|  |
| --- |
| Programmazione ad Oggetti: JAVA  Esercitazione  Docente  Ing. Mariateresa Celardo [mariateresa.celardo@uniroma2.it](mailto:mariateresa.celardo@uniroma2.it) |

|  |
| --- |
| * Scrivere la classe **Motorino** che ha i seguenti attributi **colore**: una stringa indicante il colore del motorino, **velocità**: un numero con la virgola indicante la velocità in Km/h che possiede il motorino, **tipo**: una stringa indicante la marca e il modello del motorino es. “Piaggio scarabeo”, l’attributo **antifurto** un boolean che indica se è stato inserito l’antifurto (ha un valore iniziale pari a false). Il costruttore ha come parametri una stringa per il colore, una stringa per il tipo, un numero con la virgola per la velocità ed assegna opportunamente i valori dei parametri agli attributi. Scrivere il metodo **getVelocità** che restituisce la velocità del motorino, scrivere inoltre il metodo **accelera** che ha come parametro un numero con la virgola indicante i Km/h che si vogliono aggiungere alla velocità, il metodo verifica il valore dell’attributo **antifurto** se è false aggiunge il valore del parametro all’attributo velocità, altrimenti non fa nulla. Scrivere il metodo **inserisciAntifurto** che assegna un valore true all’attributo **antifurto**. |

• Scrivere la classe **Motorino** che ha i seguenti attributi **colore**: una stringa indicante il colore del motorino, **velocità**: un numero con la virgola indicante la velocità in Km/h che possiede il motorino, **tipo**: una stringa indicante la marca e il modello del motorino es. “Piaggio scarabeo”, l’attributo **antifurto** un boolean che indica se è stato inserito l’antifurto (ha un valore iniziale pari a false).

|  |
| --- |
| * Scrivere la classe **Motorino** che ha i seguenti attributi **colore**: una **stringa** indicante il colore del motorino, **velocità**: un **numero con la virgola** indicante la velocità in Km/h che possiede il motorino, **tipo**: una **stringa** indicante la marca e il modello del motorino es. “Piaggio scarabeo”, l’attributo **antifurto** un **boolean** che indica se è stato inserito l’antifurto (ha un valore iniziale pari a false).   **class** Motorino { **String colore**; **float velocità; String tipo;**  **boolean antifurto** =**false**;  } |

|  |
| --- |
| * Il costruttore ha come parametri una stringa per il colore, una stringa per il tipo, un numero con la virgola per la velocità ed assegna opportunamente i valori dei parametri agli attributi.   Motorino(**String** c, **String** t, **float** x){  **colore=c**; **velocità**=**x**; **tipo**=**t**;  } |

**float getVelocità**(){

}

|  |  |
| --- | --- |
| * Il costruttore ha come parametri una stringa per il colore, una stringa per il tipo, un numero con la virgola per la velocità ed assegna opportunamente i valori dei parametri agli attributi. | |
|  | Alternativamente il costruttore poteva essere: Motorino(**String** colore, **String** tipo, **float** velocità){  this.**colore=**colore; this.**velocità**=velocità; this.**tipo**=tipo;  } |
|  | |

|  |
| --- |
| * Scrivere il metodo **getVelocità** che restituisce la velocità del motorino   Il metodo restituisce la velocità occorre capire il tipo di dato per tale attributo. Nell’esercizio:  Scrivere la classe **Motorino** che ha i seguenti attributi:  …. **velocità**: un **numero con la virgola** indicante la velocità in Km/h che possiede il motorino  **ho definito l’attributo come float velocità;**  **float getVelocità**(){ **return velocità**;  } |

• Scrivere inoltre il metodo **accelera** che ha come

parametro un numero con la virgola indicante i Km/h che si vogliono aggiungere alla velocità, il metodo verifica il valore dell’attributo **antifurto** se è false aggiunge il valore del parametro all’attributo velocità, altrimenti non fa nulla.

**void** accelera(**float x**){

}

• Scrivere inoltre il metodo **accelera** che ha come

parametro un numero con la virgola indicante i Km/h che si vogliono aggiungere alla velocità,

|  |
| --- |
| * Scrivere inoltre il metodo **accelera** che ha come parametro un numero con la virgola indicante i Km/h che si vogliono aggiungere alla velocità, il metodo verifica il valore dell’attributo **antifurto** se è false aggiunge il valore del parametro all’attributo velocità, altrimenti non fa nulla.   **void** accelera(**float x**){  if(antifurto == false) {velocità = velocità+x; }  } |

|  |
| --- |
| * Scrivere il metodo **inserisciAntifurto** che assegna un valore true all’attributo **antifurto**.   **void** inserisciAntifurto(){  **antifurto**=**true**;  } |

|  |
| --- |
| class Motorino { String colore; float velocità; String tipo;  boolean antifurto =false; Motorino(String c, String t, float x){ colore=c;  velocità=x; tipo=t;  }  float getVelocità(){ return velocità;  }  void accelera(float x){  if(antifurto == false) velocità = velocità+x;  }  void inserisciAntifurto(){ antifurto=true;  }  } |

|  |
| --- |
| * Scrivere la classe **MotorinoImmatricolato** sottoclasse della classe Motorino che ha in più 2 attributi: **maxVelocità** un numero con la virgola (coerente con la scelta fatta per l’attributo velocità) indicante la velocità massima in Km/h che il motorino può raggiungere; **targa** una stringa indicante la targa del motorino (ad entrambi gli attributi viene assegnato un valore nel costruttore). Aggiungere il metodo **getMax** il metodo stampa il valore dell’attributo maxVelocità. Ridefinire il metodo **accelera** in modo che prima di modificare la velocità effettui un controllo sulla velocità massima raggiunta. Il metodo definisce una variabile s (dello stesso tipo di velocità) ed assegna ad s la somma tra il valore del parametro del metodo ed il valore dell’attributo velocità; se s è minore del valore dell’attributo maxVelocità assegna il valore di s all’attributo velocità altrimenti assegna all’attributo velocità il valore dell’attributo maxVelocità. |

**class** MotorinoImmatricolato extends Motorino {

**class** MotorinoImmatricolato extends Motorino {

**float maxVelocità**;

|  |
| --- |
| * Scrivere la classe **MotorinoImmatricolato**   sottoclasse della classe Motorino  **class** MotorinoImmatricolato extends Motorino {  } |

|  |
| --- |
| * Scrivere la classe **MotorinoImmatricolato** sottoclasse della classe Motorino che ha in più 2 attributi:   **maxVelocità** un numero con la virgola (coerente con la scelta fatta per l’attributo velocità) indicante la velocità massima in Km/h che il motorino può raggiungere;  **targa** una stringa indicante la targa del motorino (a entrambi gli attributi viene assegnato un valore nel costruttore).  **class** MotorinoImmatricolato extends Motorino {  **float maxVelocità; String targa;**  } |

|  |
| --- |
| La sottoclasse non eredita il costruttore!   * Nel testo: (a entrambi gli attributi viene assegnato un valore nel costruttore).   Motorino(**String** c, **String** t, **float** x){  Costruttore **colore=c**;  superclasse **velocità**=**x**;  **tipo**=**t**;  }  MotorinoImmatricolato(**String** c, **String** t, **float** x,  **float y, String targa**){ **super(c,t,x); maxVelocità = y; this.targa = targa;**  } //costruttore sottoclasse |

|  |
| --- |
| * Aggiungere il metodo **getMax** il metodo stampa il valore dell’attributo maxVelocità.   **void getMax(){ System.out.println(maxVelocità);**  **}** |

|  |
| --- |
| * Ridefinire il metodo **accelera** in modo che prima di modificare la velocità effettui un controllo sulla velocità massima raggiunta.   Metodo della superclasse Motorino  **void** accelera(**float x**){  if(antifurto == false) velocità = velocità+x;  } |

|  |
| --- |
| * Il metodo definisce una variabile s (dello stesso tipo di velocità) ed assegna ad s la somma tra il valore del parametro del metodo e il valore dell’attributo velocità; se s è minore del valore dell’attributo maxVelocità assegna il valore di s all’attributo velocità altrimenti assegna all’attributo velocità il valore dell’attributo maxVelocità.   **void accelera(float x){ float s=velocità+x;** if(antifurto == false) **{**  **if(s<=maxVelocità) velocità=s; else velocità=maxVelocità;** }  } |

|  |
| --- |
| class MotorinoImmatricolato extends Motorino { float maxVelocità;  String targa;  MotorinoImmatricolato(String c, String t, float x, float y, String targa){  super(c,t,x); maxVelocità = y; this.targa = targa;  }  void getMax(){ System.out.println(maxVelocità);  }  void accelera(float x) { float s=velocità+x; if(antifurto == false)  if(s<=maxVelocità) velocità=s; else velocità=maxVelocità;  }  } |

|  |
| --- |
| * Scrivere la classe **UsoMotorino** che ha il metodo main in cui si istanzia un oggetto di tipo Motorino di colore *grigiometallizzato* con velocità *40,5* e tipo *Piaggio Liberty* , richiamare il metodo getVelocità memorizzandone il valore in una variabile. Istanziare un altro oggetto di tipo MotorinoImmatricolato con colore *rosso*, velocità *30,5* tipo *Aprilia Scarabeo*, maxvelocità *60* targa *CV1234*, e richiamare il metodo getMax. Infine richiamare il metodo accelera con parametro pari a *30,7* per entrambi gli oggetti e quindi richiamare il metodo getVelocità per entrambi gli oggetti stampando i due valori. |

|  |
| --- |
| * Scrivere la classe **UsoMotorino** che ha il metodo main   class UsoMotorino {  public static void main( String args[]){  }  } |

|  |
| --- |
| * Scrivere la classe **UsoMotorino** che ha il metodo main in cui si istanzia un oggetto di tipo Motorino di colore *grigiometallizzato* con velocità *40,5* e tipo *Piaggio Liberty* |
| **Motorino m;**  **m= new Motorino("grigiometallizzato","Piagio Liberty",40.5);** |
|  |

|  |  |  |
| --- | --- | --- |
|  | | |
| **Motorino m;**  **m= new Motorino("grigiometallizzato","Piagio Liberty",40.5);** | | |
| Scrivendo in tal modo ho un errore | | |
| C:\esercizi>javac UsoMotorino.java UsoMotorino.java:5: cannot resolve symbol  symbol : constructor Motorino (java.lang.String,java.lang.String,**double**) location: class Motorino  m= new Motorino("grigiometallizzato","Piagio Liberty",40.50);  ^ | | |
| Il compilatore avvisa che nella classe usoMotorino trova la chiamata ad un costruttore che vuole come parametri una stringa,una stringa, un double (40.5) ma non esiste | | |
| il costruttore  è stato definito | Motorino(String c, String t, **float** x){ colore=c;  velocità=x; tipo=t;  } |  |

|  |
| --- |
| * Scrivere la classe **UsoMotorino** che ha il metodo main in cui si istanzia un oggetto di tipo Motorino di colore *grigiometallizzato* con velocità *40,5* e tipo *Piaggio Liberty* |
| class UsoMotorino {  public static void main( String args[]){  **Motorino m;**  **m= new Motorino("grigiometallizzato","Piagio Liberty",40.5f)**  }  } |
|  |

|  |
| --- |
| * Scrivere la classe **UsoMotorino** che ha il metodo main in cui si istanzia un oggetto di tipo Motorino di colore *grigiometallizzato* con velocità *40,5* e tipo *Piaggio Liberty* , richiamare il metodo getVelocità memorizzandone il valore in una variabile. |
| class UsoMotorino {  public static void main( String args[]){  **Motorino m;**  **m= new Motorino("grigiometallizzato","Piagio Liberty",40.5f)**  float x = m.getVelocità();  }  } |
|  |

|  |
| --- |
| * Istanziare un altro oggetto di tipo MotorinoImmatricolato con colore *rosso*, velocità *30,5* tipo *Aprilia Scarabeo*, maxvelocità *60* targa *CV1234*,   **MotorinoImmatricolato mI;**  **mI= new MotorinoImmatricolato(“rosso",“Aprilia Scarabeo",30.5f, 60.0f,”CV1234”);** |

|  |
| --- |
| * richiamare il metodo getMax   Il metodo nella classe è definito void getMax(){  System.out.println(maxVelocità);  }  **mI.getMax();** |

|  |
| --- |
| * Infine richiamare il metodo accelera con parametro pari a *30,7* per entrambi gli oggetti e quindi richiamare il metodo getVelocità per entrambi gli oggetti stampando i due valori.   **mI.accelera(30.7f); m.accelera(30.7f); System.out.println(mI.getVelocità()); System.out.println(m.getVelocità());** |

|  |
| --- |
| class UsoMotorino {  public static void main( String args[]){ Motorino m;  m= new Motorino("grigiometallizzato","Piagio Liberty",40.5f); float x = m.getVelocità();  MotorinoImmatricolato mI;  mI= new Motorino(“rosso",“Aprilia Scarabeo",30.5f, 60.0f,”CV1234”);  mI.getMax(); mI.accelera(30.7f); m.accelera(30.7f);  System.out.println(mI.getVelocità()); System.out.println(m.getVelocità());  }  } |

|  |
| --- |
| Scrivere la classe **Dipendente** che ha i seguenti attributi **matricola**: una stringa indicante il numero di matricola del dipendente, **stipendio**: un numero con la virgola indicante lo stipendio base che possiede il dipendente, **straordinario**: un numero con la virgola indicante l’importo dovuto per ciascuna ora di straordinario effettuata dal dipendente.  Il costruttore ha come parametri una stringa per la matricola, un numero con la virgola per lo stipendio ed un numero con la virgola per lo straordinario ed assegna opportunamente i valori dei parametri agli attributi.  Scrivere il metodo **getStipendio** che restituisce il valore dell’attributo stipendio, scrivere inoltre il metodo **paga** che ha come parametro un numero intero indicante il numero di ore di straordinario effettuate dal dipendente, il metodo restituisce il valore ottenuto sommando all’attributo stipendio il risultato del prodotto tra il parametro del metodo e l’attributo straordinario. Scrivere il metodo **stampa** che stampa il valore degli attributi della classe |

|  |
| --- |
| Scrivere la classe **Dipendente** che ha i seguenti attributi **matricola**: una stringa indicante il numero di matricola del dipendente, **stipendio**: un numero con la virgola indicante lo stipendio base che possiede il dipendente, **straordinario**: un numero con la virgola indicante l’importo dovuto per ciascuna ora di straordinario effettuata dal dipendente.  class Dipendente{ String matricola;  double stipendio, straordinario; |

|  |
| --- |
| Il costruttore ha come parametri una stringa per la matricola, un numero con la virgola per lo stipendio ed un numero con la virgola per lo straordinario ed assegna opportunamente i valori dei parametri agli attributi.  Dipendente(String m, double x, double y){ matricola=m;  stipendio=x; straordinario=y;  } |

|  |
| --- |
| Scrivere il metodo **getStipendio** che restituisce il valore dell’attributo stipendio, scrivere inoltre il metodo **paga** che ha come parametro un numero intero indicante il numero di ore di straordinario effettuate dal dipendente, il metodo restituisce il valore ottenuto sommando all’attributo stipendio il risultato del prodotto tra il parametro del metodo e l’attributo straordinario.  **double getStipendio(){ return stipendio;**  **}**  **double paga(int x){**  **return stipendio+x\*straordinario;**  **}** |

|  |
| --- |
| Scrivere il metodo **stampa** che stampa il valore degli attributi della classe |
| void stampa(){  System.out.println("matricola ="+matricola+" stipendio ="+ stipendio +" straordinario= "+straordinario);  /\* alternativamente  System.out.println(matricola+" "+ stipendio +" "+straordinario);\*/  } |
|  |

|  |
| --- |
| class Dipendente{ String matricola;  double stipendio, straordinario; Dipendente(String m, double x, double y){ matricola=m;  stipendio=x; straordinario=y;  }  double getStipendio(){ return stipendio;  }  double paga(int x){  return stipendio+x\*straordinario;  }  void stampa(){  System.out.println("matricola ="+matricola+" stipendio ="+ stipendio +" straordinario= "+straordinario);  }  } |

|  |
| --- |
| Scrivere la sottoclasse **DipendenteA** della classe Dipendente che ha in più l’attributo **malattia**: un numero intero indicante i giorni di malattia presi che viene inizializzato a zero. Aggiungere il metodo **prendiMalattia** che ha come parametro un numero intero indicante i giorni di malattia presi, il metodo modifica il valore dell’attributo malattia aggiungendogli il valore del parametro. Ridefinire il metodo **paga** in modo che prima definisce una variabile **p** a cui assegna il valore ottenuto richiamando il metodo paga della superclasse, poi se l’attributo malattia è uguale a zero il metodo restituisce il valore di **p** altrimenti restituisce il valore ottenuto sottraendo a **p** il valore del prodotto tra malattia e 15,0. Definire un metodo **stampaMalattia** che stampa il valore dell’attributo malattia |

|  |
| --- |
| Scrivere la sottoclasse **DipendenteA** della classe Dipendente che ha in più l’attributo **malattia**: un numero intero indicante i giorni di malattia presi che viene inizializzato a zero.  class DipendenteA extends Dipendente{ int malattia;  DipendenteA(String m, double x, double y){ super(m,x,y);  malattia=0;  } |

|  |
| --- |
| Aggiungere il metodo **prendiMalattia** che ha come parametro un numero intero indicante i giorni di malattia presi, il metodo modifica il valore dell’attributo malattia aggiungendogli il valore del parametro.  void prendiMalattia(int x){ malattia=malattia+x;  }  **Alternativamente:**  void prendiMalattia(int x){ malattia+=x;  } |

|  |
| --- |
| Ridefinire il metodo **paga** in modo che prima definisce una variabile **p** a cui assegna il valore ottenuto richiamando il metodo paga della superclasse, poi se l’attributo malattia è uguale a zero il metodo restituisce il valore di **p** altrimenti restituisce il valore ottenuto sottraendo a **p** il valore del prodotto tra malattia e 15,0.  **double paga(int x){**  **double p= super.paga(x); if (malattia==0) return p;**  **else return p-malattia\*15.0;**  **}** |

|  |
| --- |
| Definire un metodo **stampaMalattia** che stampa il valore dell’attributo malattia  void stampaMalattia(){ System.out.println("malattia ="+malattia);  }  **Alternativamente**  void stampaMalattia(){ System.out.println(malattia);  } |

|  |
| --- |
| class DipendenteA extends Dipendente{ int malattia;  DipendenteA(String m, double x, double y){ super(m,x,y);  malattia=0;  }  void prendiMalattia(int x){ malattia+=x;  }  double paga(int x){ double p= super.paga(x); if (malattia==0) return p;  else return p-malattia\*15.0;  }  void stampaMalattia(){ System.out.println("malattia ="+malattia);  }  } |

|  |
| --- |
| Scrivere la classe **CalcolaStipendi** che ha il metodo main in cui istanziare un oggetto di tipo **Dipendente** con matricola *00309* stipendio *1000,00* e straordinario *7,50*. Richiamare il metodo **paga** con parametro pari a *10*, stampare il valore dello stipendio di tale oggetto richiamando **getStipendio**. Istanziare un altro oggetto di tipo **DipendeteA** con matricola *00201* stipendio *1500,0* e straordinario *8,50*. Richiamare il metodo **prendiMalattia** con parametro pari a *5*, richiamare il metodo **paga** con parametro pari a *3* ed il metodo **stampaMalattia**. |

|  |
| --- |
| Scrivere la classe **CalcolaStipendi** che ha il metodo main in cui istanziare un oggetto di tipo **Dipendente** con matricola *00309* stipendio *1000,00* e straordinario *7,50*. Richiamare il metodo **paga** con parametro pari a *10*, stampare il valore dello stipendio di tale oggetto richiamando **getStipendio**.  class CalcolaStipendi {  public static void main( String args[]){ Dipendente a;  a= new Dipendente("00309",1000.00,7.50); System.out.println(a.paga(10)); System.out.println(a.getStipendio()); |

|  |
| --- |
| Istanziare un altro oggetto di tipo **DipendeteA** con matricola *00201* stipendio *1500,0* e straordinario *8,50*. Richiamare il metodo **prendiMalattia** con parametro pari a *5*, richiamare il metodo **paga** con parametro pari a *3* ed il metodo **stampaMalattia**.  DipendenteA a1;  a1= new DipendenteA("00201",1500.00,8.50); a1.prendiMalattia(5); System.out.println(a1.paga(3)); a1.stampaMalattia(); |

|  |
| --- |
| class CalcolaStipendi  {  public static void main( String args[]){ Dipendente a;  a= new Dipendente("00309",1000.00,7.50); System.out.println(a.paga(10)); System.out.println(a.getStipendio());  DipendenteA a1;  a1= new DipendenteA("00201",1500.00,8.50); a1.prendiMalattia(5); System.out.println(a1.paga(3)); a1.stampaMalattia();  }  } |

|  |
| --- |
| Scrivere la classe **TipoCapoAbigliamento** che ha i seguenti attributi **marca**:una stringa indicante la marca, **modello**: una stringa indicante il modello, **costo**:un numero con la virgola indicante il costo in euro. Il costruttore ha come parametri una stringa per la marca, una stringa per il modello, un numero con la virgola per il costo ed assegna opportunamente i valori dei parametri agli attributi. Scrivere il metodo **getCosto** che stampa il valore dell’attributo costo. Scrivere inoltre il metodo **applicaSconto** che ha come il parametro un numero con la virgola indicante la percentuale di sconto che si vuole applicare, il metodo restituisce il costo del capo di abbigliamento scontato ossia il risultato della sottrazione tra costo e costo moltiplicato il valore del parametro diviso 100. Scrivere il metodo **modificaCosto** che ha come parametro un numero con la virgola (dello stesso tipo di costo) indicante il nuovo costo ed assegna all’attributo costo il valore del parametro. |

|  |
| --- |
| class TipoCapoAbbigliamento{ String marca, modello;  double costo;  TipoCapoAbbigliamento(String marca, String modello, double costo){  this.marca=marca; this.modello=modello; this.costo=costo;  }  void getCosto(){ System.out.println(costo);  }  double applicaSconto(double x){ return costo-(costo\*x/100);  }  void modificaCosto(double x){ costo=x;  }  } |

|  |
| --- |
| Scrivere la classe **CapoAbbigliamento** sottoclasse della classe **TipoCapoAbbigliamento** che ha in più 3 attributi: **taglia** un numero intero che indica la taglia; **quantitàAcquistata** un numero intero indicante il numero di pezzi acquistati, **quantitàDisponibile** un numero intero indicante il numero di pezzi disponibili (agli attributi viene assegnato un valore del costruttore). Scrivere il metodo **venduto** avente come parametro un numero intero indicante il numero di capi venduti, il metodo se **quantitàDisponibile** è maggiore del risultato della sottrazione tra quantitàDisponibile ed il parametro, modifica il valore dell’attributo **quantitàDisponibile** sottraendogli il valore del parametro. Scrivere il metodo **getDettagli** che stampa tutti gli attributi della classe. |

|  |
| --- |
| class CapoAbbigliamento extends TipoCapoAbbigliamento{ int taglia, quantitàAcquistata, quantitàDisponibile;  CapoAbbigliamento(String marca, String modello, double costo, int t,int a,int d){  super(marca,modello,costo); taglia=t; quantitàAcquistata=a; quantitàDisponibile=d;  }  void venduto(int x){  if(quantitàDisponibile>quantitàDisponibile-x) quantitàDisponibile=quantitàDisponibile-x;  }  void getDettagli(){ System.out.println("marca="+marca+"  modello="+modello+" costo="+costo+" taglia="+taglia+" quantitàAcquistata="+quantitàAcquistata+ "quantitàDisponibile="+quantitàDisponibile);  }  } |

|  |
| --- |
| Scrivere la classe **Negozio** che ha il metodo main in cui istanziare un oggetto di tipo **TipoCapoAbbigliamento** di marca *Colmar* modello *Giacca E45* con prezzo *350,5* e richiamare il metodo **getCosto** ed il metodo **applicaSconto** con parametro *30* stampando il valore restituito dal metodo. Istanziare un oggetto di tipo **CapoAbbigliamento** con marca *Levis* modello *jeans 570SC* costo *79,8* taglia *40*, quantità acquistata *30* quantità disponibile *30* e richiamare il metodo **venduto** con parametro pari a *2*. Infine richiamare il metodo **getDettagli**. |

|  |
| --- |
| class Negozio {  public static void main( String args[]){ TipoCapoAbbigliamento x;  x= new TipoCapoAbbigliamento("Colmar","Giacca E45",350.5);  System.out.println(x.applicaSconto(30));  CapoAbbigliamento y;  y= new CapoAbbigliamento("Levis","jeans 570SC",79.8,40,30,30);  y.venduto(2); y.getDettagli();  }  } |