CS202

Assignment 5: Pond Simulator

Description

For this assignment we implement a pond simulator. We maintain a ”2D” array of objects (fish or plants) that represent the aquatic life in the simulation. The idea is fish swim around an consume plants to try an stay alive, after each week based on each fish’s consumption of plants the fish either grows or shrinks, once its weight becomes 0 then fish would die, same concept for plants once their weight is 0 they die. However plants do not need to consume anything to grow, they just need water and sun, yeah I guess that’s all plants really need, maybe electrolytes too? I think plants crave that too..anyway plants keep growing and only lose weight when a fish consumes a portion of them.

We use inheritance once again along with virtual functions to implement this simulation since all animals and plants are organisms and all fish (herbivores) are animals and thus are all organisms, we can maintain a pointer of an organism type and have that point to a fish object (herbivore object) or plant object. And we use virtual functions to call the correct functions that are overridden in derived classes and we make use of dyanmic casting to downcast pointers to determine the type of objects any organism pointer points to.

So each object is contained in an element of the ”2D” array, so they can be thought of being in a shape of a box, so this is sort of like minecraft. The simulation essentially randomly picks two elements in the grid and attempts to match two organisms to perform an action, and we do this several times to simulate a week in the pond over a span of several weeks. The rest of the pages goes over the more technical details.

Organism Class Description

```
class organism
{
public:
  organism(double = 1, double = 0);
  virtual void simulateWeek();
  void assignRate(double);
  void alterSize(double);
  void death();
  double getSize() const;
  double getRate() const;
  bool isAlive() const;
  virtual ~organism();
protected:
  double growthRate;
private:
  double size;
};
```

The members of `organism` class are described below

- `double growthRate` denotes the growth rate of the organism
- `double size` denotes the size of the organism
- `organism::organism(double initSize, double initRate)` is the constructor that sets `growthRate` and `size` with the values passed in
- `void organism::simulateWeek()` increments the `size` by `growthRate`
- `void organism::assignRate(double newRate)` assigns `growthRate` with the value passed in
- `void organism::alterSize(double amount)` increases the `size` by `amount`
- `void organism::death()` sets `growthRate` and `size` to 0
- `double organism::getSize() const` returns `size`
- `double organism::getRate() const` returns `growthRate`
- `bool organism::isAlive() const` returns `true` if `size` is larger than 0 and returns `false` otherwise
- `organism::~organism()` - sets `growthRate` and `size` to 0

Plant Class Description

```
class plant : public organism
{
public:
  plant(double = 1, double = 0);
  void nibbledOn(double);
  ~plant();
};
```

The members of `plant` class are described below

- `plant::plant(double initSize, double initRate)` is the constructor that sets `size` and `growthRate` with the values passed in
- `void plant::nibbledOn(double amount)` decreases the `size` by `amount`, if the `amount` is larger than the `size`, then set the `size` to 0 (by decreasing the `size` such that the end result becomes 0)
- `plant::~plant()` - outputs that `PLANT DIED`

Animal Class Description

```
class animal : public organism
{
public:
  animal(double = 1, double = 0, double = 0);
  void assignNeed(double);
  void eat(double);
  void simulateWeek();
  double stillNeed() const;
  double totalNeed() const;
  ~animal();
private:
  double needThisWeek;
  double eatenThisWeek;
};
```

The members of `animal` class are described below

- `double needThisWeek` denotes the amount of food that needs to be consumed in a week
- `double eatenThisWeek` denotes the amount of food that was eaten in the current week
- `animal::animal(double initSize, double initRate, double initNeed)` is the constructor that sets `size`, `growthRate`, and `needThisWeek` with the values assigned, and sets `eatenThisWeek` to 0
- `void animal::assignNeed(double newNeed)` reassigns `needsThisWeek` to the value passed in
- `void animal::eat(double amount)` increments `eatenThisWeek` by `amount`
- `void animal::simulateWeek()` - the function does the following
  1. Changes the sign on `growthRate` based on whether the value returned by `stillNeed()`; sets the sign to negative if `stillNeed()` returns a positive number and sets the sign to positive if `stillNeed()` returns 0
  2. Then the function calls organism's `simulateWeek()` function, sets `eatenThisWeek` to 0, checks if `stillAlive()` if no longer alive then call `death()` function
- `double animal::stillNeed() const` returns 0 if `eatenThisWeek` is greater than or equal to `needThisWeek`, and returns the difference between `needThisWeek` and `eatenThisWeek` otherwise
- `double animal::totalNeed() const` returns `needThisWeek`
- `animal::~animal()` - sets `needThisWeek` and `eatenThisWeek` to 0

Herbivore Class Description

```
class herbivore : public animal
{
public:
  static const double PORTION;
  static const double MAX_FRACTION;
  herbivore(double = 1, double = 0, double = 0);
  void nibble(plant&);
  ~herbivore();
};
```

The members of `herbivore` class are described below

- `const double herbivore::PORTION` will be set to 0.5 (in the implementation file, the organism.cpp file), this regulates how big of a portion of the `plant` object it will consume
- `const double herbivore::MAX_FRACTION` will be set to 0.1 (in the implementation file, the organism.cpp file), this regulates how big of a portion the `herbivore` can consume in one feeding (when `nibble` is called)
- `herbivore::herbivore(double initSize, double initRate, double initNeed)` is the constructor that sets `size`, `growthRate`, `needThisWeek` with the values passed in and sets `eatenThisWeek` to 0
- `void herbivore::nibble(plant& meal)` is a function that consumes a portion of a `plant` object, the amount can be computed in the following way

  1. amount is set to `PORTION` multiplied by size of the `meal` object
  2. if the amount is larger than `MAX_FRACTION` multiplied by `needThisWeek`, then set amount to `MAX_FRACTION` multiplied by `needThisWeek`
  3. if the amount is larger than the value returned by `stillNeed()`, then set amount to `stillNeed()`
  4. Call `eat` and the `meal` object's `nibbledOn` functions and pass the same amount into both functions

- `herbivore::~herbivore()` - outputs `FISH DIED`

RandNum Class Description

```
class randNum
{
public:
  randNum(string);
  int getNextRand();
private:
  ifstream infile;
};
```

The members of `randNum` class are described below

- `ifstream infile` - filestream where the random numbers are read
- `randNum::randNum(string filename)` - constructor that opens the filestream
- `int randNum::getNextRand()` - returns the next random number from the file

Contents of main

The `int main()` is written for you, however you will need to implement the following functions

- `void buildPondSimulator(ifstream& infile, organism *** pond)` - this function takes in an already opened filestream that reads from a CSV file and your empty "2D" array of `organism` objects, your job will be to insert an `herbivore` or `plant` object into `pond[r][c]`. Every line in the CSV file will contain

  ```
  ORGANISM_TYPE,SIZE,GROWTH_RATE,X_COORDINATE,Y_COORDINATE
  ```

  Once you parse the line and convert each element into its correct type (`SIZE` and `GROWTH_RATE` should be converted in `double` and `X_COORDINATE` and `Y_COORDINATE` should be converted into `int`, and `ORGANISM_TYPE` will remain a `string`), you then assign `pond[X_COORDINATE][Y_COORDINATE]` with `new herbivore(size, rate, size * 0.1)` or `new plant(size, rate)` based on whether `ORGANISM_TYPE` contains `"FISH"` or `"PLANT"` respectively.

  You may use `getNextField` function from previous assignments or you can use a `stringstream` type to parse each comma separated line. You can see a description of that in the video (it's linked at the end of the pdf).

  Not all elements of `pond[r][c]` will be assigned to point to an object, thus for those elements assign `pond[r][c] = NULL` or `pond[r][c] = nullptr`

- `void simulateAWeek(organism *** pond, randNum& rN)` - this function simulates one week of the pond simulator, you will simulate 100 activities and each activity does the following

  1. Get a random x_1, y_1, x_2, and y_2 indices to investigate two potential organisms in the pond array
  2. Create two pointers `organism * o1, * o2` and set them with the content in pond$[x_1][y_1]$ and pond$[x_2][y_2]$ respectively
  3. Check the contents in `o1` and `o2` for whether they are `NULL` or if they point to an `herbivore` and/or `plant` object (this could involve using a `dynamic_cast`)
  4. If they are both `NULL` then this is a failed activity, thus nothing happens
  5. If they both point to `plant` objects, then this would be a failed activity, thus nothing happens
  6. If one of the pointers is `NULL` and the other is a `plant` then nothing happens
  7. If one of the pointers is `NULL` and the other points to an `herbivore`, assign `NULL` to where the `herbivore` object used to be be and assign the `herbivore` obect to where the `NULL` used to be
  8. If they both point to `herbivore` objects then you need to swap their positions in the pond array
  9. If one of the pointers points to an `herbivore` object and the other points to a `plant` object, then you must called the `herbivore` object's nibble function and pass in the `plant` object as the parameter (you may need to dereference, use the arrow operator, and/or use `dynamic_cast` here), then you swap their positions in the pond array (I guess the plant submerges underground and spawns where the fish used to be)

  Once all 100 activities are simulated, you call `simulateWeek()` function for each element. Check each object if it is still alive, if no longer alive then deallocate the object `pond[r][c]` and then set `pond[r][c]` to `NULL` or `nullptr`

  Then you call the outputOrganism function for each object in the pond.

- `void outputOrganism(organism * org)` - Outputs the string `"Fish with weight"` or `"Plant with weight"` depending on whether `org` points to an `herbivore` or `plant` object and then output `org->getSize()`

- `void clearSimulation(organism *** pond)` - deallocates all the objects in each `pond[r][c]` position and then deallocates the pointers as well (this part would be the same as deallocating any 2D dynamic array)

Specifications

- Comment your code and your functions
- Do not add extra class members or remove class members and do not modify the member functions of the class
- No global variables (global constants are ok)
- Make sure your program is memory leak free

Sample Run

```
$ make
g++    -c -o organism.o organism.cpp
g++    -c -o animal.o animal.cpp
g++    -c -o plant.o plant.cpp
g++    -c -o herbivore.o herbivore.cpp
g++    -c -o randNum.o randNum.cpp
g++    -c -o main.o main.cpp
g++ organism.o animal.o plant.o herbivore.o randNum.o main.o
-Wall -pedantic -o PondSimulator
$ ./PondSimulator
Enter pond data file: PondData.csv
Enter amount of weeks for the simulation: 20
WEEK 1 RESULTS
Fish With Weight 41.6
Fish With Weight 46.7
Plant weight 15.3
Fish With Weight 48.5
Plant weight 22.61
Plant weight 19.28
Fish With Weight 35.1
Fish With Weight 31.2
Fish With Weight 23.1
Plant weight 10.38
Fish With Weight 33.8
Plant weight 10.4
Plant weight 23.67
Fish With Weight 17.5
Fish With Weight 48.3
Plant weight 17.26
Plant weight 9.88
Plant weight 22.41


WEEK 2 RESULTS
Plant weight 22.96
Fish With Weight 22.2
Plant weight 10.48
```

```
Fish With Weight 44.4
Fish With Weight 16
Fish With Weight 47
Fish With Weight 29.4
Plant weight 10.4
Plant weight 19.58
Plant weight 14.19
Plant weight 24.56
Fish With Weight 45.6
Plant weight 10.86
Plant weight 23.93
Plant weight 18.1
Fish With Weight 39.2
Fish With Weight 31.6
Fish With Weight 33.2


WEEK 3 RESULTS
Plant weight 25.76
Plant weight 19.08
Fish With Weight 27.6
Fish With Weight 42.1
Plant weight 14.12
Plant weight 19.2
Fish With Weight 45.5
Plant weight 9.51
Plant weight 10.56
Plant weight 26.3
Fish With Weight 31.3
Fish With Weight 42.9
Fish With Weight 29.4
Fish With Weight 36.8
Fish With Weight 21.3
Plant weight 10.42
Fish With Weight 14.5
Plant weight 24.14


WEEK 4 RESULTS
Plant weight 10.86
Fish With Weight 13
Plant weight 19.96
Plant weight 14.61
Plant weight 24.46
Plant weight 10.61
Plant weight 9.57
Fish With Weight 27.2
Fish With Weight 39.8
Fish With Weight 34.4
Fish With Weight 40.2
Fish With Weight 20.4
Fish With Weight 29.4
Fish With Weight 44
Fish With Weight 25.8
```

```
Plant weight 27.24
Plant weight 27.7
Plant weight 19.11


WEEK 5 RESULTS
Fish With Weight 19.5
Plant weight 27.93
Plant weight 11.41
Plant weight 8.91
Fish With Weight 37.5
Fish With Weight 27.5
Plant weight 26.56
Fish With Weight 24
Fish With Weight 37.5
Plant weight 20.12
Fish With Weight 32
Plant weight 11.02
Plant weight 29.25
Plant weight 21.15
Fish With Weight 42.5
Plant weight 15.92
Fish With Weight 25
Fish With Weight 11.5


WEEK 6 RESULTS
Plant weight 19.85
Plant weight 30.01
Plant weight 11.98
Fish With Weight 41
Fish With Weight 22.8
Plant weight 27.71
Fish With Weight 29.6
Plant weight 16.93
Fish With Weight 22.2
Fish With Weight 25.6
Plant weight 28.16
Fish With Weight 18.6
Fish With Weight 10
Plant weight 9.49
Fish With Weight 34.8
Plant weight 11.18
Plant weight 21.4
Fish With Weight 35.2


WEEK 7 RESULTS
Plant weight 18.1
Fish With Weight 23.7
Fish With Weight 27.2
Plant weight 29.11
Fish With Weight 32.1
Fish With Weight 20.4
```

```
Plant weight 11.85
Fish With Weight 8.5
Plant weight 11.92
Fish With Weight 32.9
Plant weight 30.36
Plant weight 9.24
Fish With Weight 17.7
Plant weight 22.07
Fish With Weight 39.5
Fish With Weight 20.6
Plant weight 30.69
Plant weight 20.72


WEEK 8 RESULTS
Plant weight 32.32
Fish With Weight 38
Fish With Weight 16.8
Fish With Weight 30.6
Plant weight 11.69
Fish With Weight 24.8
Fish With Weight 18.6
Fish With Weight 21.8
Plant weight 9.46
Plant weight 22.82
Plant weight 31.1
Fish With Weight 29.4
Plant weight 30.69
Plant weight 17.41
Plant weight 13.16
Fish With Weight 7
Fish With Weight 18.4
Plant weight 21.18


WEEK 9 RESULTS
Fish With Weight 19.9
Plant weight 22.65
Plant weight 18.42
Plant weight 32.28
Plant weight 13.79
Fish With Weight 15.9
Plant weight 10.37
Plant weight 22.16
Fish With Weight 26.7
Fish With Weight 16.2
Fish With Weight 5.5
Fish With Weight 16.8
Plant weight 12.29
Fish With Weight 28.3
Fish With Weight 36.5
Plant weight 32.93
Fish With Weight 22.4
Plant weight 32.13
```

```
WEEK 10 RESULTS
Plant weight 11.88
Plant weight 14.54
Fish With Weight 15
Plant weight 23.17
Fish With Weight 24
Plant weight 9.68
Plant weight 34.19
Plant weight 18.57
Fish With Weight 4
Fish With Weight 15
Fish With Weight 18
Fish With Weight 14
Fish With Weight 35
Fish With Weight 26
Plant weight 32.86
Fish With Weight 20
Plant weight 33.05
Plant weight 23


WEEK 11 RESULTS
Plant weight 9.73
Plant weight 24.26
Fish With Weight 23.7
Plant weight 18.67
Fish With Weight 2.5
Fish With Weight 13.2
Plant weight 34.26
Fish With Weight 21.3
Fish With Weight 16.1
Fish With Weight 11.8
Fish With Weight 33.5
Plant weight 12.18
Fish With Weight 17.6
Plant weight 34.08
Fish With Weight 14.1
Plant weight 15.11
Plant weight 23.07
Plant weight 33.63


WEEK 12 RESULTS
Plant weight 23.16
Plant weight 33.95
Plant weight 9.66
Fish With Weight 11.4
Plant weight 15.73
Fish With Weight 1
Plant weight 25.26
Plant weight 34.4
Fish With Weight 18.6
```

```
Plant weight 12.68
Fish With Weight 32
Fish With Weight 13.2
Plant weight 35.53
Fish With Weight 9.6
Fish With Weight 15.2
Plant weight 17.81
Fish With Weight 21.4
Fish With Weight 14.2


WEEK 13 RESULTS
Plant weight 9.11
Fish With Weight 15.9
FISH DIED
Fish With Weight 30.5
Plant weight 23.27
Plant weight 36.31
Plant weight 12.75
Fish With Weight 19.1
Fish With Weight 9.6
Fish With Weight 12.8
Fish With Weight 12.3
Plant weight 37.63
Fish With Weight 12.3
Fish With Weight 7.4
Plant weight 17.04
Plant weight 35.72
Plant weight 18.58
Plant weight 25.7


WEEK 14 RESULTS
Fish With Weight 7.8
Plant weight 16.94
Fish With Weight 5.2
Plant weight 12.21
Plant weight 8.7
Plant weight 18.71
Plant weight 25.02
Plant weight 37.8
Fish With Weight 10.4
Fish With Weight 13.2
Fish With Weight 29
Plant weight 38.88
Plant weight 23.21
Fish With Weight 11.4
Plant weight 36.56
Fish With Weight 16.8
Fish With Weight 10.4


WEEK 15 RESULTS
Plant weight 16.6
Plant weight 23.76
Plant weight 39.4
Fish With Weight 27.5
Plant weight 36.33
Plant weight 24.45
Plant weight 9.47
Fish With Weight 3
Plant weight 12.56
Fish With Weight 14.5
Fish With Weight 10.5
Plant weight 40.54
Fish With Weight 8.5
Fish With Weight 6
Fish With Weight 10.5
Plant weight 18.72
Fish With Weight 8


WEEK 16 RESULTS
Fish With Weight 0.8
Plant weight 24.61
Plant weight 17.04
Fish With Weight 26
Plant weight 18.42
Plant weight 12.16
Fish With Weight 5.6
Plant weight 37.43
Plant weight 42.14
Plant weight 8.64
Fish With Weight 6.6
Fish With Weight 4.2
Plant weight 39.63
Plant weight 24.83
Fish With Weight 7.8
Fish With Weight 12.2
Fish With Weight 9.6


WEEK 17 RESULTS
Plant weight 25.09
Fish With Weight 4.7
Fish With Weight 8.7
Plant weight 18.21
Fish With Weight 2.4
FISH DIED
Plant weight 38.4
Fish With Weight 3.2
Fish With Weight 5.1
Plant weight 8.24
Plant weight 26.26
Fish With Weight 24.5
Plant weight 18.6
Fish With Weight 9.9
Plant weight 43.08
Plant weight 40.85
Plant weight 12.9


WEEK 18 RESULTS
Fish With Weight 23
Plant weight 41.97
Plant weight 12.56
Fish With Weight 0.6
Plant weight 8.15
Plant weight 19.29
Fish With Weight 2.8
Fish With Weight 2.4
Fish With Weight 7.8
Plant weight 44.19
Plant weight 26.1
Plant weight 17.85
Fish With Weight 0.8
Fish With Weight 7.6
Plant weight 39.67
Plant weight 27.56


WEEK 19 RESULTS
Fish With Weight 21.5
Fish With Weight 6.9
Plant weight 43.83
Plant weight 45.79
Plant weight 18.98
Plant weight 8.88
FISH DIED
Plant weight 19.79
Plant weight 27.23
FISH DIED
Fish With Weight 0.9
Fish With Weight 5.3
Plant weight 13.22
Plant weight 40.35
FISH DIED
Plant weight 28.5


WEEK 20 RESULTS
Fish With Weight 20
Plant weight 13.95
Plant weight 21.05
Plant weight 45.19
Plant weight 19.74
Plant weight 29.69
Plant weight 46.15
FISH DIED
Plant weight 9.49
Fish With Weight 6
Plant weight 28
Plant weight 42.55
Fish With Weight 3


Ending simulation
FISH DIED
PLANT DIED
PLANT DIED
PLANT DIED
PLANT DIED
PLANT DIED
PLANT DIED
PLANT DIED
FISH DIED
PLANT DIED
PLANT DIED
FISH DIED
```

Submission

Compress organism.h, organism.cpp, plant.h, plant.cpp, animal.h, animal.cpp, herbivore.h, herbivore.cpp, main.cpp, and makefile and upload to the canvas site by the deadline

References

- Link to the top image can be found at $https://custom-cursor.com/en/collection/spongebob/sb-fred-my-leg$
- Supplemental video part 1 https://youtu.be/Bvwr9Oczpug
- Supplemental video part 2 https://youtu.be/BcuL7g18-Vc