# CENTRI VACCINALI

Manuale Tecnico


UNIVERSITÀ DEGLI STUDI
DELL'INSUBRIA

Università degli Studi dell'Insubria – Laurea Triennale in Informatica

Progetto Laboratorio A: Centri Vaccinali

Sviluppato da: Lorenzo DeBlasi, matricola 745124

Claudio Ricci, matricola 747555

# Sommario

# Introduzione

Il progetto Centri Vaccinali è stato sviluppato per Laboratorio A all'interno del corso di laurea in Informatica presso l'Università degli Studi dell'Insubria.

Il progetto è stato sviluppato in Java 12, sviluppato e testato sul sistema operativo Windows 10.

In particolare, l'applicazione si occupa di salvare o leggere tutte le informazioni e i dati riguardanti centri vaccinali e cittadini vaccinati da file di testo (formato "*.txt").

# Struttura generale del sistema di classi

Il progetto è strutturato in due aree, l'area dedicata agli operatori dei centri vaccinali, Centri Vaccinali e l'area dedicata a tutti i cittadini, Cittadini.

## Classi utilizzate

### Centri vaccinali (main)

Classe contenente main

### Cittadini

Classe creata per poter creare oggetti di tipo "cittadini", utilizzati all'interno del main quando si vuole aggiungere un nuovo cittadino a un centro vaccinale.

# Classi per la lettura/scritture su file di testo

### FileInputStream

È una classe del package java.lang molto utile per leggere dati da un file in forma di una sequenza di byte.

es.
FileInputStream(File file)
crea un file stream di input per poter leggere dal file specificato come parametro.

### Scanner

Scanner è una classe del package java.util usata per ottenere in input dei tipi primitivi come, int, double, strings, etc. È il metodo più semplice per leggere degli input in un programma Java.
All'interno del programma viene utilizzata sia per leggere input inseriti dall'user del programma che per leggere automaticamente file di testo.
Per leggere input dei tipi primitivi:

es.

```
Scanner sc = new Scanner(System.in);
sc.nextLine();  //legge ciò che inserisce l'utente
```

Per creare un oggetto della classe Scanner che legge in input un file di testo basta, al momento della creazione dell'oggetto, passare come parametro un oggetto FileInputStream avente come parametro l'url del file.

es.

Scanner sc = new  Scanner(new FileInputStream("url"));

In questo caso vengono utilizzati molto spesso 2 metodi di tale classe, "hasNextLine()", controlla che all'interno del file preso in input esista un'altra linea di testo, successiva a quella appena letta. E "nextLine()" che va a leggere il contenuto della prossima linea.

### BufferedWriter

Per la scrittura su file di testo è stata usata la classe BufferedWriter che fornisce un meccanismo di bufferizzazione che rende più efficienti le operazioni di scrittura.
Tale classe ha un costruttore che riceve semplicemente un argomento di tipo Writer (nel nostro caso FileWriter) e crea un collegamento tra la sorgente e la destinazione.
Per la scrittura di una stringa su un file utilizza il metodo "write()" e successivamente il metodo "newLine()" che permette di andare a scrivere sulla linea successiva la prossima volta che si scrive.
Infine, gli stream, una volta finite le operazioni di scrittura vanno chiusi, questo è fattibile tramite il metodo "close()".

### FileWriter

Classe appartenente al package java.io è stata usata per scrivere dati, in forma di caratteri, all'interno di un file. Se il file su cui bisogna andare a scrivere non esiste ancora questa classe lo va a creare.

es.

BufferedWriter bw = new BufferedWriter(new FileWriter(file));

## File utilizzati dal programma

- "centrivaccinali.txt"
- "cittadiniregistrati.txt"
- "vaccinati_nomeCentro.txt"
- "valoriAltro_nomeCentro.txt"
- "valoriCrisiI_nomeCentro.txt"
- "valoriDolori_nomeCentro.txt"
- "valoriFebbre_nomeCentro.txt"
- "valoriLinfoadenopatia_nomeCentro.txt"
- "valoriMdT_nomeCentro.txt"
- "valoriTachicardia_nomeCentro.txt"

Viene presentato ora l'uso dei file ("*.txt") usati dal programma.

### Centrivaccinali.txt

Questo file di testo contiene un elenco di tutti i centri vaccinali registrati all'interno del sistema, ogni linea del file contiene un solo centro vaccinale e le relative informazioni, quali, nome del centro vaccinale, indirizzo

del centro (qualificatore via/v.le/piazza, nome, numero civico, comune, sigla provincia, cap) e tipologia (ospedaliero, aziendale, hub).

### Cittadiniregistrati.txt

File di testo contenente un elenco di tutti quei cittadini che, dopo la vaccinazione, hanno effettuato la registrazione al sistema, anche in questo file in ogni linea sono presenti le informazioni di un solo cittadini, quali, nome e cognome, codice fiscale, indirizzo di posta elettronica, userid, password per poter accedere al sistema e id univoco di vaccinazione (fornito al momento della vaccinazione).

### Vaccinati_nomeCentro.txt

Esistono tanti file di questo tipo tanti quanti sono i centri vaccinali registrati all'interno del sistema, "nomeCentro" verrà sostituito dinamicamente con il nome del centro in questione. Ogni riga del file contiene le informazioni riguardo un solo cittadini vaccinato presso quel centro vaccinale. Tutti i file di questo tipo vengono creati dal sistema, non quando un operatore aggiunge un nuovo centro vaccinale, ma quando viene vaccinato un primo cittadino presso quel centro vaccinale.

Le informazioni riguardanti i cittadini contenute all'interno di questo file sono: nome e cognome del cittadino, codice fiscale, data somministrazione vaccino (GG/MM/AAAA), vaccino somministrato (Pfizer, Moderna, J&J) e id univoco di vaccinazione (id numerico su 16 bit generato casualmente).

Inoltre, all'interno di questo file, per i cittadini che hanno segnalato eventi avversi, vengono salvati i valori da loro inseriti e le relative note.

### valoriNomeEventoAvverso_nomeCentro.txt

questo file, come tutti gli altri file "valori*_nomeCentro.txt", contiene i valori relativi agli eventi avversi segnalati dai cittadini. Quando un cittadino segnala gli eventi avversi avuti, per ogni evento avverso inserisce un valore dal 1 al 5, una volta inseriti questi valori vengono salvati, anche, all'interno di questi file, perché tutti i valori vengono presi dal sistema e utilizzati per calcolare la gravità media relativa a ogni evento avverso ogni qual volta che un cittadino desidera consultare le informazioni riguardo a un centro vaccinale.

## Principali funzioni utilizzate dal programma

- registraCentroVaccinale()
- registraVaccinato()
- cercaCentroVaccinale()
- cercaCentroVaccinale_nome()
- visualizzaInfoCentroVaccinale()
- registraCittadino()
- inserisciEventiAvversi()

## registraCentroVaccinale()

Quando un operatore vaccinale inserisce i dati per registrare un nuovo centro vaccinale, i dati vengono presi dal sistema e salvati all'interno del file "centrivaccinali.txt", la funzione in questione si occupa di prendere questi dati e scriverli all'interno del file sopracitato.

```
public static void registraCentroVaccinale(String[] input)
```

come parametro a questa funzione viene passato un array di String contenente tutti i dati inseriti dall'operatore relativi al centro vaccinale. La funzione non fa altro che creare un collegamento con il file su cui scrivere, tramite oggetto della classe BufferedWriter, scorrere l'array tramite un ciclo *for* e scrivere il contenuto dell'array sul file. Infine, chiude il file.

## registraVaccinato()

Quando un operatore vaccinale inserisce i dati per registrare un nuovo cittadino a un centro vaccinale, i dati vengono presi dal sistema e salvati all'interno del file "vaccinati_nomeCentro.txt", la funzione in questione si occupa di prendere questi dati e scriverli all'interno del file sopracitato se il file esiste già, altrimenti prima di inserire i dati crea il file.

```
public static void registraVaccinato(String[] input,String nomeCentro)
```

i parametri che vengono passati a questa funzione sono un array di String, contenente tutti i dati relativi a un cittadino inseriti dall'operatore sanitario e il nome del centro vaccinale presso cui il cittadino è stato vaccinato. La funzione, quindi, si collega tramite a un oggetto BufferedWriter al file su cui scrivere, con un ciclo *for* scorre gli elementi presenti nell'array e gli scrive sul file. Poi la funzione genera l'id univoco di autenticazione, generando un numero casuale e andando a controllare che questo non sia già associato ad un altro cittadino.

Quando un operatore inserisce un cittadino in centro vaccinale, se il centro vaccinale in questione non aveva ancora cittadini al suo interno, cioè il cittadino è il primo ad essere inserito in quel centro, viene creato un nuovo file di testo denominato "*Vaccinati_NomeCentroVaccinale.txt*" all'interno del quale saranno presenti tutte le informazioni riguardanti il cittadino appena inserito e di tutti quelli che verranno inseriti successivamente.

## registraCittadino()

Funzione che si occupa di scrivere sul file "cittadiniregistrati.txt" tutte le informazioni di un cittadino quando esso, successivamente alla vaccinazione, si registra al sistema.

```
public void registraCittadino()
```

La funzione apre tramite oggetto di BufferedWriter il file, prende l'array di String associato all'oggetto cittadini, scorre i suoi elementi tramite un ciclo *for* e stampa tutte le informazioni presenti sull'array all'interno del file "cittadiniregistrati.txt".

## InserisciEventiAvversi()

Superato il controllo del login l'utente può andare ad inserire gli eventi avversi post vaccinazione tramite la funzione *inserisciEventiAvversi()*, gli eventi avversi vengono salvati nel file contenente tutte le informazioni di ogni cittadino, "*Vaccinati_NomeCentroVaccinale.txt*".

```
public static void inserisciEventiAvversi(String newString, String nomeCentro)
```

La funzione riceve come parametri una stringa di caratteri contenente tutti i dati di un cittadino più i valori relativi agli eventi avversi e il nome del centro vaccinale che serve per andare a selezionare il file su cui scrivere.

La funzione va a modificare il file "vaccinati_nomeCentro.txt" scrivendo al suo interno la stringa "newString" passata in input.

## cercaCentroVaccinale() e cercaCentroVaccinale_nome()

La ricerca di un centro vaccinale avviene tramite la funzione *cercaCentroVaccinale()*, questa può avvenire per nome del centro (il sistema prende in input una stringa di caratteri e restituisce tutti i centri vaccinali nel cui nome compare questa stringa) oppure per comune e tipologia, presi in input un comune e una tipologia, il sistema restituisce i centri vaccinali che soddisfano le richieste dell'utente, andando a cercare le informazioni all'interno del file "*CentriVaccinali.txt*".

Queste due funzioni sono molto simili e si occupano entrambe di andare a cercare all'interno del file "centrivaccinali.txt". Entrambe le funzioni lavorano con un oggetto della classe Scanner che legge tutte le linee del file in questione andando poi a stampare quelle linee che contengono i parametri cercati dall'utente.

La differenza tra le due sta nelle caratteristiche che vanno a cercare, appunto

```
public static int cercaCentroVaccinale(String comuneCentro, String tipoCentro)
```

ha due parametri passati in input, il nome di un comune e la tipologia di centro vaccinale.

Mentre

```
public static int cercaCentroVaccinale_Nome(String nomeCentro)
```

ha un solo parametro, il nome (o parte di esso) di un centro vaccinale.

## visualizzaInfoCentroVaccinale()

Una volta effettuata la ricerca il sistema stampa i risultati ottenuti, mandando in output i nomi dei centri vaccinali le cui caratteristiche corrispondono con quelle cercate dall'utente. Il cittadino può selezionare il centro vaccinale, tra quelli presentati in output dal sistema, di cui vuole leggere più informazioni; ciò è possibile tramite la funzione *visualizzaInfoCentroVaccinale()*. Questa funzione stampa un prospetto riassuntivo degli eventi avversi segnalati dai cittadini riguardo il centro vaccinale in questione, il quale riporta in forma anonima la severità media e il numero di segnalazioni di eventi avversi.

```
public static void visualizzaInfoCentroVaccinale(int mediaMdT, int mediaFebbre,
int mediaDolori, int mediaLinfoadenopatia, int mediaTachicardia, int
mediaCrisiI, int mediaAltro, int nSegnalazioni)
```

questa funzione, passati come parametri le medie, relative al centro vaccinale in questione, di ogni evento avverso (precedentemente calcolate) si occupa di stampare la tabella riassuntiva degli eventi avversi. Inoltre

l'ultimo parametro è un numero intero che rappresenta il numero di segnalazioni, su cui si basa il prospetto, che viene anch'esso stampato per maggiore chiarezza.

## Altre funzioni utilizzate

Il sistema oltre alle funzioni precedentemente descritte utilizza altre funzioni, alcune utilizzate per effettuare dei controlli e altre utilizzate per inviare dati a file di testo

### Controllo creazione id

```
public static boolean controllo_creazioneId(String id_input)
```

Sotto-funzione della funzione registravaccinato(), essa si occupa di controllare che l'id a 16 bit generato casualmente dalla funzione non appartenga già ad un altro cittadino, il sistema appunto controlla all'interno del file "idCittadini.txt" che non esista già un altro cittadino con lo stesso id. Se l'id generato non è un id già esistente restituisce un valore boolean = true che permette di avanzare.

### Controllo id Registrazione

```
public static boolean controllo_idRegistrazione(String id_input)
```

Funzione che permette di controllare se un cittadino ha già effettuato la registrazione al sistema, controlla che l'id inserito non sia già presente all'interno del file "cittadiniregistrati.txt", cioè che l'utente non sia già registrato. Se l'id è già presente restituisce un valore boolean = true che non permette di proseguire con la registrazione.

### Controllo id

```
public static boolean controllo_id(String nomeCentro, String id)
```

Funzione che permette di verificare se un cittadino che sta effettuando la registrazione abbia inserito il centro vaccinale giusto. La funzione verifica che all'interno del file "vaccinati_nomeCentro.txt" esista una linea contenente l'id passato come input.

### Controllo Login

```
public static boolean controllo_Login(String nomeUtente, String password)
```

Funzione che controlla che le credenziali inserite da un cittadino al momento del login siano corrette, se queste sono corrette la funzione restituisce un valore boolean = true che permette di superare il login. Altrimenti l'utente dovrà provare ad inserire altre credenziali.

La funzione lavora andando a leggere tramite uno Scanner il contenuto del file "cittadiniregistrati.txt" cercando una linea in cui sono presenti il nome utente e la password inseriti.

### Controllo Centro

Sia quando un utente si registra al sistema e sia ogni volta che effettua il login ad esso, gli viene chiesto di inserire il centro vaccinale presso il cui ha effettuato la vaccinazione

```
public static boolean controllo_centro(String nomeCentro, String nomeUtente)
```

questa funzione preso in ingresso il nome del centro vaccinale inserito dal cittadino e il nome utente di quest'ultimo, controlla tramite un oggetto della classe Scanner all'interno del file "vaccinati_nomeCentro.txt" che esista una linea contenente il nome utente inserito, cioè controlla che esista una linea contenente i dati di quel cittadino. Si occupa di controllare che il centro vaccinale inserito sia effettivamente quello presso cui il cittadino è stato vaccinato, se questo è vero restituisce una variabile boolean = true che permette rispettivamente di completare la registrazione o superare il login.

## Controllo Esistenza Centro

Funzione che viene utilizzata due volte, quando un operatore inserisce un nuovo cittadino e quando un cittadino cerca informazioni su un centro vaccinale.

Al momento dell'inserimento di un cittadino in un centro vaccinale la funzione effettua un controllo per verificare che l'operatore stia inserendo il cittadino in un centro vaccinale registrato.

Al momento della ricerca di un centro vaccinale, dopo che il sistema ha stampato tutti quei centri che soddisfano le richieste dell'utente, a quest'ultimo viene chiesto di selezionarne uno di questi scrivendone il suo nome per esteso. Nel caso però l'utente sbagli a digitare tale nome o ne scriva uno di un centro non esistente è necessario che il sistema si fermi.

```
public static boolean controlloEsistenzaCentro(String nomeCentro)
```

questa funzione, quindi, controlla utilizzando la classe Scanner che il nome inserito dall'utente sia effettivamente quello di un centro esistente all'interno del file "centrivaccinali.txt", nel caso in cui all'interno del file esista un centro con tale nome la funzione restituisce un valore boolean = true che permette poi di far avanzare il programma e quindi stampare la tabella riassuntiva degli eventi avversi. Se il nome inserito è sbagliato restituisce un valore boolean = false che non permette di proseguire e il sistema stampa un messaggio d'errore.

## Controllo Eventi Avversi

Funzione che controlla se un cittadino ha già inserito gli eventi avversi

```
public static boolean controlloEventiAvversi(String nomeCentro, String
nomeUtente)
```

il funzionamento della funzione si basa su un oggetto della classe Scanner che va a cercare all'interno del file "vaccinati_nomeCentro.txt" la linea contenente i dati del cittadino, quella che contiene il nome utente. Una volta trovata la linea controlla se all'interno di questa ci sia scritto "Eventi avversi", se è scritto vuole dire che il cittadino ha già inserito i dati e restituisce una variabile boolean = true. Se questa variabile è true il cittadino non potrà andare avanti ad inserire eventi avversi, in caso contrario invece si.

## Controllo Dati Registrazione

Al momento della registrazione l'utente deve inserire una serie di dati e questi devono corrispondere con quelli che l'operatore sanitario aveva inserito nel file "vaccinati_nomecentro.txt" al momento della vaccinazione.

```
public static boolean controllo_registrazione(String idUtente, String dato,
String nomeCentro)
```

Questa funzione ogni volta che l'utente inserisce una dato (che è presente anche nel file "vaccinati_nomeCentro.txt") controlla che il dato inserito e quello già registrato coincidano. Se un dato coincide restituisce un valore boolean = true che permette di continuare con l'inserimento di altri dati, altrimenti l'utente deve riprovare inserendo altri dati.

## Trova Linea Cittadino

Dopo che un cittadino ha inserito degli eventi avversi la linea contenente i suoi dati nel file "vaccinati_nomeCentro.txt" viene modificata.

```
public static String trovaLineaCittadino(String nomeCentro, String nomeUtente,
String[] input_eventi)
```

Questa funzione si occupa di andare a trovare la linea contenente i dati del cittadino nel file sopracitato e la restituisce (tramite *return*). La String restituita verrà trattata in seguito dalla funzione "inserisciEventiAvversi()".

## Numero di segnalazioni

Abbiamo già detto che quando il sistema stampa la tabella contenente il riassunto degli eventi avversi relativi a un centro vaccinale per una maggiore trasparenza specifica anche su quante segnalazioni si basa tale riassunto

```
public static int numSegnalazioni(String nomeCentro)
```

questa funzione si occupa di contare il numero di segnalazioni relative ad un centro vaccinale, tramite un'oggetto della classe Scanner legge un file del tipo "vaccinati_nomeCentro" e ogni qual volta incontra una linea che al suo interno contiene le segnalazioni dell'utente (contiene "Eventi avversi") aumenta di uno il conteggio di un contatore (contatore++). Finito di leggere tutte le linee del file, restituisce il valore di tale contatore.

## Invio_valore

Questa funzione viene utilizzata più volte, ogni volta che un cittadino inserisce un valore relativo ad un evento avverso essa si occupa di mandare tale valore al file contenente giusto.

```
public static void invio_valore(String nomeCentro, String evento, String value)
```

Tale funzione tramite un oggetto della classe BufferedWriter va ad aprire il file "valoriEvento_nomeCentro.txt" e il valore inserito dall'utente (value), viene scritto all'interno di questo file dove saranno presenti tutti gli altri valori riferiti a tale evento segnalati da cittadini di un determinato centro vaccinale.

Se questo file non esiste ancora, perché nessun cittadino di questo centro vaccinale ha ancora segnalato eventi avversi, il file viene automaticamente creato.

## Media

Questa funzione viene utilizzata più volte, per ogni evento avverso segnalabile dagli utenti.

```
public static int media(String nomeCentro, String evento)
```

questa funzione viene utilizzata quando un cittadino vuole vedere il prospetto riassuntivo degli eventi avversi legati a un centro vaccinale. Quindi, apre il file "valoriEvento_nomeCentro.txt", legge (tramite Scanner) tutti i valori presente al suo interno ed effettua una media aritmetica dividendo la somma dei valori ottenuta per il numero di questi. La media viene poi restituita e verrà stampata all'interno della tabella riassuntiva.

### Note

Funzione utilizzata più volte. Dopo che un utente ha inserito l'intensità di un evento avverso gli viene chiesto se vuole inserire delle note.

```
public static String note()
```

Questa funzione si occupa della procedura di inserimento delle note. Chiede se l'utente vuole inserire delle note, in caso di risposta affermativa si occupa di controllare che queste non superino i 256 caratteri mentre. Restituisce una variabile di tipo String contenente o la note inserita dall'utente o un messaggio standard se l'utente non ha inserito note.

## Bibliografia


Geeksforgeeks,

- Scanner class - https://www.geeksforgeeks.org/scanner-class-in-java/
- FileWriter class - https://www.geeksforgeeks.org/filewriter-class-in-java/
- FileInputStream class - https://www.geeksforgeeks.org/java-io-fileinputstream-class-java/

"Dai fondamenti agli oggetti", Giovanni Pighizzini, Pearson – BufferedWriter class