seconda sia minori o uguali del voto+1 degli studenti di quinta.

for (i in 8…0) {

voti2[i] += voti2[i+1]

voti5[i] += voti5[i+1]

if (voti5[i+1] < voti2[i])

return false

}

return true

}

## Analisi di complessità dell’algoritmo nel caso pessimo

Graphical user interface, text, application

Description automatically generatedLa complessità dell’algoritmo la possiamo valutare analizzando i due cicli che lo compongono:

Il primo ciclo esegue il calcolo delle frequenze dei vari voti per ogni classe, esso viene eseguito N volte.

Graphical user interface, text, application

Description automatically generated

Il secondo, invece, verifica la disponibilità degli studenti di quinta per i compagni di seconda, calcolando quanti studenti hanno raggiunto il voto in analisi (ogni iterazione corrisponde a un voto che parte da 8 poiché gli studenti di quinta devono avere il voto maggiore di un'unità, quindi, partono da i+1. Il ciclo, dunque, viene eseguito 9 volte indipendentemente dall’input.

Da queste informazioni possiamo calcolare la complessità, considerando che tutte le altre istruzioni hanno complessità costante. Non influiscono sulla complessità totale, possiamo concludere che l’algoritmo nel caso è O(n).

## Screenshot della piattaforma di valutazione con i test passati o falliti

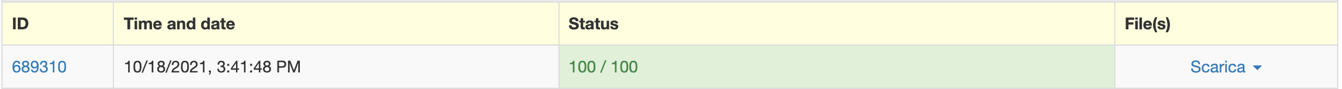


Figura Puntaggio del test effettuato

Graphical user interface, table, Excel

Description automatically generatedGraphical user interface, application, table, Excel

Description automatically generated

Figura Risultati dei test per ogni subtask

Come si può vedere dalle immagini, il programma supera tutti i vari test.

Inizialmente è stato riscontrato un problema nell’avvio del sistema di compilazione. In seguito alla consultazione dei messaggi di errore si è arrivati alla conclusione che erano presenti due funzioni main nell’applicazione, procedura non consentita.

La soluzione è stata quella di inserire esclusivamente il codice della funzione risolutiva.

## Descrizione delle procedure di test effettuate

Sono stati utilizzati i test presenti sulla piattaforma per collaudare il codice, in base alle richieste del problema. Durante lo sviluppo sono stati effettuati piccoli test in locale, attraverso la creazione di file di input appositi.

## Lista di problemi noti

Come già citato nella descrizione degli screenshot dei test, l’unico problema che è stato riscontrato è che nella piattaforma di test era necessario inserire esclusivamente la funzione risolutiva del problema.

## Sitografia:

<https://training.olinfo.it/#/task/abc_studioamico/statement>, 25.10.2021, Studio Amico