ESERCIZI IN JAVA

Scrivere un programma Java nel cui main viene svolto il seguente compito

1. Calcolare la media di N voti inseriti da tastiera
2. Dato come input il numero di secondi, convertire il tempo in ore, minuti e secondi
3. Una scuola è composta da N classi. Per ogni classe, viene inserito da tastiera il numero di studenti. Calcolare quanti frequentano la scuola e in media quanti studenti ci sono per classe, il numero minimo e il numero massimo.
4. Scrivere una programma che legge da tastiera un intero e stampa un messaggio a video secondo la seguente tabella:

Valore	*Messaggio*
v >= 700	Montagna
700 > v >= 300	Collina
300 > v >= 0	Pianura

5. Scrivere una programma java che legga da input una stringa rappresentante il nome di un mese dell'anno, e stampi un messaggio indicante il numero di giorni di quel mese [si considera sempre un anno come non bisestile].
6. Data una sequenza di stringhe con il seguenti formato *cognome*nome*eta*sesso*, che termina con l'introduzione della parola fine, visualizzare per ogni stringa introdotta il cognome in maiuscolo e il nome con l'iniziale maiuscola. Al termine degli inserimenti calcolare l'età media delle donne e degli uomini. (Usare i metodi della classe String: toUpperCase, indexOf, substring, concat,…)

Oggetti in Java

Dei seguenti esercizi rappresenta il diagramma delle classi in UML e implementa la classe e un main di esempio in Java

7. Dichiarare una classe di nome “temperatura” che ha come attributo la temperatura in gradi centigradi (temp e i metodi getGradiC e setGradiC). Definire un costruttore per inizializzare la temperatura, e un metodo (getGradiF) per convertire la temperatura da gradi centigradi a Fahrenheit, sapendo che: F = 32 + (9 * C / 5)
8. Dichiarare una classe di nome “velocita” che ha come attributo la velocità in km/h (vel). Possiede solo il costruttore senza parametri che inizializza la velocità a 0. Definire i 4 metodi get e set che restituiscono/impostano la velocità in Km/h o in m/s.
9. La classe TELECOMANDO deve essere in grado di: accendere/spegnere la televisione, cambiare il canale (su di uno o giù di uno) o impostare un canale (se esiste se non resta invariato), abbassare e alzare il volume o mettere a muto. Usare gli attributi privati: canale, acceso, volume, VOLUME_MAX, CANALE_MAX
10. La classe Cerchio deve avere due costruttori: uno senza parametri che imposta il raggio a 0 e uno con un parametro corrispondente al raggio che si vuole impostare. Deve fornire oltre ai metodi get/set opportuni anche i metodi getArea() e getCirconferenza()
11. La classe rettangolo deve permettere di calcolare Area e Perimetro e di dire se è un quadrato. Possiede due costruttori uno senza parametri e uno con base e altezza passati come argomenti.
12. Date le classi Cerchio e Rettangolo degli esercizi precedenti costruire un main in cui dati un cerchio di raggio r e un rettangolo b x h dire se sono equivalenti (hanno la stessa area)
13. Scrivere la classe Motorino con i seguenti attributi: colore, velocità in km/h, tipo (marca + modello), antifurto (boolean che esprime se l'antifurto è inserito oppure no), velocitàMax. Il costruttore ha come parametri un colore, un tipo e una velocità massima. Scrivere il metodo getVelocità che restituisce la velocità del motorino, il metodo accelera(diX) che incrementa diX la velocità se non è inserito l'antifurto, se no non fa nulla e il metodo inserisciAntifurto oltre ai metodi get/set che ritieni opportuni
14. Scrivere una classe Java Concatena2 che stampi: le prime tre lettere del proprio nome concatenate con le prime tre lettere del proprio cognome. Utilizzare il metodo concat ed il metodo substring
15. Dichiarare una classe di nome “persone” che ha come attributi un vettore di tipo string (nomi) e un attributo di tipo char (vocale). Definire un costruttore per inizializzare la vocale, e 2 metodi, il primo (add) per aggiungere un nome, il secondo (visualizza) per visualizzare tutti i nomi che iniziano con quella vocale.
16. Utilizzate un array unidimensionale per risolvere questo problema: leggete 20 numeri compresi tra 10 e 100. Per ogni numero letto, visualizzatelo solo se esso non è già stato immesso in precedenza.

Eccezioni Personalizzate – Vettori di Oggetti

17. Scrivere una programma java per rappresentare dei giocatori di baseball. Per ciascun giocatore devono essere rappresentati il nome come stringa, un punteggio che rappresenta la bravura del giocatore in battuta e la sua età. Si devono quindi realizzare i seguenti metodi:
    - Giocatore(String nome, int eta) costruttore con due argomenti.
    - Giocatore(String nome, int eta, double valore) costruttore con tre argomenti.
    - String getNome() che ritorna il nome del giocatore.
    - double getPunteggio() che ritorna il punteggio del giocatore.
    - void setPunteggio(double nuovoPunteggio) che pone il punteggio del giocatore a nuovoPunteggio.
    - int getEta() che ritorna l'età del giocatore
    - String toString() che visualizza tutti i dati del giocatore

    Fai un menu in cui sia possibile inserire un nuovo giocatore, visualizzare i suoi dati.
18. Scrivere una classe java Squadra che ha come attributo un nome e un array di Giocatore dell'esercizio precedente ma con in più l'attributo ruolo che assume i valori di una enumerazione (BATTITORE, PRIMA BASE, …). Il costruttore crea una classe con un array di N_MAX_GIOCATORI vuoto e possiede i seguenti metodi:
    - +addGiocatore(Giocatore g)
    - +void setPunteggio(String nome, double valore)
    - +double getPunteggioMedio() della squadra
    - +passatoAnno() incrementa tutte le età
    - +String toString()
19. Scrivere una classe java UtilityStringhe1 che contenga un metodo statico static void stampaIniziali(String nome, String cognome) che stampa a schermo le iniziali di nome e cognome. Scrivere una classe cliente ClienteUtilityStringhe1 che contiene un metodo main che prende due stringhe da tastiera (nome e cognome) e utilizzando il metodo stampaIniziali stampare le iniziali a schermo
20. Creare le classi necessarie per provare a vincere all'ambo su una ruota, raddoppiando sempre la giocata fatta. Quando si vince visualizzare la vincita (250 volte la puntata) e la somma complessiva giocata
21. Simulare il lancio di un tot di dadi e fare la somma dei numeri usciti. Deve essere possibile stabilire il numero di facce di un dado
22. Creare una classe *VettoriInteri* che:
    - Metodo costruttore con parametro un numero corrispondente alla dimensione desiderata che inizializza tutti gli elementi a 0
    - Metodo costruttore che riceve una stinga con gli elementi separati dalla "|" e carica il vettore con essi dimensionandolo della lunghezza giusta. Se nella stringa ci sono dei dati non corretti deve generare una eccezione *NumberFormatException*
    - Abbia un metodo che ritorna l'elemento minimo di un array
    - Metodo che cerca un elemento nell'array e ritorna la posizione (se l'elemento non esiste ritorna -1)
    - Metodo che elimina la prima occorrenza nell'array, dato un numero (se non trova nulla deve restituire false)
    - Metodo toString() che restituisce gli elementi separati da "|"
23. Dichiarare una classe di nome carburante che ha come attributo un vettore di tipo float (contenente i vari prezzi benzina) e 3 metodi di nome media, max e min, che restituiscono il prezzo medio, massimo e minimo.
24. Scrivere una programma java che accetti in ingresso una serie di stringhe, le memorizzi in un oggetto e, al termine dell'inserimento, stampi la stringa più lunga immessa.
25. In un porto si affittano 100 posti barche di diverso tipo. Per ogni affitto si salvano il C.F e il nome del cliente, la data inizio e fine dell'affitto (tipo GregorianCalendar) e la barca che lo occuperà. Una barca si caratterizza per matricola, dimensione in metri e anno di fabbricazione. Un affitto si calcola moltiplicando il numero di giorni (affitto) per i metri della barca per un coefficiente € al metro. Realizzare oltre i metodi set e get opportuni, i seguenti metodi
    - Costruttore del porto di 100 posti inizialmente liberi con il prezzo al metro passato per parametro
    - Metodo *nomeAffittuario*: dato in input un posto barca, restituire il nome dell' affittuario
    - Metodo *getPrezzo*: dato in input un posto barca restituire il prezzo
    - Metodo *controlla*: dato in input posto barca restituire se è libero o no
    - Metodo *affittaPosto*: dato in input posto barca se è libero affittarlo al nuovo affittuario per il periodo e per la barca indicata, se no genera un'eccezione
    - Metodo *liberaPosto*: dato in input posto barca se è occupato renderlo libero visualizzando il totale dovuto, se no genera un'eccezione
    - Metodi *toString()* per la barca, l'affitto e il porto

26. Scrivere una classe Proprietà che mantiene informazioni su proprietari di appartamenti. Ogni oggetto Proprietario deve contenere una stringa che indica il nome del proprietario, ed un vector di appartamenti di tipo stringa che possono contenere ciascuno l'indirizzo di un appartamento posseduto da quel proprietario. Le classi devono contenere:
    - costruttore che imposta il nome del proprietario ed il numero di appartamenti (ti serve per dimensionare il vector appartamenti, imposta l'incremento a 2)
    - metodo per assegnare un indirizzo di un appartamento ad un proprietario
    - metodo che restituisce l'indirizzo contenuto in una cella o "nessuno" se la cella è vuota
    - metodo che restituisce il numero di appartamenti inseriti
    - metodo che visualizza tutti gli appartamenti di un proprietario X inserito da tastiera
    - metodo che elimina un appartamento numero N di un proprietario X inserito da tastiera
    - metodo che visualizza tutti i proprietari con relativi appartamenti
    - metodo che elimina il proprietario X inserito da tastiera e i suoi appartamenti

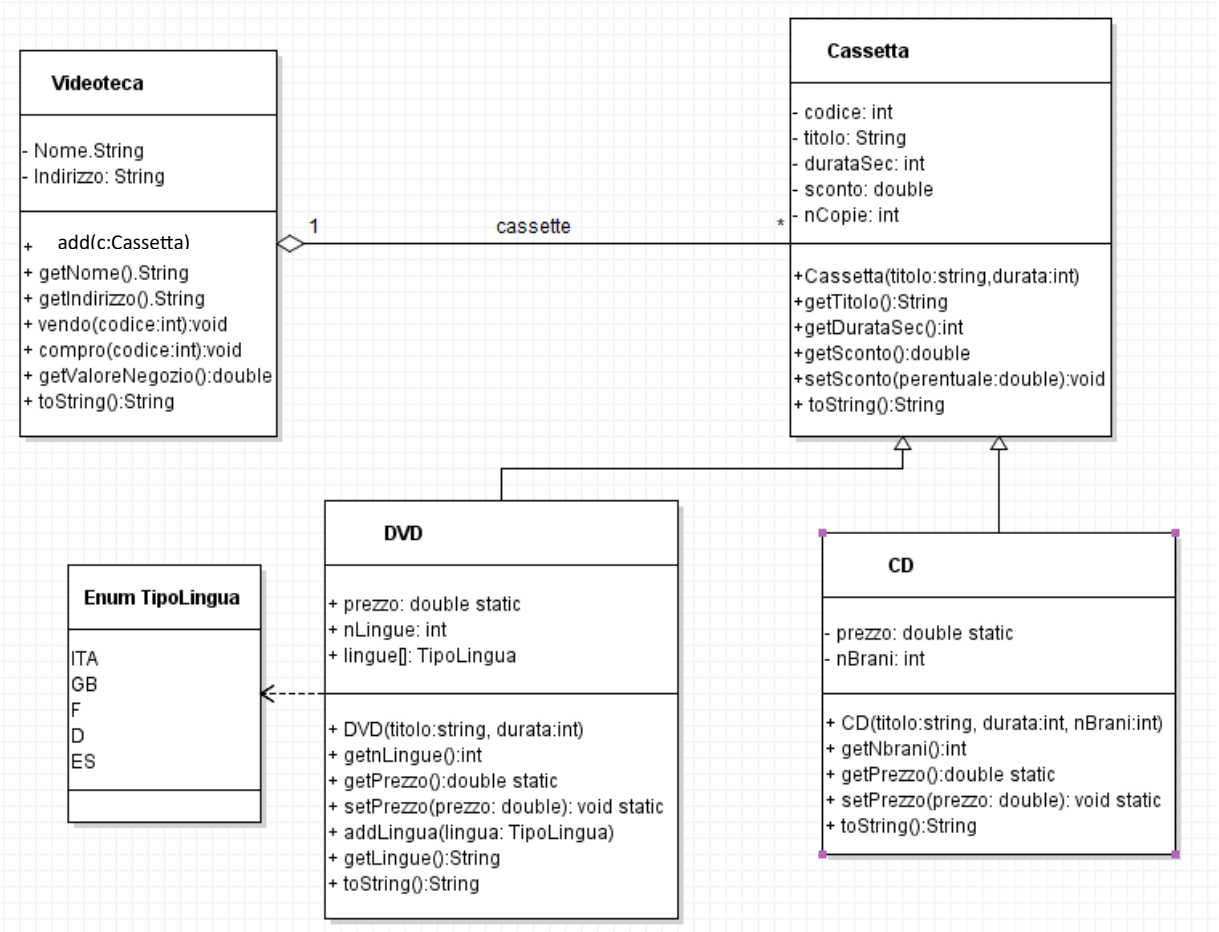
    Realizzare la classe `Inizio` con un menu per verificare il corretto funzionamento di tutte le funzioni.
27. Un importante locale commerciale della provincia di cuneo, chiede la realizzazione di un programma che gestisca le sue vendite. Il programma in questione deve permettere:
    A. Inserimento dei prodotti in vendita. Ogni prodotto deve avere: Codice(valore numerico), Descrizione e prezzo unitario.
    B. Registrazione vendite: di ogni vendita si conosce data, Codice venditore (Numero da 0 - 9), prodotto venduto, quantità venduta e come hanno pagato (0 - CONTANTI, 1 - BONIFICO, 2 - CARTA DI CREDITO).
    C. Determinare quale venditore ha venduto più articoli.
    D. Determinare se un venditore(dato in input) ha realizzato una vendita, se sì mostrare tutti i dati degli articoli venduti, altrimenti messaggio di errore!
    E. Determinare il valore totale di tutte le vendite.
    F. Visualizzare la singola vendita con incasso più elevato pagata con carta di credito.
28. Creare un applicazione JAVA per gestire un noleggio video. Il noleggio video possiede diversi tipi di prodotti. Tutti i prodotti possiedono: Titolo, Genere (Horror, Dramma, Avventura, Bambini), Prezzo noleggio al giorno, noleggiato (si/no). Si gestisca una lista di noleggi: NominativoCliente, Prodotto noleggiato, DataPrestito, DataReso. Creare un'applicazione (console) che:
    - Al momento della registrazione del reso visualizzi l'importo dovuto in base ai giorni di noleggio
    - Visualizza la lista dei film noleggiati
    - Dato un NominativoCliente visualizza i film noleggiati
    - Dato un NominativoCliente visualizza l'importo speso fino ad oggi
    - Visualizzare tutti i film dato il genere
    - Eliminare un cliente (dato il NominativoCliente)
    - Eliminare un film (dato il titolo)
29. Gestire i propri voti Scritto, Orale e Pratico per le varie materie e la data in cui sono stati presi. Gestire le eccezioni se i dati inseriti non sono corretti (per la data deve essere non successiva ad oggi e nel periodo corrente). Realizzare un programma che:
    - All'avvio viene richiesta la data di inizio e fine anno scolastico e di fine 1^ quadrimestre
    - Visualizzi i voti di una materia e ne calcoli la media
    - Calcolare la media globale di tutte le materia del primo, secondo e dell'anno
    - Media di S, O e P e media complessiva dei voti registrati fino ad ora di una materia
    - Cancellare i voti insufficienti
30. Una fabbrica di automobili produce 4 modelli di auto con prezzo di vendita: Modella A 9000 €, Modella B 10.500€, Modella C 14.500, Modella D 17.200€. Realizzare un programma che memorizzi in un file le varie vendite e visualizzi:
    - Il volume di vendita totale (in €)
    - La percentuale vendite di ogni modello
    - Dato in input un modello visualizzare quante auto sono state vendute

EREDITARIETÀ

31. Si scriva una classe Libro le cui istanze rappresentano libri. Ogni libro è caratterizzato dal numero di pagine. La classe deve contenere una variabile di istanza intera, numPagine, un costruttore che assegna al libro un numero specificato di pagine, ed un metodo pageMessage() che stampa il numero di pagine del libro.

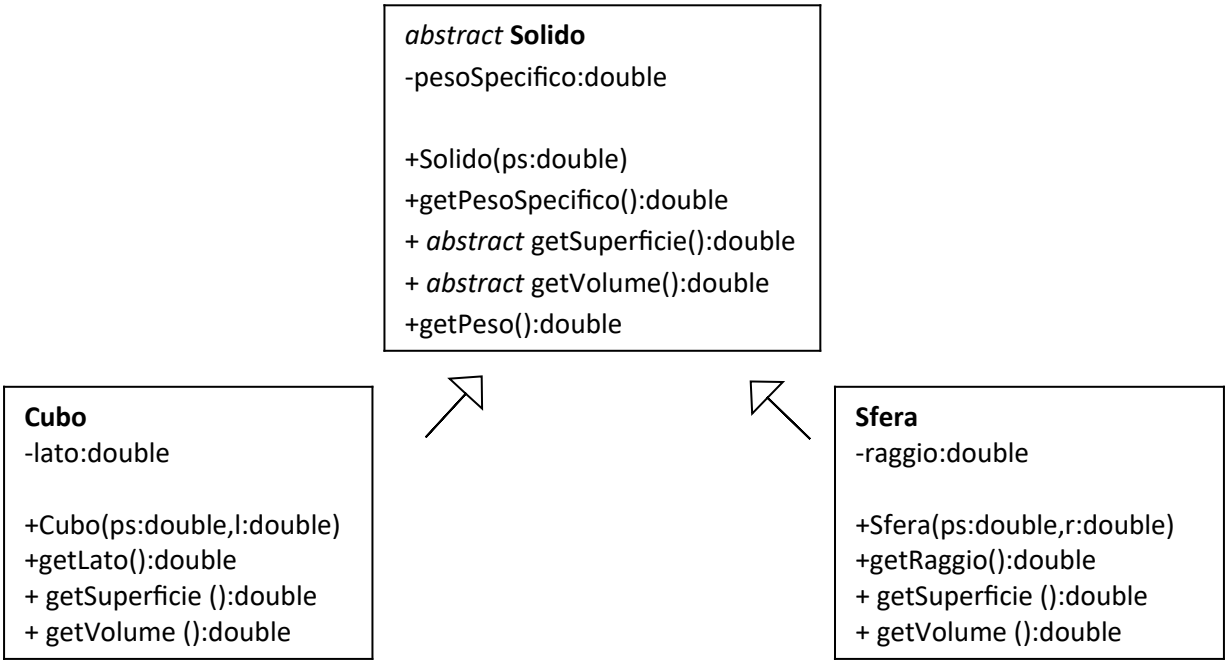
Scrivere quindi una classe Vocabolario che estende la classe Libro. La classe deve contenere una nuova variabile di istanza, numDefinizioni, un costruttore che assegna al vocabolario un numero di pagine e un numero di definizioni specificati, ed un metodo definitionMessage() che stampa un messaggio contenente il numero di pagine, il numero di definizioni ed il numero medio di definizioni per pagina del vocabolario.
Scrivere infine un programma di prova per collaudare le classi e i metodi.

32. Si scriva una classe Lavoratore le cui istanze rappresentano lavoratori. Ogni lavoratore è caratterizzato da un nome, un livello ed uno stipendio (mensile). La classe deve contenere una variabile di istanza di tipo stringa nome e due variabili di istanza intere, livello e stipendio, un costruttore che assegna al lavoratore un nome ed un livello specificati, ed un metodo stampaStipendio() che stampa lo stipendio del lavoratore.
Scrivere quindi una classe LavoratoreConStraordinariPagati che estende la classe Lavoratore. La classe deve contenere una nuova variabile di istanza, oreStraordinario, un costruttore che assegna al lavoratore con straordinari pagati un nome, un livello e un numero di ore di straordinario specificati, un metodo stampaStraordinari() che stampa un messaggio contenente il numero di ore di straordinario. La classe deve inoltre contenere una variabile statica reale retribuzioneOraria, inizializzata a 10.0, corrispondente alla retribuzione di un'ora di straordinario (che si suppone uguale per tutti). La classe deve inoltre ridefinire il metodo stampaStipendio() per tenere conto della retribuzione delle ore di straordinario: alla retribuzione base (ereditata da Lavoratore) va sommata la retribuzione degli straordinari, ottenuta come retribuzione oraria dello straordinario per numero di ore di straordinario effettuate. Scrivere infine un programma di prova per collaudare le classi e i metodi.
33. Si vogliono gestire delle videoteche, che commerciano film in DVD e CD musicali. Il prezzo dei DVD è unico per tutti i DVD, lo stesso per i CD. Ogni cassetta ha un codice numerico gestito in automatico dal sistema, può avere o meno uno sconto da applicare al prezzo base (inizialmente 0), ha una durata complessiva in secondi e un numero di copie presenti in negozio (questa quantità inizialmente è 0, ma si modifica di 1 vendendo o comprando delle copie con i rispettivi metodi). I DVD sono caratterizzati dalle lingue possibili per il film (al massimo 5), che vengono aggiunte con il metodo *addLingua*() e sono visualizzabili con il metodo *getLingue().* I CD sono caratterizzati dal numero di brani presenti. Si vuole in ogni momento conoscere il valore complessivo delle cassette presenti in negozio con il metodo *getValoreNegozio()* (tenendo conto del numero di copie e degli eventuali sconti). Questa realtà è rappresentata dal seguente UML



Realizzare una classe Test in cui si istanzia una videoteca con almento 2 DVD e 2 CD, si vendano e si comprino delle cassette e si visualizzi il valore complessivo

34. Realizza le seguenti classi e crea un main in cui si utilizza un vettore di Solidi e si istanziano 2 sfere e 2 cubi e si visualizzano tutte le caratteristiche



35. Le Figure si dividono in Cerchi, Quadrati e Triangoli. Ogni figura mette a disposizione i metodi draw() e erase() per rispettivamente stampare e cancellare la figura. Ipotizzando che draw() e erase() stampino semplicemente delle stringhe (esempio "Cerchio disegnato" e "Cerchio cancellato") che riportano l'operazione che si sta eseguendo realizzare le varie classi a partire dall'interfaccia Figura.
    Creare una nuova classe RandomShapeGenerator che mediante un metodo next() ritorni in modo random un'istanza di Circle, Square o Triangolo.

```
double x = Math.random();              // valore compreso [0..1)
double x = Math.random() * 10;         // valore compreso [0..10)
```

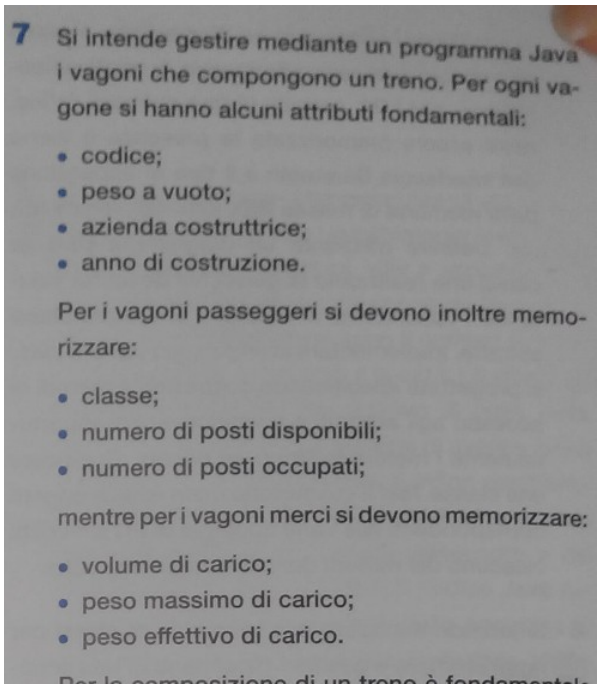
DIAGRAMMI UML

Progettare il diagramma delle classi per realizzare in Java il programma che risolve i seguenti problemi:

36.

6 Un'agenzia di pratiche automobilistiche deve commissionare un software per il pagamento delle tasse annuali sui veicoli. Un professionista viene incaricato di progettare e implementare la gerarchia di classi che rappresentano i veicoli. Durante un'intervista al proprietario dell'agenzia emerge quanto segue:

- i veicoli possono essere motoveicoli o autoveicoli;
- per ogni veicolo è necessario memorizzare la targa, la marca, il modello, l'anno di immatricolazione e il numero di passeggeri consentito oltre al conducente;
- i motoveicoli sono sempre alimentati a benzina e sono caratterizzati dalla potenza espressa in HP: la tassa viene calcolata moltiplicando per la potenza un valore che a oggi vale 1,5 €/HP;
- gli autoveicoli tradizionali possono essere alimentati a benzina o a gasolio e sono caratterizzati dalla potenza espressa in HP: la tassa viene calcolata moltiplicando per la potenza un valore che a oggi vale 2,5 €/HP;
- per gli autoveicoli alimentati a gas, oltre alla potenza è necessario memorizzare il tipo gas (GPL o metano): questi autoveicoli non pagano nessuna tassa per i primi 5 anni dall'immatricolazione, trascorso questo periodo la tassa viene calcolata moltiplicando per la potenza un valore che a oggi vale 0,5 €/HP per il metano e 0,75 €/HP per il GPL;
- gli autoveicoli alimentati a gas idrogeno pagano una tassa che aumenta di 0,1 €/HP per ogni anno di vita del veicolo a partire da una tassa iniziale pari a 0 €/HP il primo anno di immatricolazione;
- come incentivo governativo gli autoveicoli elettrici non pagano alcuna tassa.

37. 

7 Si intende gestire mediante un programma Java i vagoni che compongono un treno. Per ogni vagone si hanno alcuni attributi fondamentali:

- codice;
- peso a vuoto;
- azienda costruttrice;
- anno di costruzione.

Per i vagoni passeggeri si devono inoltre memorizzare:

- classe;
- numero di posti disponibili;
- numero di posti occupati;

mentre per i vagoni merci si devono memorizzare:

- volume di carico;
- peso massimo di carico;
- peso effettivo di carico.

Per la composizione di un treno è fondamentale la gestione del peso dei vagoni, che nel caso dei carri merci è di immediata determinazione, mentre per le carrozze passeggeri deve essere stimato considerando un peso medio per occupante di 65 kg (valore che potrebbe essere necessario modificare).
Dopo avere disegnato il diagramma UML delle classi della soluzione proposta e averlo implementato in linguaggio Java, codificare una classe Java *Treno* con uno o più metodi per l'aggiunta di vagoni: la classe dovrà prevedere un metodo che restituisca il peso complessivo del treno esclusa/e la/e motrice/i.

38. Si vogliono gestire le contravvenzioni effettuate dai vari comandi dei vigili di un comune. Di ogni contravvenzione interessa il veicolo a cui è stata effettuata, il vigile, il numero di verbale e il luogo in cui è stata emessa. Di ogni vigile interessa il nome, il cognome ed il numero di matricola. Di ogni comando interessa il nome della caserma, l'indirizzo, il numero di telefono e i vigili che fanno parte di esso. Di ogni veicolo interessa la targa. Esistono solamente due categorie di veicoli: automobili e motocicli. Delle automobili interessa la potenza in kilowatt, dei motocicli il numero di telaio. Scrivere in Java il codice della classe che permette l'inserimento di una nuova contravvenzione e quella relativa ad un Autoveicolo.
39. Le officine riparano i veicoli. Di ogni officina interessano nome, indirizzo, numero dipendenti, dipendenti (con l'informazione su quanti anni di servizio), e direttore. Dei dipendenti e dei direttori interessano: codice fiscale, l'indirizzo, numero telefono. Dei direttori interessa anche l'età. Delle riparazioni interessano: codice, veicolo, ora e data di accettazione, e ora e data di riconsegna. Dei veicoli interessano: modello, tipo, targa, anno di immatricolazione, e proprietario. Dei proprietari interessa codice fiscale, indirizzo, numero telefono.

Inner Class (composizione)

40. Nell'ambito di un programma di geometria, si implementi la classe Triangolo, il cui costruttore accetta le misure dei tre lati. Se tali misure non danno luogo ad un triangolo, il costruttore deve lanciare un'eccezione. Il metodo getArea restituisce l'area di questo triangolo. Si implementino anche la classe Triangolo.Rettangolo, il cui costruttore accetta le misure dei due cateti, e la classe Triangolo.Isoscele, il cui costruttore accetta le misure della base e di uno degli altri lati.
    Si ricordi che:
    - Tre numeri a, b e c possono essere i lati di un triangolo a patto che a<b+c, b<a+c e c<a+b.
    - L'area di un triangolo di lati a, b e c è data da: (formula di Erone), dove p è il semiperimetro

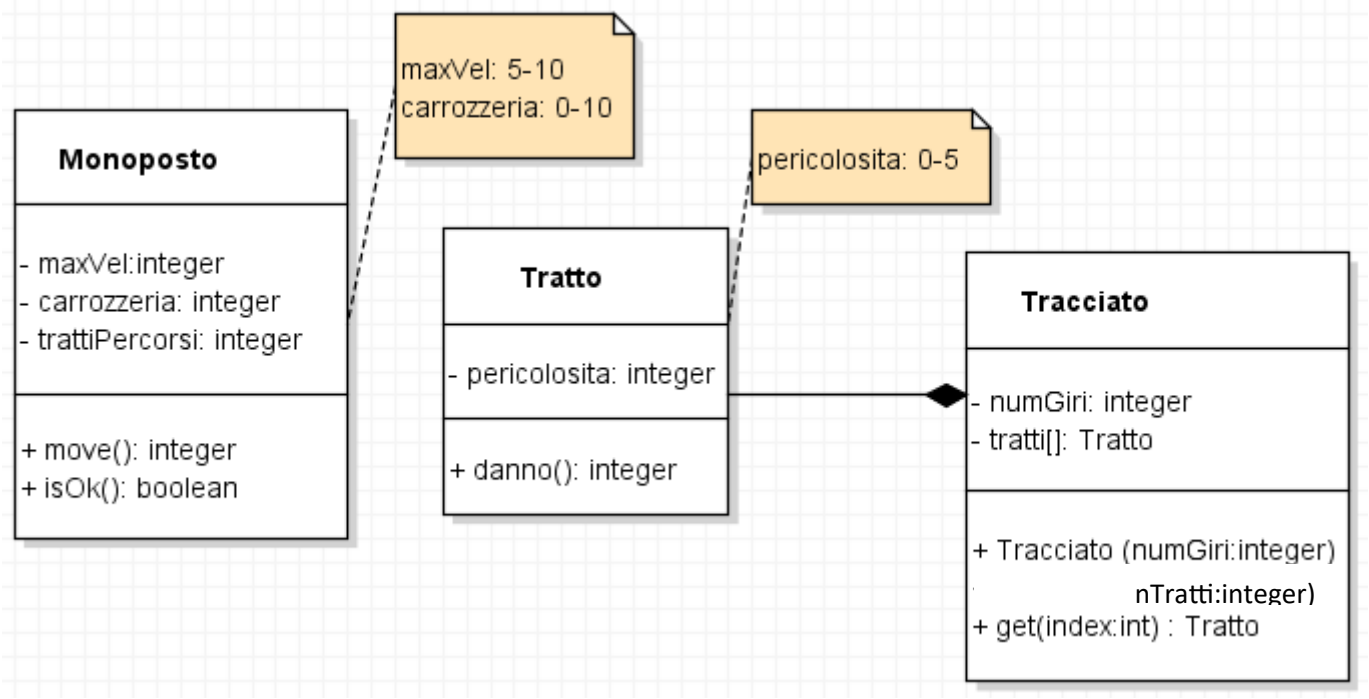
```
Triangolo x = new Triangolo(10,20,25);
Triangolo y = new Triangolo.Rettangolo(5,8);
Triangolo z = new Triangolo.Isoscele(6,5);

System.out.println(x.getArea());
System.out.println(y.getArea());
System.out.println(z.getArea());
```

Output dell'esempio d'uso:

```
94.9918
19.9999
12.0
```

41. Creare le classi rappresentate dal seguente diagramma UML per realizzare una gara di automobili:

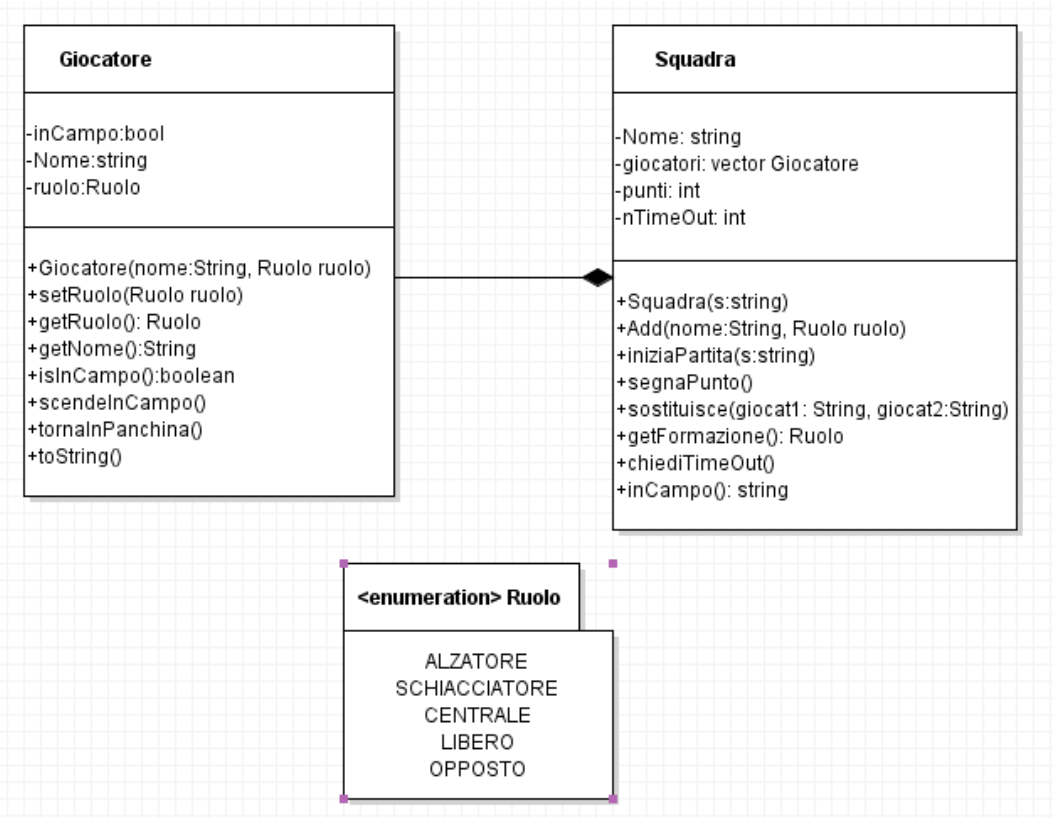


- Il metodo costruttore delle monoposto inizializza in modo casuale i parametri carrozzeria e maxVelocita
- Il metodo move() fa avanzare di un numero n casuale rispetto alla velocità max possibile la monoposto, facendole percorrere n tratti, in ciascuno dei quali può ricevere un danno
- Il metodo isOk() dice se l'automobile è ancora "viva" (attributo carrozzeria)
- Il costruttore di tratto imposta in modo casuale la pericolosità e il metodo danno() restituisce un possibile danno nel range stabilito dalla pericolosità
- il costruttore di Tracciato crea un percorso formato da nTratti aggiunti con valori casuali di pericolosità
- Realizza la classe Main che simula la gara tra due automobili

42. Scrivere una programma java per rappresentare delle cisterne. Per ciascuna cisterna si deve rappresentare un identificativo numerico univoco per la cisterna, una capacità massima e lo stato attuale della cisterna (quantità di materiale contenuto nella cisterna). Si devono quindi realizzare i seguenti metodi:
    - `Cisterna(int identificativo, int capacita)` costruttore con due argomenti pone lo stato della cisterna a 0.
    - `Cisterna(int identificativo, int capacita, int stato)` costruttore con tre argomenti.
    - `int getId()` che ritorna l'identificativo della cisterna.
    - `int getCapacita()` che ritorna la capacità massima della cisterna.
    - `int getStato()` che ritorna lo stato attuale della cisterna.
    - `boolean aggiungiValore(int quantita)` che aggiunge il valore quantità allo stato della cisterna. Se lo stato eccede la capacità lo stato deve essere posto pari alla capacità della cisterna e viene sollevata un'eccezione indicante cisterna piena ed il metodo ritorna il valore false. Altrimenti il metodo ritorna il valore true.
    - `int togliValore(int quantita)` che preleva il valore quantità dallo stato della cisterna e restituisce lo stato attuale. Se lo stato risulta minore di quantità il metodo restituisce come valore lo stato attuale = 0 e e viene sollevata un'eccezione indicante cisterna vuota.
    - `boolean equivalente(Cisterna c)` che ritorna true se l'oggetto di invocazione è uguale alla cisterna c. Due cisterne sono equivalenti se hanno la stessa capacità residua: capacità massima - stato attuale.

    Si realizzi inoltre una classe di massimo 10 cisterne che ne permetta l'inserimento di cisterne con identificatore diverso, la modifica della cisterna richiesta e la classe `Inizio` con un menu per verificare il corretto funzionamento di tutte le funzioni
43. Una rubrica contiene informazioni (nome, indirizzo,numero telefonico) su un certo numero di persone (per esempio 5) prestabilito (le informazioni sono reintrodotte nel metodo main()). L'utente dovrà fornire all'applicazione un nome da riga di comando e l'applicazione dovrà restituire le informazioni relative alla persona. Se il nome non è fornito, o se il nome immesso non corrisponde al nome di una persona preintrodotta dall'applicazione, deve essere restituito un messaggio significativo.

44. Realizzare un progetto Java che implementi la classe Giocatore e la classe Squadra (di pallavolo) descritti dall'UML sottostante



    - il costruttore della squadra riceve il Nome della squadra e i suoi componenti con i relativi ruoli sotto forma di stringa divisa da "," (Esempio: Nome: Grizzlies, Componenti squadra: Dentello,Donati,Bagnis,Ribero,Nazari, Peirone)
    - iniziaPartita() riceve la stringa con il nome dei giocatori che mette in campo e azzera i punti
    - segnaPunto() la squadra vince un punto durante una partita;
    - sostituisce() la squadra sostituisce un componente con un un'altra persona;
    - chiediTimeOut()la squadra richiede un time-out (al massimo due);

    Il programma principale deve prevedere di prendere in input due squadre che si affrontano nella partita, complete di nome e componenti. Successivamente deve iniziare un ciclo nel quale chiede all'utente l'evento che capita – i possibili eventi sono:
    - P – una squadra segna un punto – il programma deve chiedere quale squadra ha vinto il punto prendendo i input un numero tra 1 e 2;
    - S – una squadra sostituisce un componente – il programma deve chiedere quale squadra ha chiesto la sostituzione, la persona da sostituire e quella che la sostituisce – le stringhe devono essere nella forma "Dentello-Cerato";
    - T – una squadra chiede un time-out – il programma deve chiedere quale squadra chiede il time-out – deve visualizzare se viene accettato o rifiutato;

    Il programma finisce quando una delle due squadre raggiunge i 25 punti e ha più di due punti di vantaggio sull'avversario.
45. Un giocatore è caratterizzato da un nome ed un età. Un giocatore di pallavolo è un giocatore che in più è caratterizzato da un ruolo e da un valore espresso da un punteggio. Si devono quindi realizzare i seguenti metodi:
    - `Giocatore(String nome, int eta)` costruttore con due argomenti.
    - `GiocatorePallavolo(String nome, int eta, double punteggio, Ruolo ruolo)` costruttore
    - `String getNome()` che ritorna il nome del giocatore.
    - `double getPunteggio()` che ritorna il punteggio del giocatore.
    - `void setPunteggio(double nuovoPunteggio)` che pone il punteggio del giocatore a nuovoPunteggio.
    - `int getEta()` che ritorna l'età del giocatore.
    - `String toString()` che restituisce tutti i valori degli attributi separati da ';'
    - `boolean isMigliore(GiocatorePallavolo g)` che ritorna true se l'oggetto di invocazione è migliore del giocatore g passato come parametro, false altrimenti. Un giocatore è migliore di un altro se ha un punteggio più alto. A parità di punteggio il giocatore con l'età più bassa è considerato migliore.

    Si realizzi inoltre una classe `Squadra` in cui si caricano i giocatori del torneo in un vettore di massimo 15 (in ogni momento sono caricati `nGiocatori`). Questa classe ha anche i metodi:

- `addGiocatore(String nome, int eta, double punteggio, Ruolo ruolo)` che aggiunge un nuovo giocatore.
- `double getPunteggioMedio()` che restituisce il punteggio medio dei giocatori della squadra
- `passatoAnno()` che incrementa di 1 tutte le eta dei giocatori

Nel main creare un programma che testi le varie funzionalità tra cui l'elezione del migliore giocatore del torneo.