

2003 AP® COMPUTER SCIENCE A FREE-RESPONSE QUESTIONS

*This question is based on the 2003 AP Computer Science Free-Response Questions. The AP questions are copyright © 2003 College Entrance Examination Board. This version is **not endorsed** by the College Board.*

4. This question involves reasoning about the code from the Marine Biology Case Study. Please reference a copy of the code.

The marine biologists want to study a species of fish that eats algae. Any position in the environment grid can contain zero or more units of algae. If there is any algae at a fish's location, the fish eats one unit of algae and does not move; otherwise, the fish does not eat. If this is the third consecutive step in which the fish has not eaten, then the fish dies and is removed from the environment. If the fish does not eat and does not die, it moves to a position among its empty neighbors that contains the most algae.

We represent the algae by adding a matrix of integers to the private data of the BoundedEnv class. This matrix is the same size as myWorld, and each entry represents the number of units of algae at that location. We add two public methods to the Environment interface, as well as modifying the BoundedEnv constructor to initialize myAlgae.

```
//      Added these methods to the Environment interface

    public int numAlgaeAt(Location loc)
    //precondition:  loc is a valid Location in the environment
    //postcondition: returns the number of units of algae at loc

    public void removeAlgae(Location loc, int numUnits)
    //precondition:  algae at Location loc exceeds numUnits
    //postcondition: algae at Location loc has been reduced by numUnits

//      Added this data member to the BoundedEnv class

    private int[][] myAlgae;
```

2003 AP® COMPUTER SCIENCE A FREE-RESPONSE QUESTIONS

We modify the `Fish` class by adding a private data member to keep track of how long since the fish ate any algae. We also modify public method `act` so that it encapsulates all of the actions of a fish for one step of the simulation, and we modify the `move` function so that the fish moves to the position among its empty neighbors that has the most algae. In order to do this we add private method `mostAlgae` to the `Fish` class.

```
//      Added this method to the Fish class

    private Location mostAlgae(ArrayList nbrs)
    //precondition:  nbrs.size() > 0, this Fish is in Environment
    //              environment()
    //postcondition: returns a Location from nbrs that contains the
    //              most algae.

//      Modified these functions in the Fish class (changes are in bold)

    public void act()
    //precondition:  this Fish is stored in environment() at location()
    //postcondition:  if there was algae at location(), this fish
    //              has not moved and one unit of algae has been
    //              removed from location(); otherwise, if this was
    //              the third consecutive step that this Fish did not
    //              eat, then this fish has been removed from
    //              environment(); otherwise, this Fish moved.
    //              myStepsSinceFed has been updated.

    private void move()
    //postcondition:  this Fish tried to move to the adjacent space to
    //              location() with the greatest number of algae in
    //              environment().

//      Added this data member to the Fish class

    private int myStepsSinceFed; // steps since this fish last ate
```

- (a) Write the `BoundedEnv` method `numAlgaeAt`, which is described as follows.
`numAlgaeAt` should return the number of units of algae at `Location loc`.

Complete the function `numAlgaeAt` below.

```
public int numAlgaeAt(Location loc)
//precondition:  loc is a valid Location in the environment
//postcondition: returns the number of units of algae at loc
{

}

}
```

2003 AP® COMPUTER SCIENCE A FREE-RESPONSE QUESTIONS

- (b) Write the `Fish` method `mostAlgae`, which is described as follows. `mostAlgae` should return a `Location` from `nbrs` that contains the most algae. If more than one position contains the maximum amount, any of those positions may be returned.

In writing `mostAlgae`, you may use any of the `Environment` public methods, including `numAlgaeAt`. Assume that `numAlgaeAt` works as specified, regardless of what you wrote in part (a).

Complete the function `mostAlgae` below.

```
private Location mostAlgae(ArrayList nbrs)
//precondition:  nbrs.size() > 0, this Fish is in Environment
//              environment()
//postcondition: returns a Location from nbrs that contains the
//              most algae.
{

}

}
```

2003 AP® COMPUTER SCIENCE A FREE-RESPONSE QUESTIONS

- (c) Rewrite the `Fish` method `act`, which is described as follows. If there is algae at the fish's current `Location`, the fish should eat one unit of algae and not move. If there is no algae and this is the third consecutive step in which the fish has not eaten, `act` will cause the fish to die by calling the `Environment` method `remove`. If the fish does not eat and does not die, then the fish should move to a neighboring location with the most algae. `act` should update the state of the `Environment` and the state of the `Fish` appropriately.

In writing `act`, you may use any method from the Marine Biology Simulation Case Study, including those added at the beginning of the question. Assume that the `Fish` method `move` has been modified to work correctly and that the `Environment` method `numAlgaeAt` and the `Fish` method `mostAlgae` work as specified, regardless of what you wrote in parts (a) and (b).

Complete function `act` below.

```
public void act()  
//precondition:  this Fish is stored in environment() at location()  
//postcondition: if there was algae at location(), this fish  
//              has not moved and one unit of algae has been  
//              removed from location(); otherwise, if this was  
//              the third consecutive step that this Fish did not  
//              eat, then this fish has been removed from  
//              environment(); otherwise, this Fish moved.  
//              myStepsSinceFed has been updated.  
{
```

```
}
```