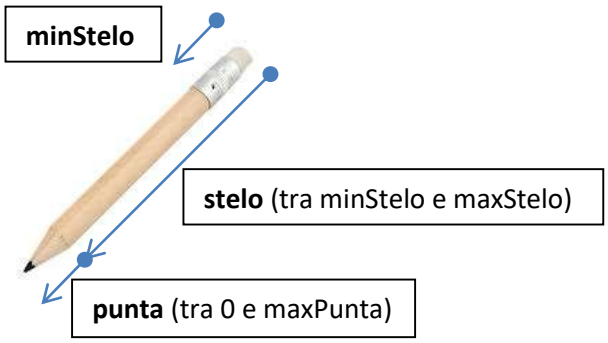
Esercitazione 01

Attributi e metodi statici e dinamici

In questa esercitazione introduciamo gli attributi statici. Sono indicati da ***public static tipo attributo*** oppure ***private static tipo attributo***. Un attributo statico non descrive un oggetto in particolare, ma l'intera classe. Qui "statico" significa ***"attributo che non fa parte di un oggetto della classe"***, dunque aggiunto alla memoria prima di iniziare l'esecuzione: **non** significa "costante". Invece, per dichiarare un attributo (qualsiasi) costante aggiungete "final": per es. ***public static final tipo attributo***. Per richiamare un attributo/metodo statico pubblico fuori dalla sua classe C scrivete ***C.attributo, C.metodo***. Dentro la classe C scrivete semplicemente ***attributo*** e ***metodo***.

La classe Matita. Vi chiediamo di scrivere una classe pubblica Matita per rappresentare virtualmente matite. Una matita è definita come uno stelo (una lunghezza intera in millimetri, da un minimo ***minStelo*** a un massimo ***maxStelo***) seguita da una punta (un intero da 0 a un massimo ***maxPunta***).

Fissate nella definizione della classe dei valori per massimi e minimi, per esempio: ***minStelo=10, maxStelo=200, maxPunta=5***.



(i) ***minStelo, maxStelo, maxPunta*** sono attributi interi ***pubblici***, ***statici*** e ***final*** della classe Matita (non legati a un oggetto ma alla classe). Invece stelo e punta sono attributi interi ***dinamici***.

(ii) Il costruttore di ***Matita*** consente di costruire una matita con punta di lunghezza massima dato lo stelo. Un assert impedisce lunghezze non accettabili dello stelo.

(iii) La classe ha i metodi ***get*** per stelo e punta e nessun metodo ***set***: non consento di cambiare la lunghezza a una matita.

(iv) Un metodo "disegna" restituisce "true" (successo) se la matita ha almeno 1mm di punta, e "false" (fallimento) altrimenti. Nel primo caso usa la matita fino a ridurne la punta di un 1mm.

(v) Un metodo "tempera" restituisce "true" (successo) se la matita è più lunga del minimo e "false" (fallimento) altrimenti. Nel primo caso riduce lo stelo di 1mm e allunga la punta di 1mm, a meno che la lunghezza della punta sia già il massimo. In questo caso la matita si accorcia ma la punta resta invariata.

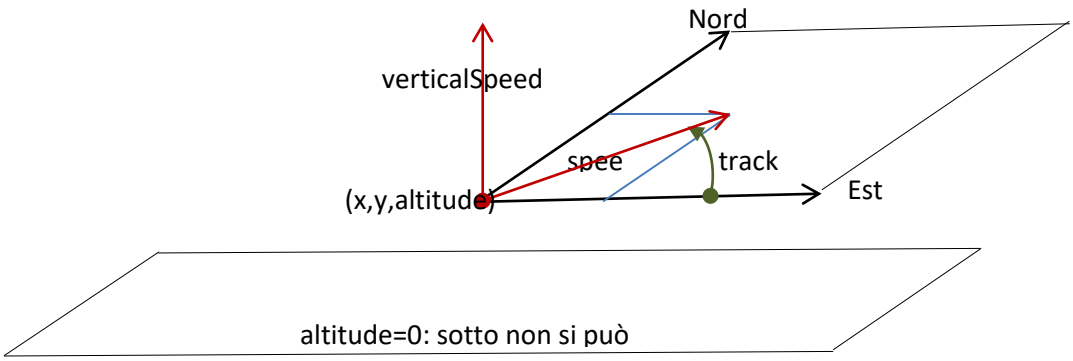
Scrivete ***Matita.maxStelo*** per richiamare il massimo dello stelo (attributo statico). Scrivete ***Math.min*** per richiamare il metodo statico min(x,y) della classe Math, che calcola il minimo. Includiamo una classe TestMatita per sperimentare la classe Matita: eseguitela e controllate che i risultati siano sensati.

```
//Una classe come test della classe Matita: salvate nel file
//TestMatita.java
```

```
//Non si compila senza la classe Matita

public class TestMatita {
 public static void main(String[] args){
  Matita m = new Matita(Matita.maxStelo);
  int s = m.getStelo(), p = m.getPunta();
  System.out.println("Matita di stelo " + s + " e punta " + p);
  System.out.println("Disegno per " + 2*p + " volte:");
  System.out.println("dopo " + p + " volte il disegno fallisce");
  for (int i = 0; i < 2*p; i++)
   System.out.println(" Successo disegno n."+i+" = "+m.disegna());
  System.out.println("Tempero di 1mm la matita"); m.tempera();
  System.out.println(" nuova lunghezza punta = " + m.getPunta());
  System.out.println(" nuova lunghezza stelo = " + m.getStelo());
  System.out.println("Stampo la matita m. Ottengo \"Matita@\" seguito
dall'indirizzo dell'oggetto (in esadecimale): " + m);
  }
}
```

La classe Elicottero. Vi chiediamo di scrivere una classe Elicottero per rappresentare virtualmente elicotteri. Un elicottero è definito con tre coordinate (intere, in km): ***x,y*** e ***altitude*** (non negativa), due velocità (intere, in hm/h), ***speed*** (orizzontale e non negativa) e ***verticalSpeed*** (verticale), e una direzione orizzontale ***track*** (un reale, un angolo in radianti tra 0 e 2π).



La classe ha i seguenti metodi. Usate degli assert per impedire valori non accettabili degli attributi.

(i) Il costruttore di Elicottero definisce un elicottero fermo in cielo, date le coordinate (x, y, altitude), con velocità nulle e angolo di direzione nullo.

(ii) La classe ha i metodi get per ogni attributo e metodi set per velocità e direzione, ma non per x, y, altitude. Non consentiamo a un elicottero di cambiare la posizione se non spostandosi con lo scorrere del tempo.

(iii) Un metodo ***void elapse(double time)*** modifica la posizione dell'elicottero dato il tempo trascorso, in base alle velocità e alla direzione, usando le formule della trigonometria. Quando assegnate il risultato a delle coordinate intere dovrete arrotondarlo, scrivendo: *(int) espressione*.

Per richiamare un attributo/metodo statico pubblico fuori dalla sua classe C scrivete ***C.attributo, C.metodo***. Per esempio scrivete ***Math.sin, Math.cos*** per i metodi statici per seno e coseno della classe Math. Includiamo una classe ***TestElicottero*** per sperimentare la classe Elicottero: eseguitela (richiede la classe Elicottero) e controllate che i risultati siano sensati. Pubblicheremo le soluzioni la prossima settimana.

```
//Una classe come test della classe Elicottero: salvate nel file
//TestElicottero.java. Non si compila senza la classe Elicottero

public class TestElicottero {

public static void main(String[] args) {
//creo un elicottero collocato all'origine del sistema di riferimento
 Elicottero a = new Elicottero(0, 0, 0);
// visualizzo le coordinate dell'elicottero
 System.out.println( "Elicottero a di coordinate" );
 System.out.println("("+a.getX()+","+a.getY()+","+
                   a.getAltitude()+")");
// imposto la velocita' dell'elicottero a 500 km/h
 a.setSpeed(500);
 // imposto la velocita' verticale dell'elicottero a +10 km/h
 a.setVerticalSpeed(10);
 // imposto la direzione dell'elicottero a nord-nord-est (in radianti)
 a.setTrack((double) (3 * Math.PI / 8));
 System.out.println("velocita'="+a.getSpeed()+" velocita' verticale="
                   + a.getVerticalSpeed()+" angolo="+a.getTrack());
 // faccio trascorrere mezz'ora
 a.elapse(0.5);
 // visualizzo le nuove coordinate: (95,230,5)
 System.out.println("Dopo mezz'ora di volo coordinate: ");
 System.out.println("("+a.getX()+","+a.getY()+","
                    +a.getAltitude()+")");
 }
}
```