

## Esercitazione 3

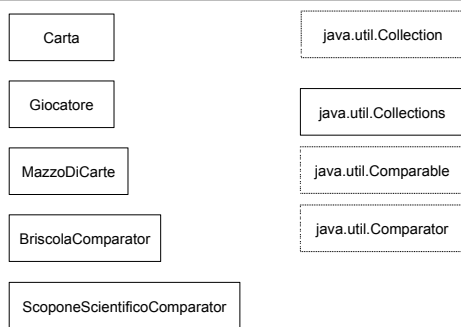
Programmazione 2

Nota: per esclusivo uso interno al corso, riferimenti bibliografici forniti a lezione.

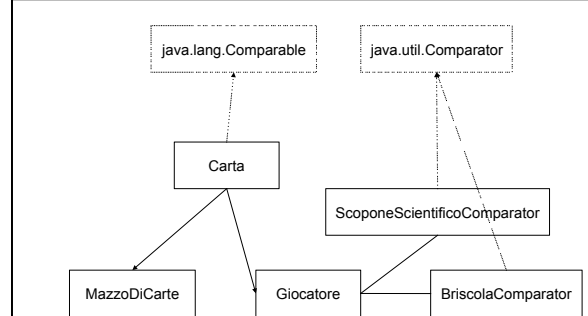
## Esercizio Carte da Gioco

Creare un'applicazione che generi in modo casuale una mano di carte da gioco, pescandola da un mazzo di carte, e le ordini in modo coerente al gioco della Briscola o dello Scopone Scientifico a seconda di un parametro passato da riga di comando.

### Traccia per il diagramma delle classi



### Traccia per il diagramma delle classi



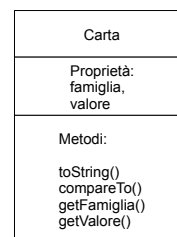
### Proprietà e metodi di ciascuna classe

Identificare le proprietà e i metodi di ciascuna classe:

- Carta
- MazzoDiCarte
- Giocatore

Implementare una classe con un metodo main che simuli la distribuzione delle carte ad un giocatore e le stampi dopo averle ordinate secondo il criterio specificato dall'utente.

### Proprietà e metodi di ciascuna classe



### Proprietà e metodi di ciascuna classe

MazzoDiCarte
Proprietà: carte
Metodi: mescola() daiCarta() eVuoto()

### Proprietà e metodi di ciascuna classe

Giocatore
Proprietà: carteInMano
Metodi: stampaMano() ordinaPerBriscola() ordinaPerScoponeSci()

### Esercizio

Documentare le classi sviluppate per l'esercizio "carte da gioco" con il tool javadoc

### Esercizio 2

Scrivere un'applicazione che, data una stringa di testo sufficientemente lunga, definisca i tre metodi:

- contaParole: deve ritornare il numero di parole presenti nel testo
- contaParoleDistinte: deve ritornare il numero di parole distinte presenti nel testo
- contaOccorrenze: deve ritornare il numero di parole distinte e per ciascuna parola il numero delle volte in cui è presente.

Scegliere delle strutture dati appropriate in cui memorizzare le informazioni lette.

### Esercizio 2

Completare l'esercizio precedente aggiungendo una funzionalità che generi automaticamente delle "terzine" con le parole presenti nel testo originale e le stampi a video.

Le "terzine" devono essere:

- gruppi di tre righe di otto parole;
- composte da parole scelte in modo casuale dalla totalità delle parole presenti nel testo