

Esercizi sull'uso di classi e oggetti

(Fondamenti di Informatica – Emilio Di Giacomo)

Esercizio 1

La classe **Punto** modella oggetti che rappresentano punti nel piano. Un oggetto **Punto** è definito attraverso le sue coordinate. La classe **Punto** ha i seguenti costruttori e metodi:

```
/* costruttore: permette di creare un punto specificando le sue coordinate */  
public Punto (double x, double y)
```

```
/* restituisce la coordinata x del punto */  
public double coordX ()
```

```
/* restituisce la coordinata y del punto */  
public double coordY ()
```

```
/* restituisce il punto medio del segmento i cui estremi sono i punti dati */  
public static Punto puntoMedio (Punto p1, Punto p2)
```

La classe **Retta** modella oggetti che rappresentano rette nel piano. Un oggetto **Retta** è definito attraverso due suoi punti (istanze della classe **Punto**). La classe **Retta** ha i seguenti costruttori e metodi:

```
/* costruttore: permette di creare una retta passante per i due punti dati */  
public Retta (Punto p1, Punto p2)
```

```
/* restituisce un oggetto di tipo Retta che rappresenta la retta parallela a quella su cui è invocato il metodo e passante per il punto dato */  
public Retta parallela (Punto p)
```

```
/* restituisce un oggetto di tipo Retta che rappresenta la retta perpendicolare a quella su cui è invocato il metodo e passante per il punto dato */  
public Retta perpendicolare (Punto p)
```

```
/* restituisce un oggetto di tipo Punto che rappresenta il punto di intersezione tra la retta su cui è invocato il metodo e la retta passata come parametro */  
public Punto intersezione (Retta r)
```

Le classi **Punto** e **Retta** sono già interamente scritte e disponibili. Si scriva una classe **EsercizioParallelogramma** che utilizza le funzionalità delle classi **Punto** e **Retta**. La classe **EsercizioParallelogramma** avrà il solo metodo speciale **main**, il quale deve svolgere nell'ordine le seguenti azioni:

- Fa inserire all'utente le coordinate di tre diversi punti **p1**, **p2** e **p3**
- Calcola un punto **p4** che insieme a **p1**, **p2** e **p3** definisce un parallelogramma.
- Visualizza all'utente le coordinate del punto **p4**.

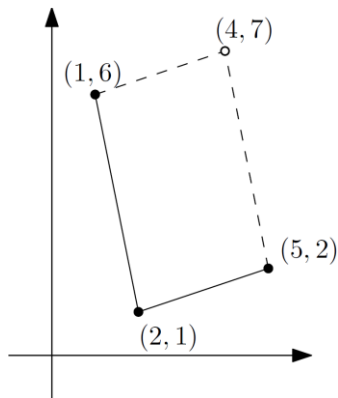


Figura 1. Esempio di input e output del programma EsercizioParallelogramma

Esercizio 2

Si chiama **ortocentro** di un triangolo il punto di intersezione delle tre altezze del triangolo. Si ricorda che un'**altezza** di un triangolo è un segmento che congiunge un vertice del triangolo con il lato opposto ed è perpendicolare a tale lato.

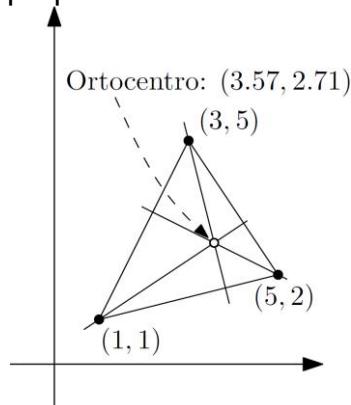


Figura 2. Ortocentro di un triangolo

Si chiama **baricentro** di un triangolo il punto di intersezione delle tre mediane del triangolo. Si ricorda che una **mediana** di un triangolo è un segmento che congiunge un vertice del triangolo con il punto medio del lato opposto.

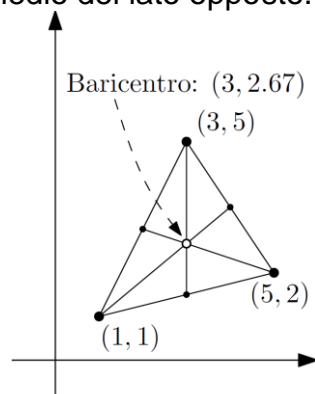


Figura 3. Baricentro di un triangolo

Si chiama **circoncentro** di un triangolo il punto di intersezione delle tre assi dei lati del triangolo. Si ricorda che l'asse di un segmento è la retta che passa per il punto medio del segmento ed è ad esso perpendicolare.

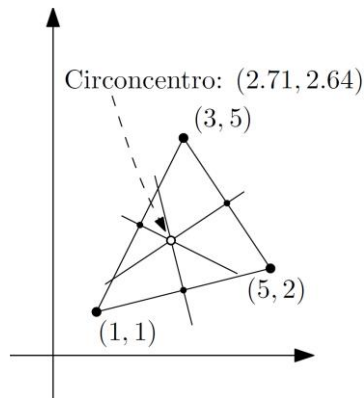


Figura 4. Circoncentro di un triangolo.

Si scriva una classe **EsercizioTriangolo** che utilizza le funzionalità delle classi **Punto** e **Retta**. La classe **EsercizioTriangolo** avrà il solo metodo speciale **main**, il quale deve svolgere nell'ordine le seguenti azioni:

- Fa inserire all'utente le coordinate di tre diversi punti **p1**, **p2** e **p3**
- Calcola l'ortocentro, il baricentro e il circoncentro del triangolo definito da **p1**, **p2** e **p3**.
- Visualizza all'utente le coordinate dei punti calcolati.

Esercizio 3

Nel package *fond.esempi.capitolo5* è presente una classe di nome **Cerchio**, i cui oggetti rappresentano cerchi nel piano. La classe dispone del seguente costruttore e dei seguenti metodi di istanza:

- **public Cerchio(double r):** costruttore per creare un oggetto **Cerchio** che rappresenta un cerchio di raggio **r**.
- **public double diametro():** metodo che restituisce il diametro del cerchio rappresentato dall'oggetto ricevente.
- **public double perimetro():** metodo che restituisce il perimetro del cerchio rappresentato dall'oggetto ricevente.
- **public double area():** metodo che restituisce l'area del cerchio rappresentato dall'oggetto ricevente.
- **public Cerchio sommaRaggio(Cerchio c):** metodo che restituisce un nuovo oggetto **Cerchio** il cui raggio è la somma dei raggi dei cerchi rappresentati dall'oggetto ricevente e dall'oggetto **c**.

Scrivere una classe di nome **UsoCerchio**, avente soltanto il metodo speciale **main**. Tale metodo deve svolgere queste azioni:

1. Fare inserire all'utente tramite tastiera i raggi **r1** e **r2** di due cerchi nel piano.
2. Creare due oggetti **Cerchio**, **c1** e **c2**, di raggio rispettivamente **r1** e **r2**.
3. Stampare sullo standard output il diametro, il perimetro e l'area dei cerchi rappresentati da **c1** e **c2**.
4. Creare un terzo oggetto **Cerchio c3**, che rappresenti un cerchio di raggio pari alla somma dei raggi dei cerchi rappresentati da **c1** e **c2**.
5. Stampare sullo standard output il diametro, il perimetro e l'area del cerchio rappresentato da **c3**.