

Appendix C

Black Box Classes

This appendix contains summary class documentation for the `Environment` interface and the marine biology simulation utility classes covered in Chapters 1 – 4 (`Debug`, `Direction`, `EnvDisplay`, `Locatable`, `Location`, and `RandNumGenerator`).

Environment interface

```
public int numRows()
    Returns number of rows in this environment (-1 if the environment is unbounded).

public int numCols()
    Returns number of columns in this environment (-1 if the environment is unbounded).

public boolean isValid(Location loc)
    Returns true if loc is valid in this environment; otherwise returns false.

public int numCellSides()
    Returns the number of sides around each cell.

public int numAdjacentNeighbors()
    Returns the number of adjacent neighbors around each cell.

public Direction randomDirection()
    Generates a random direction. The direction returned by randomDirection reflects the
    direction from a cell in the environment to one of its adjacent neighbors.

public Direction getDirection(Location fromLoc, Location toLoc)
    Returns the direction from one location to another.

public Location getNeighbor(Location fromLoc, Direction compassDir)
    Returns the adjacent neighbor of a location in the specified direction (whether valid or invalid).

public java.util.ArrayList neighborsOf(Location ofLoc)
    Returns the adjacent neighbors of a specified location. Only neighbors that are valid locations in
    the environment will be included.

public int numObjects()
    Returns the number of objects in this environment.

public Locatable[] allObjects()
    Returns all the objects in this environment.

public boolean isEmpty(Location loc)
    Returns true if loc is a valid location in the context of this environment and is empty; false otherwise.

public Locatable objectAt(Location loc)
    Returns the object at location loc; null if loc is not in the environment or is empty.

public void add(Locatable obj)
    Adds a new object to this environment at the location it specifies.
    (Precondition: obj.location() is a valid empty location.)
```

`public void remove(Locatable obj)`
Removes the object from this environment.
(Precondition: `obj` is in this environment.)

`public void recordMove(Locatable obj, Location oldLoc)`
Updates this environment to reflect the fact that an object moved.
(Precondition: `obj.location()` is a valid location and there is no other object there.
Postcondition: `obj` is at the appropriate location (`obj.location()`), and either `oldLoc` is equal to `obj.location()` (there was no movement) or `oldLoc` is empty.)

Debug class

`public static boolean isOn()`
Checks whether debugging is on (not necessary when using `Debug.print` and `Debug.println`).

`public static boolean isOff()`
Checks whether debugging is off (not necessary when using `Debug.print` and `Debug.println`).

`public static void turnOn()`
Turns debugging on.

`public static void turnOff()`
Turns debugging off.

`public static void restoreState()`
Restores the previous debugging state. If there is no previous state to restore, `restoreState` turns debugging off.

`public static void print(java.lang.String message)`
Prints debugging message without appending a newline character at the end. If debugging is turned on, message is printed to `System.out` without a newline.

`public static void println(java.lang.String message)`
Prints debugging message, appending a newline character at the end. If debugging is turned on, message is printed to `System.out` followed by a newline.

Direction class

Public class constants: NORTH, NORTHEAST, EAST, SOUTHEAST, SOUTH, SOUTHWEST, WEST, NORTHWEST

Note: these are class constants of type `Direction` (e.g., `Direction.NORTH`) that represent compass directions in degrees (0°, 45°, 90°, 135°, 180°, 225°, 270°, 315°, respectively)

```
public Direction()
```

Constructs a default `Direction` object facing North.

```
public Direction(int degrees)
```

Constructs a `Direction` object — initial compass direction in degrees.

```
public Direction(java.lang.String str)
```

Constructs a `Direction` object — compass direction specified as a string, e.g., “North”.

```
public int inDegrees()
```

Returns this direction value in degrees.

```
public boolean equals(java.lang.Object other)
```

Indicates whether some other `Direction` object is “equal to” this one.

```
public int hashCode()
```

Generates a hash code for this direction (will not be tested on the AP Exam).

```
public Direction toRight()
```

Returns the direction that is a quarter turn to the right of this `Direction` object.

```
public Direction toRight(int deg)
```

Returns the direction that is `deg` degrees to the right of this `Direction` object.

```
public Direction toLeft()
```

Returns the direction that is a quarter turn to the left of this `Direction` object.

```
public Direction toLeft(int deg)
```

Returns the direction that is `deg` degrees to the left of this `Direction` object.

```
public Direction reverse()
```

Returns the direction that is the reverse of this `Direction` object.

```
public java.lang.String toString()
```

Returns a string indicating the direction.

```
public Direction roundedDir(int numDirections, Direction startingDir)
```

Rounds this direction to the nearest “cardinal” direction (will not be tested on the AP Exam).

```
public static Direction randomDirection()
```

Returns a random direction.

EnvDisplay interface

`public void showEnv()`
Shows the current state of the environment.

Locatable interface

`public Location location()`
Returns the location of this object.

Location class (implements Comparable)

`public Location(int row, int col)`
Constructs a `Location` object.

`public int row()`
Returns the row coordinate of this location.

`public int col()`
Returns the column coordinate of this location.

`public boolean equals(java.lang.Object other)`
Returns `true` if `other` is at the same row and column as the current location; `false` otherwise.

`public int hashCode()`
Generates a hash code for this location (will not be tested on the AP Exam).

`public int compareTo(java.lang.Object other)`
Compares this location to `other` for ordering. Returns a negative integer, zero, or a positive integer as this location is less than, equal to, or greater than `other`. Locations are ordered in row-major order.
(Precondition: `other` is a `Location` object.)

`public java.lang.String toString()`
Returns a string indicating the row and column of the location in (row, col) format.

RandNumGenerator class

`public static java.util.Random getInstance()`
Returns a random number generator. Always returns the same `Random` object to provide a better sequence of random numbers.