Platypus.cpp

```
#include <iostream>
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
using namespace std;

#include "platypus.h"
platypus::platypus()                              //create default constructor
        :alive(0),mutant(0), weight(0.0), age(0), name('0'), gender('0')
{
}
platypus:: platypus(char name, char gender, float weight, short age) //create constructor
{
        this->name = name;
        this->age = age;
        this->weight = weight;
        this->gender = gender;
        this->alive = 1;
        this->mutant = 0;

}

ostream& operator<<(ostream& out, platypus a)
{
        a.print(out);
        return out;
}
void platypus::print(ostream& out)//create output
{
        cout << "name  " << "age  " << "gender   " << "weight   " << "mutant  "<<"alive
"<< endl;
        out << name << "      " << age << "      " << gender << "       " << weight << "
" <<mutant<<"      "<<alive<< endl;
}
void platypus::age_me()
{
        float deadChance;
        if (this->alive == 1)   // if platypus dead you can't increase age.
        {
                srand(time(NULL));
                this->age = this->age + 1;
                if ((rand() % 100 + 1) <= 2) //2% chance to become mutant
                        this->mutant = 1;
                else
                        this->mutant = 0;
                deadChance = 10 * weight;//dead chance is ten times platypus weight
                if ((rand() % 100 + 1) <= deadChance)
                {
                        this->alive = 0;
                }
                else
                {
                        this->alive = 1;
                }
```

```
       }
}
void platypus::fight(platypus& other)
{
       if (this->alive == 1 && other.alive == 1)
       {
              srand(time(NULL));
              float ratio;
              ratio = (this->weight / other.weight) * 50;
              if ((rand() % 100 + 1) < ratio) //if random number is less then ratio then
"other" wins
              {
                     other.alive = 1;
                     this->alive = 0;
              }
              else
              {
                     other.alive = 0;
                     this->alive = 1;
              }


       }
}
void platypus ::eat()
{
       srand(time(NULL));
       if (this->alive == 1)
       {
          float randomNum;
          randomNum = (rand() % 50 + 1) / 10;
          this->weight = this->weight + this->weight * (randomNum / 100);
       }

}
void platypus::hatch()
{
       this->alive = 1;
       this->age = 0;
       this->mutant = 0;
       srand(time(NULL));
       if (rand() % 1 == 0)
              this->gender = 'M';
       else
              this->gender = 'F';
       this->weight = (float)(rand() % 10 + 1) / 10;
       this->name = 'a' + rand() % 26;
}
```

Platypus.h

```cpp
#pragma once
#include <iostream>
#include <string>
#include <ctime>


#ifndef PLATYPUS
#define PLATYPUS
using namespace std;

class platypus
{
private:
        // data members
        float weight;
        short age;
        char name;
        char gender;
        bool alive;
        bool mutant;
public:
        //constructors
        platypus();
        platypus(char name, char gender, float weight, short age);
        //mutators
        void age_me();
        void fight(platypus & other);
        void eat();
        void hatch();
        //output
        void print(ostream & out);

};
ostream& operator<<(ostream& out, platypus a);
#endif // !PLATYPUS
```

Drive.cpp

```cpp
#include <iostream>
#include "platypus.h"

using namespace std;

int main()
{
	platypus p1('m', 'F', 1.4, 5); //create platypus1
	cout << "platypus1" << endl;
	p1.print(cout);
	platypus p2;//create default platypus2
	cout << "platypus2" << endl;
	p2.print(cout);
	cout << "hatch platypus2" << endl;
	p2.hatch(); //hatch platypus2
	p2.print(cout);
	p2.age_me();
	p2.eat();
	cout << "fight results:" << endl;
	p1.fight(p2);//fight platypus1 and platypus2
	cout << "platypus1" << endl;
	p1.print(cout);
	cout << "platypus2" << endl;
	p2.print(cout);

	//you can increase age and feed alive platypus but not dead one
	cout << "increase age and feed platypus1:" << endl;
	cout << "platypus1"<<endl;
	p1.age_me();
	p1.eat();
	p1.print(cout);
	cout << "increase age and feed platypus2:" << endl;
	cout << "platypus2" << endl;
	p2.age_me();
	p2.eat();
	p2.print(cout);

	cout << "fight each other dead platypus and alive one" << endl;
	p1.fight(p2); // you can't fight each other if one is dead platypus.
	p1.print(cout);
	p2.print(cout);



}
```