```
class ProgCerchioNull {

  public static void main(String[] args) {

        Cerchio tavolo = new Cerchio (1);

        System.out.println ("\narea cerchio 1^ = " + tavolo.area ());
        System.out.println (    "perimetro    1^ = " + tavolo.perim());

        System.out.println ("\narea cerchio 1A^= " + tavolo.areaArrot ());
        System.out.println (    "perimetro    1A^= " + tavolo.perimArrot());

        tavolo.setRaggio (3.5);

        System.out.println ("\narea cerchio 2A^= " + tavolo.areaArrot ());
        System.out.println (    "perimetro    2A^= " + tavolo.perimArrot());

        tavolo = null;

        System.out.print   ("\narea cerchio 3A^= ");

        if (tavolo != null)
              System.out.println  (tavolo.areaArrot ());
        else
              System.out.println  ("\n ATTENZIONE!!! riferimento a NULL\n");

        System.out.println ("\narea cerchio 4A^= " + tavolo.areaArrot ());
  }
}
```

Il programma **ProgCerchioNull.java** dichiara ed istanzia l'**oggetto tavolo** di **classe Cerchio** di cui richiama **i metodi (pubblici):**

- **Cerchio (double r)** **COSTRUTTORE esplicito**
- **area ()**
- **perim()**
- **setRaggio (double r)**
- **areaArrot ()** con arrotondamento a 3 decimali
- **perimArrot()** con arrotondamento a 3 decimali

Espone area e perimetro per un cerchio di raggio 1 e poi, per lo stesso cerchio, area e perimetro arrotondati a 3 decimali.

Successivamente modifica il raggio al valore 3,5 richiamando il metodo **setRaggio (double r)** e ne espone area e perimetro arrotondati a 3 decimali.

Poi imposta **tavolo** a NULL quindi il riferimento **tavolo** non conterrà più l'indirizzo di un oggetto e l'istruzione **if** esporrà il messaggio di errore.

Infine richiama comunque un metodo della classe e l'interprete Java intercetta e segnala l' **exception**

```
Exception in thread "main" java.lang.NullPointerException
        at ProgCerchioNull.main(ProgCerchioNull.java:27)
```

Programmi con classe CERCHIO

```
class  Cerchio {
  private double raggio;
  private double area;
  private double perim;
  public Cerchio (double r) {          // --- COSTRUTTORE
        raggio = r;
        area   = calcArea();          //- richiamo metodo privato
        perim  = calcPerim();         //- richiamo metodo privato
  }
  public void setRaggio (double r) {
        raggio = r;
        area   = calcArea();
        perim  = calcPerim();
  }
  public double area () {
        return area;
  }
  public double perim () {
        return perim;
  }
  public double areaArrot () {
        return (arrotonda (area));    //- richiamo metodo privato
  }
  public double perimArrot () {
        return (arrotonda (perim));   //- richiamo petodo privato
  }
  private double calcArea() {
        return (raggio * raggio * Math.PI);
  }
  private double calcPerim() {
        return (2 * raggio * Math.PI);
  }
  private double arrotonda (double x) {
    double temp;
        temp = x * 1000;
        temp = Math.round (temp);    //--- intero
        temp = temp / 1000;          //--- double
        return temp;
  }
}
```

La classe Cerchio ha 3 attributi privati e 9 metodi:

- un **costruttore** (**pubblico**) che calcola anche gli attributi area e perimetro
- 5 metodi **pubblici** richiamabili dall'esterno della classe
- 3 metodi **privati** (utilizzabili solo dalla classe)

```
area cerchio 1^ = 3.141592653589793
perimetro    1^ = 6.283185307179586

area cerchio 1A^= 3.142
perimetro    1A^= 6.283

area cerchio 2A^= 38.485
perimetro    2A^= 21.991

area cerchio 3A^=
 ATTENZIONE!!! riferimento a NULL

Exception in thread "main" java.lang.NullPointerException
        at ProgCerchio.main(ProgCerchio.java:27)


...Program finished with exit code 1
Press ENTER to exit console.
```

Il programma **ProgCerchioNull.java** serve solo a testare la classe **Cerchio**; il progetto completo deve acquisire il raggio in input e poi istanziare un oggetto della classe **Cerchio**

```
import java.io.*;
class ProgCerchio {
 public static void main(String[] args) {
   InputStreamReader input = new InputStreamReader(System.in);
   BufferedReader   tastiera = new BufferedReader(input);
   Cerchio tavolo, ruota;
   double raggio;
        System.out.println ("digita il raggio di un tavolo (in metri)");
        raggio = leggiDouble(tastiera);
        tavolo = new Cerchio (raggio);
        System.out.println ("\narea tavolo = " + tavolo.areaArrot ());
        System.out.println (  "perimetro   = "  + tavolo.perimArrot());

        System.out.println ("\ndigita il raggio di una ruota (in metri)");
        raggio = leggiDouble(tastiera);
        ruota = new Cerchio (raggio);
        System.out.println ("\narea ruota = " + ruota.areaArrot ());
        System.out.println (   "perimetro  = " + ruota.perimArrot());
 } //---------------------- fine main

 private static double leggiDouble (BufferedReader tast){
   double rr = 0;
   String leggiD;
     for (int k = 1; k <= 1 ; k++){
        System.out.print ("raggio ? ");
        try {
             leggiD = tast.readLine();
             rr =Double.valueOf(leggiD).doubleValue();
             if  (rr <= 0) {
                System.out.println ("---valore negativo o nullo!\n");
                k--;
              }
        }
        catch(Exception e){
             System.out.println ("---valore errato!\n");
              k--;
        }
     }     //------ fine for per controllo input di un singolo campo
    return rr;
 }       //------ fine metodo leggiDouble
        }       //---------------------- fine classe ProgCerchio
```

Il programma **ProgCerchio dichiara** 2 oggetti di tipo **Cerchio** : **tavolo** e **ruota**. Sia per **tavolo** che per **ruota**, dopo aver acquisito in input il raggio, **istanzia** l'oggetto e ne richiama i metodi per ottenere area e perimetro (arrotondati a 3 decimali).

Per acquisire il raggio si utilizza il metodo **privato** e statico **leggiDouble** (il main, che è un metodo statico, può richiamare solo metodi statici) a cui va passato il riferimento all'oggetto **tastiera** dichiarato nel main (specificandone la classe: BufferedReader).

Per digitazione di un valore corretto di tipo double, va fatto anche il controllo di che il valore non sia un numero negativo o zero (non corretto per una misura) utilizzando, anche nel **try**, la tecnica del decremento del contatore **k** del ciclo **for** di controllo in modo da ripetere la richiesta del raggio.