1) In Java non esiste la funzione scanf() come in C ma ci sono comunque diverse possibilità per leggere l'input da tastiera. Considerare la seguente classe:

```
import java.util.Scanner;
public class ProgrammaInterattivo {
    public static void main(String args[]) {
            Scanner scanner = new Scanner(System.in);
            String stringa = "";
            System.out.println("Digita qualcosa e batti enter, oppure " +
                                "scrivi \"fine\" per terminare il programma");

            while(!(stringa = scanner.next()).equals("fine")) {
                    System.out.println("Hai digitato " + stringa.toUpperCase() + "!");
            }
            System.out.println("Fine programma!");
    }
}
```

ProgrammaInterattivo legge l'input da tastiera mediante la classe Scanner del package java.util. Il metodo next() usato nel costrutto while è un metodo bloccante che legge l'input da tastiera sino a quando si preme il tasto "enter" (invio). Il programma termina quando si digita la parola "fine".
Modificare il programma precedente in modo tale che diventi un moderatore di parole, ovvero che non stampi nulla se si digita una delle parole alfa, beta, gamma, delta, epsilon, kappa, lambda, sigma, omega (come parole singole, non all'interno di una frase).
(Suggerimenti: memorizzate in un array le parole da non stampare e verificate quindi ogni volta che si inserisce una parola da tastiera che questa non sia contenuta nell'array)

2) Scrivere un programma con i seguenti requisiti.
   - Utilizza una classe *Principale* che, nel metodo *main()*, crea una lista (vuota) di oggetti *Persona* con l'istruzione `ArrayList<Persona> miaLista = new ArrayList<>();`
   - Crea alcuni oggetti *Persona* e li aggiunge alla lista col comando *add*
   - Verifica la dimensione della lista col metodo *size*
   - Prende l'i-esimo elemento dalla lista col metodo *get* e lo stampa
   - Elimina l'i-esimo elemento dalla lista col metodo *remove*
   - Stampa il contenuto dell'intera lista usando un ciclo *for* e il metodo *size* per conoscere la dimensione della lista oppure usando un ciclo for migliorato (che succede se passate *l'ArrayList* alla *println*?)
   - Cancella l'intera lista col metodo *clear* e verifica che si sia effettivamente svuotata col metodo *size*

   Nota:
   Per usare un'*ArrayList* dovete inserire il comando `import java.util.ArrayList;` nella classe che lo utilizza. Verificare nella pagina della documentazione ufficiale di Java come si usano i metodi indicati.

3) Modificate il moderatore di parole dell'esercizio 1 utilizzando un `ArrayList<String>` invece dell'array per le parole da non stampare.
   (Suggerimento: la classe ArrayList ha un metodo boolean contains(Object o) che verifica se un oggetto, nel vostro caso una String, è contenuta nell'ArrayList)

4) Basandovi sul primo e secondo esercizio dell'esercitazione provate a scrivere un programma interattivo che, utilizzando la classe *Persona* e un *ArrayList<Persona>*, dia la possibilità, tramite un semplice menu testuale, di scegliere fra le seguenti opzioni:
   a. inserire una nuova persona nella lista con dati inseriti dall'utente
   b. cercare una persona nella lista in base al nome
   c. stampare l'intera lista
   d. cancellare l'intera lista
   e. esci

SEGUE

5) Un punto nel piano si può rappresentare tramite una classe con due variabili d'istanza: *x* e *y* di tipo *double*. Scrivete una classe con tutti i metodi e i costruttori che ritenete utili (incluso il metodo `String toString()` e `boolean equals(Punto)`). Scrivere quindi un programma che realizzi le seguenti operazioni con punti nel piano:
   - Crea due punti con dati inseriti da tastiera dall'utente (usate il metodo *nextDouble* di *Scanner*)
   - Dati i due punti, ne calcola la loro distanza
   - Dati i due punti, determina il punto medio del segmento che li unisce
   - Dati i due punti verifica se sono uguali (usando il metodo *equals* che avete implementato)

NOTE PER COMPILAZIONE E TEST A RIGA DI COMANDO IN AMBIENTE LINUX:

`javac -d . nomeClasse.java`	*compila e genera il bytecode*
`java nomePackage.nomeClasse`	*esegue il bytecode sulla JVM*