Inserire nel package **it.unipa.community.nomecognome.prg.n09.es0X** le seguenti applicazioni.

1) Riprendete la gerarchia di classi dell'esercitazione 7 in cui avete implementato in ogni classe la propria versione della funzione membro **draw()** per il tracciamento della figura.
   Utilizzate le vostre classi per creare una **List<Shape>** in cui memorizzare alcuni oggetti delle varie classi (usate il metodo **add()**). Dopo aver inserito gli oggetti nella **List<Shape>** attraversatela interamente utilizzando un **Iterator<Shape>** e chiamate il metodo **draw()** di ciascun oggetto per stampare i dettagli.

2) Utilizzando una **Map<String, Shape>** scrivete un programma in cui inserite in una mappa usando il metodo **get()** forme geometriche di vario tipo con valori casuali e a cui assocerete un nome univoco (ad esempio "cerchio1", "quadrato2", "Miorettangolo", etc…). Verificate come col metodo **get()** sia possibile accedere al valore dell'elemento della mappa in base al valore della chiave.

3) Scrivete un metodo static nella classe che contiene il main a cui passare una **ArrayList** di oggetti di qualsiasi tipo e che, tramite un **Iterator**, stampa l'intera collezione.

4) Scrivete un metodo analogo a quello dell'esercizio precedente ma che riceve un **ArrayList** di oggetti che estendono **Shape** e, tramite un **Iterator**, invoca il metodo **draw()** per tutti gli elementi della collezione. Provate a passare un **ArrayList<String>** e verificate che non compila.

5) Definite una classe generica **Stack<T>** che implementi una pila di 100 elementi di tipo T tramite un array. Le funzioni membro della classe devono essere:
   `void push(char), char pop(), boolean isEmpty(), boolean isFull().`
   Scrivete un programma che crea un oggetto **Stack<String> e uno Stack<Integer>** e verifica il corretto funzionamento della classe.

6) Prendendo spunto dall'esempio **Animale**, **Carnivoro**, **Erbivoro** visto a lezione si modifichino le classi **Numero, Razionale, Complex** delle esercitazioni precedenti in modo da dare un errore in compilazione quando si tenta di effettuare operazioni non consentite (operazione aritmetica fra **Razionale** e **Complex**) invece di lanciare una IllegalArgumentException a tempo di esecuzione.
   Definite un'interfaccia generica per rappresentare **Aritmetica<Complex>** e **Aritmetica<Razionale>** in cui sono definite le 4 operazioni aritmetiche. Le classi **Razionale** e **Complex** dovranno quindi implementare rispettivamente **Aritmetica<Razionale>** e **Aritmetica<Complex>**.

7) Scrivete un programma che, utilizzando il metodo **join()**, concatena le stringhe "uno", "due", "tre" separandole con uno spazio, le memorizza in una nuova stringa e la stampa. Convertite quindi tutta la stringa in maiuscolo e, utilizzando **replace()**, sostituite allo spazio due trattini "**--**". Stampate quindi il risultato. Riconvertite ora la stringa in minuscolo e, utilizzando il metodo **split("--")**, estraete nuovamente le stringhe "uno", "due", "tre" e stampatele.

8) Scrivete un programma che, utilizzando il metodo split su una stringa contenente il testo di questo esercizio, determina il numero totale di parole presenti nel testo e la parola che compare con maggiore frequenza. Potreste anche pensare di utilizzare una **Map<String, Integer>** per memorizzare la frequenza di ciascuna parola utilizzando la parola stessa come chiave.

9) Implementate i metodi **equals()** e **hashcode()** per le classi **Razionale**, **Complex** e **Date** e verificatene il corretto funzionamento con un programma di test.

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

*Creare una cartella col proprio cognome sulla scrivania con all'interno le directory **classes** e **src**.*

*Aprire una finestra di **terminale** e digitare:*
cd Desktop/cognome/classes** oppure **cd Scrivania/cognome/classes

*Creare i file sorgente con **gedit** e salvarli nelle sottodirectory di **src** corrispondenti al package.*

Digitare:
***javac -d . ../src/it/unipa/.../prg/n09/es0X/*.java** (compila e genera il bytecode)*
***java it.unipa.community.....prg.n09.es0X.Main** (esegue il bytecode sulla JVM)*