**Class Location**

A Location object keeps track of a location on a two-dimensional plane.

**Specification**

- **Constructor for the Location**
  
  ```java
  public Location(double xInitial, double yInitial)
  
  Construct a Location with specified coordinates.
  
  Parameters:
  
  xInitial – the initial x coordinate of this Location
  
  yInitial – the initial y coordinate of this Location
  
  Postcondition:
  
  This Location has been initialized at the given coordinates.
  
  ```

- **clone**
  
  ```java
  public Location clone()
  
  Generate a copy of this Location.
  
  Returns:
  
  The return value is a copy of this Location. Subsequent changes to the copy will not affect the original, nor vice versa. Note that the return value must be typecast to a Location before it can be used.
  
  ```

- **distance**
  
  ```java
  public static double distance<Location p1, Location p2>
  
  Compute the distance between two Locations.
  
  Parameters:
  
  p1 – the first Location
  
  p2 – the second Location
  
  Returns:
  
  The distance between p1 and p2.
  
  Note:
  
  The answer is Double.POSITIVE_INFINITY if the distance calculation overflows. The answer is Double.NaN if either Location is null.
  
  ```

- **equals**
  
  ```java
  public Boolean equals(Object obj)
  
  Compare this Location to another object for equality.
  
  Parameters:
  
  Obj – an object with which this Location is compared
  
  Returns:
  
  ```
A return value of true indicates that obj refers to a Location object with the same value as this Location. Otherwise, the return value is false.

Note:
If obj is null or is not a Location object, then the answer is false.

• **getX and getY**

  public double getX() -and- public double getY()

  Get the x or y coordinate of this Location.

  Returns:
  The x or y coordinate of this Location.

• **midpoint**

  public static Location midpoint(Location p1, Location p2)

  Generates and returns a Location halfway between two others.

  Parameters:
  p1 – the first Location
  p2 – the second Location

  Returns:
  A Location that is halfway between p1 and p2.

  Note:
  The answer is null if either p1 or p2 is null.

• **rotate90**

  public void rotate90()

  Rotate this Location 90° in a clockwise direction.

  Postcondition:
  This Location has been rotated clockwise 90° around the origin.

• **shift**

  public void shift(double xAmount, double yAmount)

  Move this Location by given amounts along the x and y axes.

  Postcondition:
  This Location has been moved by the given amounts along the two axes.

  Note:
  The shift may cause a coordinate to go above Double.MAX_VALUE or below – Double.MAX_VALUE. In these cases, subsequent calls to getX or getY will return Double.POSITIVE_INFINITY or Double.NEGATIVE_INFINITY.
- **toString**
  
  ```java
  public String toString()
  
  Generate a string representation of this Location.
  
  Returns:
  
  A string representation of this Location.
  ```